ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка Unity-модуля генерації 3D- моделей по текстовому запиту за допомогою методів штучного інтелекту »

на здобуття освітнього ступеня бакалавра
зі спеціальності 122 Комп'ютерні науки
(код, найменування спеціальності)
освітньо-професійної програми Штучний інтелект
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Максим ПАВЛЮК
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконала: здобувачка вищої освіти гр ШІД-41
Максим ПАВЛЮК

Керівник: Тетяна Кисіль
Старший викладач, Старший викладач

Рецензент:
науковий ступінь, вчене звання _____
Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Штучного інтелекту

Ступінь вищої освіти Бакалавр

Спеціальність 122 Комп'ютерні науки

Освітньо-професійна програма Штучний інтелект

ЗАТВЕРДЖУЮ
Завідувач кафедрою Штучного інтелекту

_____ Ольга ЗІНЧЕНКО
«_____» _____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Павлюку Максиму Ігоровичу

1. Тема кваліфікаційної роботи: Розробка Unity-модуля генерації 3D- моделей по текстовому запиту за допомогою методів штучного інтелекту

керівник кваліфікаційної роботи Тетяна КИСІЛЬ старший викладач,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» 02.2024р. № 36

2. Строк подання кваліфікаційної роботи «31» травня 2024р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, вимоги до ігрової сцени.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження та аналіз можливостей штучного інтелекту в розробці ігор
2. Дослідження обчислень та генерації 3D моделі
3. Інструмент для генерації

5. Перелік графічного матеріалу: *презентація*
   1. Аналіз можливостей штучного інтелекту в розробці ігор
   2. Дослідження обчислень та генерації 3D-моделі
   3. Модуль генерації 3D-моделі

6. Дата видачі завдання«27» лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	27.02-05.11.23	Виконав
2	Вивчення матеріалів для аналізу розвитку генерації 3Д-моделей	05.11-12.11.23	Виконав
3	Дослідження нейромереж	13.11-19.11.23	Виконав
4	Аналіз особливостей моделей створень нейромереж	20.11-25.11.23	Виконав
5	Дослідження технологій машинного навчання	27.11-03.12.23	Виконав
6	Застосування штучного інтелекту в проєкті	04.12-10.12.23	Виконав
7	Оформлення роботи: вступ, висновки, реферат	11.12-20.12.23	Виконав
8	Розробка демонстраційних матеріалів	21.12-31.05.24	Виконав

Здобувачка вищої освіти _____ Максим ПАВЛЮК

Керівник
кваліфікаційної роботи _____ Тетяна КИСІЛЬ

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 62 стор., 3 табл., 24 рис., 18 джерел.

Мета роботи - прискорення процесу створення 3D -моделей в ігровому рушію Unity за допомогою штучного інтелекту

Об'єкт дослідження - процес проскорення створення 3D -моделей

Предмет дослідження - штучний інтелект та нейромережі

Короткий зміст роботи: У роботі проведено дослідження методів генерації нейромереж та створення модулю для генерації 3D моделей. Ключовою ідеєю цього дослідження полягало в отриманні знань про різні проблемні питання генерації 3D моделей з 2D зображень. Кінцевий результат цієї роботи полягав в пришвидченні та спрощені процесу розробки 3D моделей для розробників

КЛЮЧОВІ СЛОВА: ШТУЧНИЙ ІНТЕЛЕКТ, НЕЙРОМЕРЕЖІ, GRAPH NEURAL NETWORKS, NVIDIA, DREAMFUSION, ГЕНЕРАЦІЯ 3Д-МОДЕЛЕЙ

ЗМІСТ

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ МОЖЛИВОСТЕЙ ШТУЧНОГО ІНТЕЛЕКТУ В РОЗРОБЦІ ІГОР 1
1.1 Багатоплатформний інструмент та ігровий рушій Unity 1
1.2 Дослідження існуючих нейронних мереж Midjourney 3
1.3 Можливості штучного інтелекту від Unity Muse 12
2 ДОСЛІДЖЕННЯ ОБЧИСЛЕНЬ ТА ГЕНЕРАЦІЇ 3Д МОДЕЛІ 16
2.1 Ознайомлення з підходом створення 16
2.2 Методологія 20
2.3 Впровадження 29
2.4 Подальші дослідження 34
2.6 Перетворення тексту в зображення 37
2.7 Генеративні 3D-моделі 38
2.11 Моделі сцен 42
2.12 Оптимізація від грубої до точної 42
2.13 Керована генерація 3D 44
2.14 Персоналізований текст у 3D 45
2.15 Оперативне редагування на основі точного налаштування 46
3 ІНСТРУМЕНТ ДЛЯ ГЕНЕРАЦІЇ 48
3.1 Graph Neural Networks 48
Переваги використання GNN 48
3.2 Інтеграція Meshy та створення інструменту 50
3.3 Показ інструменту 53
ПЕРЕЛІК ПОСИЛАНЬ 57
ДОДАТКИ 59
ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛЯ 59
ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ 61

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ МОЖЛИВОСТЕЙ ШТУЧНОГО ІНТЕЛЕКТУ В РОЗРОБЦІ ІГОР

1.1 Багатоплатформний інструмент та ігровий рушій Unity

У сфері розробки ігор і не тільки, Unity є потужним програмним забезпеченням, пропонуючи універсальну платформу для творців, щоб матеріалізувати своє бачення в інтерактивний досвід. Незалежно від того, чи ви початківець-розробник ігор, архітектор, який візуалізує проекти, чи викладач, який створює імерсивні навчальні модулі, можливості Unity роблять його незамінним інструментом для втілення ідей у життя.

Unity - це кросплатформний ігровий рушій, розроблений компанією Unity Technologies, вперше випущений у 2005 році. З роками він перетворився на всеосяжну екосистему, що охоплює не лише розробку ігор, але й застосування в таких галузях, як архітектура, кіно, автомобільна промисловість тощо. В основі Unity лежить зручний інтерфейс у поєднанні з потужними інструментами та функціоналом, що робить його доступним як для досвідчених розробників, так і для початківців.

Можливості Unity:

1. Кросплатформна розробка: Однією з ключових переваг Unity є можливість безперешкодно розгортати проекти на різних платформах. Незалежно від того, чи націлені ви на мобільні пристрої, консолі, ПК або навіть на нові платформи, такі як VR та AR, Unity спрощує процес, дозволяючи розробникам охопити ширшу аудиторію з мінімальними зусиллями.
2. Багате візуальне середовище: Unity дозволяє творцям створювати візуально приголомшливі світи завдяки потужним можливостям рендерингу. Від реалістичних ефектів освітлення і затінення до динамічних систем частинок

3. і ефектів постобробки, Unity надає інструменти, необхідні для створення захоплюючих вражень.
4. Гнучке написання сценаріїв: Завдяки підтримці багатьох мов програмування, включаючи C#, Unity пропонує гнучке середовище написання сценаріїв, яке підходить для розробників різного рівня кваліфікації. Незалежно від того, чи ви досвідчений програміст, чи новачок у програмуванні, скриптові можливості Unity дозволять вам з легкістю втілювати свої ідеї в життя.
5. Сховище ресурсів: Unity Asset Store слугує скарбницею ресурсів, пропонуючи величезну колекцію готових ресурсів, скриптів, плагінів та інструментів, створених як Unity Technologies, так і спільнотою. Ця велика бібліотека не тільки прискорює розробку, але й сприяє співпраці та інноваціям в екосистемі Unity.
6. Фізика та моделювання: Вбудований фізичний рушій Unity дозволяє розробникам моделювати реалістичні взаємодії у своїх віртуальних світах. Незалежно від того, чи це імітація руху об'єктів, реалізація складних зіткнень або створення реалістичної анімації, фізична система Unity забезпечує основу для створення динамічного та інтерактивного досвіду.
7. Спільна розробка: Функції спільної роботи в Unity спрощують процес розробки, дозволяючи декільком членам команди працювати над одним проектом одночасно. Незалежно від того, співпрацюєте ви з колегами в одному офісі чи в різних куточках світу, хмарні інструменти Unity сприяють безперебійній комунікації та контролю версій, гарантуючи, що всі залишаються на одній сторінці протягом усього циклу розробки.

Застосування Unity:

8. Розробка ігор: Unity є синонімом розробки ігор, завдяки якому створюється величезна кількість ігор у різних жанрах і на різних платформах. Від інді-розробників до студій AAA, Unity надає інструменти та ресурси, необхідні для створення всього - від 2D-платформерів до масштабних пригод у відкритому світі.

9. Доповнена і віртуальна реальність: Unity відіграє ключову роль у зростаючих сферах доповненої та віртуальної реальності, дозволяючи розробникам створювати захоплюючий досвід, який стирає межу між віртуальним та фізичним світом. Незалежно від того, чи це створення інтерактивних AR-додатків для смартфонів, чи розробка VR-симуляцій для навчання та освіти, можливості Unity знаходяться на передовій цих нових технологій.
10. Архітектурна візуалізація: Архітектори та дизайнери використовують можливості рендерингу Unity у реальному часі для візуалізації архітектурних проектів у віртуальному середовищі. Від візуалізацій пропонованих будівель до інтерактивних 3D-моделей інтер'єрів, Unity дозволяє архітекторам ефективно презентувати свої проекти та досліджувати різні концепції в цифровому просторі.
11. Освіта та навчання: Unity все частіше використовується в освітніх установах для створення інтерактивного навчального процесу. Від інтерактивних симуляцій та віртуальних лабораторій до гейміфікованих освітніх додатків, Unity надає освітянам потужну платформу для залучення студентів та сприяння навчанню з ефектом занурення.

Універсальність та потужний набір функцій Unity роблять його наріжним каменем сучасної інтерактивної розробки у багатьох галузях. Незалежно від того, чи ви розробник ігор, який розширює межі інтерактивних розваг, чи архітектор, який візуалізує майбутнє міських ландшафтів, Unity дає можливість творцям втілювати свої бачення в реальність. З розвитком технологій Unity залишається на передовій, стимулюючи інновації та формуючи майбутнє інтерактивного досвіду.

1.2 Дослідження існуючих нейронних мереж Midjourney

Midjourney - це надійна платформа, яка слугує комплексним рішенням для бізнесу та приватних осіб, які прагнуть створювати захоплюючий віртуальний досвід. Розроблена компанією Midjourny Technologies, ця платформа пропонує

широкий спектр можливостей, пристосованих для різних галузей, включаючи нерухомість, туризм, освіту тощо.

Midjourney - це передова платформа віртуального досвіду, призначена для спрощення створення і розгортання контенту з ефектом присутності.

Незалежно від того, чи демонструєте ви об'єкти нерухомості, просуваєте напрямки або покращуєте освітній досвід, Midjourney надає інструменти та функціональні можливості, необхідні для захоплення аудиторії та стимулювання залученості.

Можливості Midjourney:

1. Віртуальні тури: Midjourney дозволяє користувачам створювати інтерактивні віртуальні тури, які дозволяють глядачам досліджувати об'єкти нерухомості, напрямки або освітні середовища з комфорту власного пристрою. Від панорамних видів та інтерактивних точок до екскурсій та інформаційних накладок - можливості віртуальних турів Midjourney пропонують динамічний та захоплюючий досвід.

2. Створення контенту на 360°: Завдяки підтримці 360° фото і відео, Midjourny дозволяє користувачам створювати захоплюючий контент, який переносить глядачів у різні місця і середовища. Незалежно від того, чи демонструєте ви мальовничий пейзаж, жваву міську вулицю або внутрішній простір, інструменти Midjourney для створення 360° контенту дають користувачам можливість створювати захоплюючий досвід, який по-справжньому занурює їхню аудиторію.

3. Кастомізація та брендування: Midjourney пропонує широкі можливості кастомізації, дозволяючи користувачам адаптувати свій віртуальний досвід до фірмового стилю та повідомлень свого бренду. Від спеціальних елементів брендингу та інтерактивних накладок до персоналізованих навігаційних меню та вбудованого мультимедійного контенту, Midjourney дозволяє користувачам створювати захоплюючий досвід, який резонує з їхньою аудиторією.

4. Аналітика та інсайти: Midjourney надає користувачам цінну аналітику та інформацію про ефективність їхнього віртуального досвіду. Від показників залучення глядачів і теплових карт до відстеження конверсій і демографічних

даних аудиторії - аналітична панель Midjourney дозволяє користувачам отримувати дієві ідеї та оптимізувати свій віртуальний контент для досягнення максимального ефекту.

5. Міжплатформенна сумісність: Незалежно від того, чи переглядається він на стаціонарному комп'ютері, мобільному пристрої або гарнітурі віртуальної реальності, віртуальний досвід Midjourney оптимізовано для крос-платформної сумісності, що забезпечує безперебійний перегляд на широкому спектрі пристроїв і платформ.

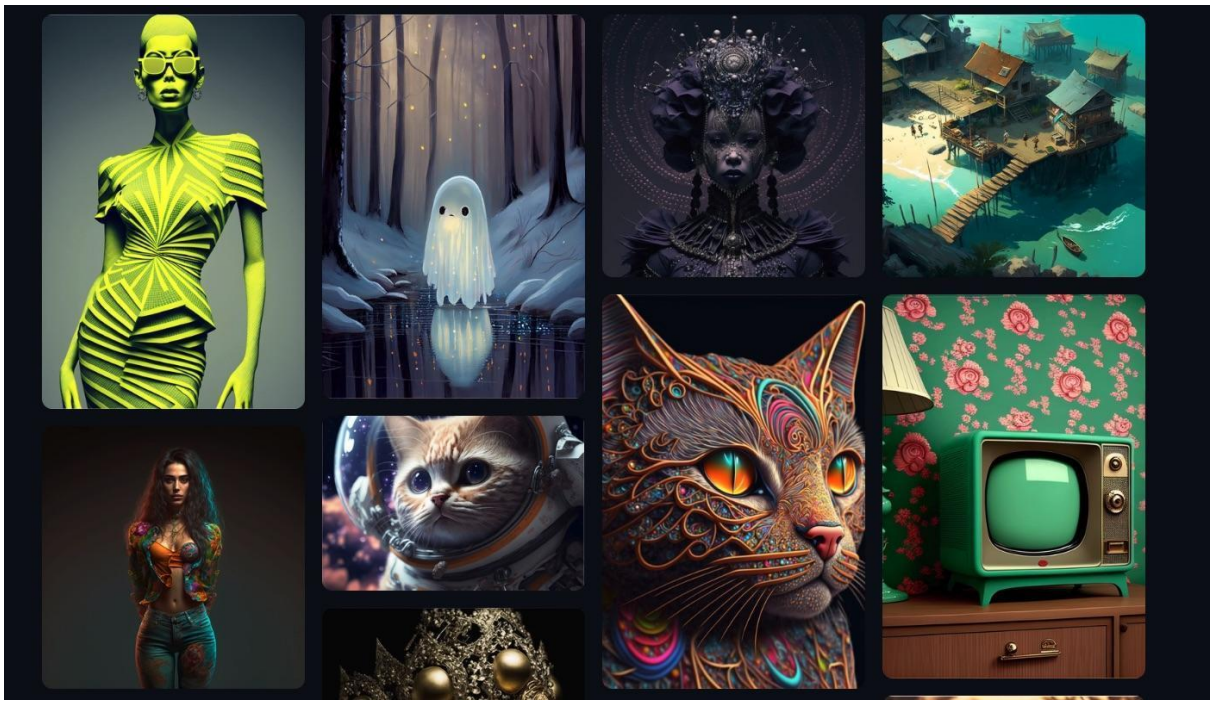

Рисунок 1.1 Результати генерації Midjourney

Застосування Midjourney:

1. Нерухомість: Midjourney широко використовується в індустрії нерухомості для створення віртуальних турів, які дозволяють потенційним покупцям дистанційно оглядати будинки, квартири та комерційні об'єкти. Пропонуючи захоплюючі віртуальні тури, агенти з нерухомості можуть демонструвати

нерухомість потенційним покупцям незалежно від їхнього місцезнаходження, покращуючи досвід купівлі та підвищуючи зацікавленість.

2. Туризм та готельний бізнес: Туристичні бюро, готелі та туристичні агенції використовують Midjourney для створення віртуальних турів популярними напрямками, визначними пам'ятками та об'єктами розміщення. Пропонуючи захоплюючий віртуальний досвід, учасники туристичного ринку можуть надихати мандрівників, демонструвати красу свого місця призначення, а також стимулювати бронювання та відвідування.

3. Освіта та навчання: Міджорні все частіше застосовуються в освітніх установах для створення захоплюючого навчального досвіду. Від віртуальних екскурсій кампусом та інтерактивних музейних експонатів до віртуальних екскурсій та освітніх симуляцій, Midjourney надає викладачам потужну платформу для залучення студентів та покращення результатів навчання.

4. Маркетинг та просування подій: Компанії та організації використовують Midjourny для створення віртуального досвіду в маркетингових та рекламних цілях. Незалежно від того, чи це демонстрація продуктів і послуг на віртуальних виставках і виставках, чи проведення віртуальних заходів і конференцій, Midjourney дозволяє компаніям налагоджувати зв'язок зі своєю аудиторією у змістовний і захоплюючий спосіб.

Технічні аспекти:

1. Зшивання зображень: Для створення панорамних видів і 360° контенту Midjourney використовує алгоритми зшивання зображень. Ці алгоритми об'єднують кілька зображень, що перекриваються, в одну безшовну панораму. Для точного вирівнювання і накладання зображень зазвичай використовуються такі методи, як виявлення об'єктів, реєстрація зображень і змішування, що забезпечує плавний перехід між різними перспективами у віртуальному середовищі.

2. Відображення текстур: У віртуальних турах і середовищах текстури відіграють вирішальну роль у реалістичному рендерингу поверхонь. Midjourney

може використовувати алгоритми відображення текстур для нанесення текстур на 3D-моделі або панорамні зображення. Такі методи, як UV-мапування та фільтрація текстур, використовуються для точного накладання текстур на поверхні, мінімізуючи спотворення та оптимізуючи продуктивність рендерингу.

3. Навігація та взаємодія: Віртуальний досвід Midjourney часто включає елементи навігації та взаємодії, що дозволяє користувачам досліджувати середовище та взаємодіяти з об'єктами. Алгоритми пошуку шляхів можуть бути використані для розрахунку оптимальних навігаційних шляхів у віртуальних просторах, забезпечуючи плавне та ефективне переміщення користувачів. Крім того, інтерактивні точки та об'єкти можуть використовувати алгоритми виявлення зіткнень для виявлення взаємодії користувачів і запуску відповідних реакцій у віртуальному середовищі.

4. Оптимізація рендерингу: Відтворення складних віртуальних середовищ у реальному часі вимагає ефективних методів оптимізації. Midjourney використовує алгоритми рендерингу та оптимізації, такі як рендеринг рівня деталізації (LOD), відсіювання оклюзій та стиснення текстур, щоб забезпечити плавну роботу на різних пристроях та платформах. Ці методи динамічно налаштовують якість і деталізацію рендерингу залежно від точки зору глядача і можливостей пристрою, максимізуючи продуктивність без шкоди для візуальної достовірності.

5. Аналітика та користувацькі інсайти: За лаштунками Midjourney збирає дані про взаємодію та залучення користувачів у віртуальний досвід. Алгоритми машинного навчання можуть використовуватися для аналізу цих даних і надання інформації про поведінку, вподобання та моделі залучення користувачів. Такі методи, як кластеризація, класифікація та прогнозне моделювання, можуть допомогти виявити тенденції, оптимізувати контент і персоналізувати досвід для окремих користувачів або цільових аудиторій.

Bonsai

Bonsai - це платформа, розроблена корпорацією Майкрософт, яка пропонує набір інструментів і сервісів для побудови та розгортання автономних систем,

керованих штучним інтелектом. На відміну від традиційних платформ ШІ, які зосереджені переважно на аналізі даних і моделях машинного навчання, Bonsai спеціалізується на навчанні з підкріпленням (RL) - галузі машинного навчання, яка дозволяє агентам вивчати оптимальну поведінку через взаємодію з навколишнім середовищем методом спроб і помилок.

Bonsai призначений для спрощення розробки та розгортання автономних систем, керованих ШІ, особливо в промислових умовах, де переважають складні завдання управління та прийняття рішень. Використовуючи методи навчання з підкріпленням, Bonsai дозволяє розробникам навчати агентів виконувати завдання і приймати рішення в динамічних і невизначених умовах, що в кінцевому підсумку призводить до створення більш адаптивних та інтелектуальних систем.

Ключові компоненти Bonsai:

1. Інтерфейс машинного навчання: В основі Bonsai лежить інтерфейс машинного навчання, який дозволяє розробникам визначати цілі, обмеження та винагороди, пов'язані з конкретним завданням. Замість того, щоб вручну кодувати поведінку або надавати марковані навчальні дані, розробники інтерактивно навчають агента, як виконувати завдання, за допомогою демонстрацій, симуляцій та циклів зворотного зв'язку.

2. Движок навчання з підкріпленням: Механізм навчання з підкріпленням Bonsai слугує основним механізмом навчання агентів. Він використовує передові алгоритми RL для оптимізації поведінки агента на основі зворотного зв'язку, що надається інтерфейсом машинного навчання. У процесі дослідження та експлуатації агент вчиться максимізувати сукупну винагороду, орієнтуючись у своєму середовищі та виконуючи завдання.

3. Імітаційне середовище: Bonsai надає симуляційне середовище, де розробники можуть тренувати та оцінювати агентів у віртуальних середовищах, що імітують реальні сценарії. Це дозволяє швидко створювати прототипи і тестувати системи, керовані ШІ, без потреби у фізичному обладнанні або інфраструктурі.

Середовище симуляції може моделювати складну динаміку, фізику і взаємодію, дозволяючи агентам навчатися надійній і адаптивній поведінці.

4. Інтеграція з існуючими системами: Bonsai розроблений для безперешкодної інтеграції з існуючими системами та робочими процесами, що полегшує розгортання рішень на основі штучного інтелекту у виробничих середовищах. Незалежно від того, чи це управління роботизованими маніпуляторами на заводі, оптимізація використання енергії в інтелектуальній мережі або управління логістикою в ланцюжку поставок, Bonsai можна інтегрувати з широким спектром апаратних і програмних систем, щоб покращити процес прийняття рішень людиною і автоматизувати складні задачі.

Застосування Bonsai:

1. Промислова автоматизація: Bonsai широко використовується в промислових умовах для автоматизації складних завдань управління та прийняття рішень. Від оптимізації виробничих процесів і планування робочих процесів до управління роботизованими системами і автономними транспортними засобами, Bonsai дозволяє компаніям підвищити ефективність, безпеку і продуктивність своїх операцій.

2. Автономні системи: Bonsai також застосовується для розробки автономних систем у різних галузях, включаючи транспорт, сільське господарство та охорону здоров'я. Незалежно від того, чи це автономні дрони, що інспектують інфраструктуру, самокеровані транспортні засоби, що орієнтуються в міському середовищі, або інтелектуальні агенти, що допомагають медичним працівникам у діагностиці та лікуванні, Bonsai надає інструменти та можливості, необхідні для побудови адаптивних та інтелектуальних автономних систем.

3. Оптимізація процесів: Bonsai можна використовувати для оптимізації складних процесів і робочих циклів, навчаючись на основі даних і зворотного зв'язку в режимі реального часу. Незалежно від того, чи це оптимізація логістики ланцюга поставок, планування заходів з технічного обслуговування або управління

енергоспоживанням, Bonsai дозволяє організаціям приймати рішення на основі даних і більш ефективно адаптуватися до мінливих умов.

Bonsai дозволяє створювати моделі штучного інтелекту, які можна вбудовувати в додатки, що полегшує інтеграцію інтелектуальної поведінки в ігри. Він спрощує складні аспекти машинного навчання до інтуїтивно зрозумілого інтерфейсу, дозволяючи розробникам, навіть тим, хто не дуже добре розбирається в AI, використовувати передові можливості штучного інтелекту у своїх іграх.

Promethean AI

Promethean AI - це інноваційна платформа, яка використовує штучний інтелект (ШІ) для спрощення та покращення процесу створення цифрового контенту, зокрема в галузі архітектурного дизайну та візуалізації. Розроблена Promethean AI, платформа використовує алгоритми штучного інтелекту, щоб допомогти архітекторам, дизайнерам і художникам більш ефективно створювати 3D-середовища та ресурси. Пропонуємо ознайомитися з Promethean AI ближче:

1. Проектування зі штучним інтелектом: Promethean AI використовує алгоритми штучного інтелекту, щоб допомогти користувачам у створенні 3D-середовищ, будівель та інших архітектурних елементів. Аналізуючи вхідні дані та вподобання користувача, платформа може автоматично генерувати детальні 3D-моделі, текстури та макети, значно скорочуючи час і зусилля, необхідні для ручного проектування.

2. Обробка природної мови: Однією з ключових особливостей Promethean AI є його можливості обробки природної мови (NLP), які дозволяють користувачам взаємодіяти з платформою, використовуючи команди та описи звичайною мовою. Користувачі можуть описувати свої дизайнерські ідеї та вподобання природною мовою, а система ШІ інтерпретує ці дані для створення відповідного 3D-контенту та оточення.

3. Генеративні алгоритми: ШІ Promethean використовує генеративні алгоритми для створення різноманітного та реалістичного 3D-контенту. Ці

алгоритми можуть генерувати широкий спектр архітектурних стилів, від сучасних хмарочосів до історичних пам'яток, на основі вхідних параметрів, таких як місце розташування, масштаб і архітектурні обмеження. Платформа постійно навчається і вдосконалює свої можливості генерації з часом, дозволяючи користувачам отримати доступ до постійно зростаючої бібліотеки варіантів дизайну.

4. Інтеграція з інструментами проектування: Promethean AI легко інтегрується з популярним програмним забезпеченням і платформами для проектування, такими як Revit і Unreal Engine від Autodesk. Ця інтеграція дозволяє користувачам включати 3D-контент, створений ШІ, безпосередньо в існуючі робочі процеси, покращуючи процес проектування без необхідності тривалого перенавчання або адаптації.

5. Спільне проектування: ШІ Promethean полегшує спільне проектування, дозволяючи декільком користувачам одночасно працювати над одним проектом. Користувачі можуть ділитися дизайнерськими ідеями, повторювати концепції та надавати зворотній зв'язок у режимі реального часу, сприяючи співпраці та творчості серед членів команди.

6. Візуалізація та рендеринг: Після завершення початкового проектування Promethean AI пропонує інструменти візуалізації та рендерингу для створення фотореалістичних зображень та анімації запропонованих архітектурних проектів. Користувачі можуть досліджувати свої проекти з різних кутів і перспектив, допомагаючи ефективно донести своє бачення до клієнтів, зацікавлених сторін і проектних команд.

Загалом, Promethean AI є значним досягненням у сфері архітектурного дизайну та візуалізації, використовуючи технологію штучного інтелекту, щоб дати можливість дизайнерам та архітекторам створювати захоплюючі та привабливі 3D-середовища більш ефективно та спільно, ніж будь-коли раніше.

1.3 Можливості штучного інтелекту від Unity Muse

Unity Muse, що працює на базі штучного інтелекту нового покоління, дозволяє творцям усіх рівнів кваліфікації блискавично розробляти ігри та 3D-додатки в реальному часі за допомогою простих текстових підказок.

Можливості Muse:

- Текстури - Використовуйте просте, природне введення - підказку, зображення, малюнок ,щоб миттєво генерувати готові до гри текстури у будь-якому стилі, і все це всередині вашого проекту в редакторі Unity.
- Спрайти - створюйте 2D-арт миттєво, безпосередньо у редакторі Unity. За допомогою кількох простих підказок ви отримуєте добірку готових спрайтів, які можна змінювати, за лічені секунди.
- Чат - Розв'язує складні завдання, знаходить ідеї для проекту та допомогає створювати придатний для використання код за допомогою діалогового чату зі штучним інтелектом, який спрямовує вас у процесі творення.
- Анімації - оживляє людиноподібних персонажів за допомогою кількох текстових підказок, заощаджуючи години традиційних налаштувань - розроблено як для початківців, так і для досвідчених аніматорів.

Як Unity Muse генерує свої результати?

Unity Muse базується на генеративному штучному інтелекті, де кожна функція забезпечується різними типами моделей штучного інтелекту. Для створення Muse Chat ліцензували сторонні великі мовні моделі (LLM) та інтегрували їх з технічною документацією Unity від сторонніх розробників (включаючи посібники, примітки до релізу, підручники, посібники зі створення сценаріїв і так далі).

Які великі мовні моделі (LLM) ви використовуються?

Unity використовував Azure AI та OpenAI при розробці чату та функцій LLM в поведінці, експериментуючи з золотим стандартом в індустрії, оскільки

створюють власні гнучкі та складні системи, щоб мати можливість перемикатися на різні LLM, щоб надати найкращі відповіді на кожне унікальне питання.

Unity Sentis

Unity Sentis працює на кросплатформенному ігровому рушію Unity і легко включає моделі ШІ в Unity Runtime, значно покращуючи гру або додаток на пристроях користувачів (Рис. 1.2).

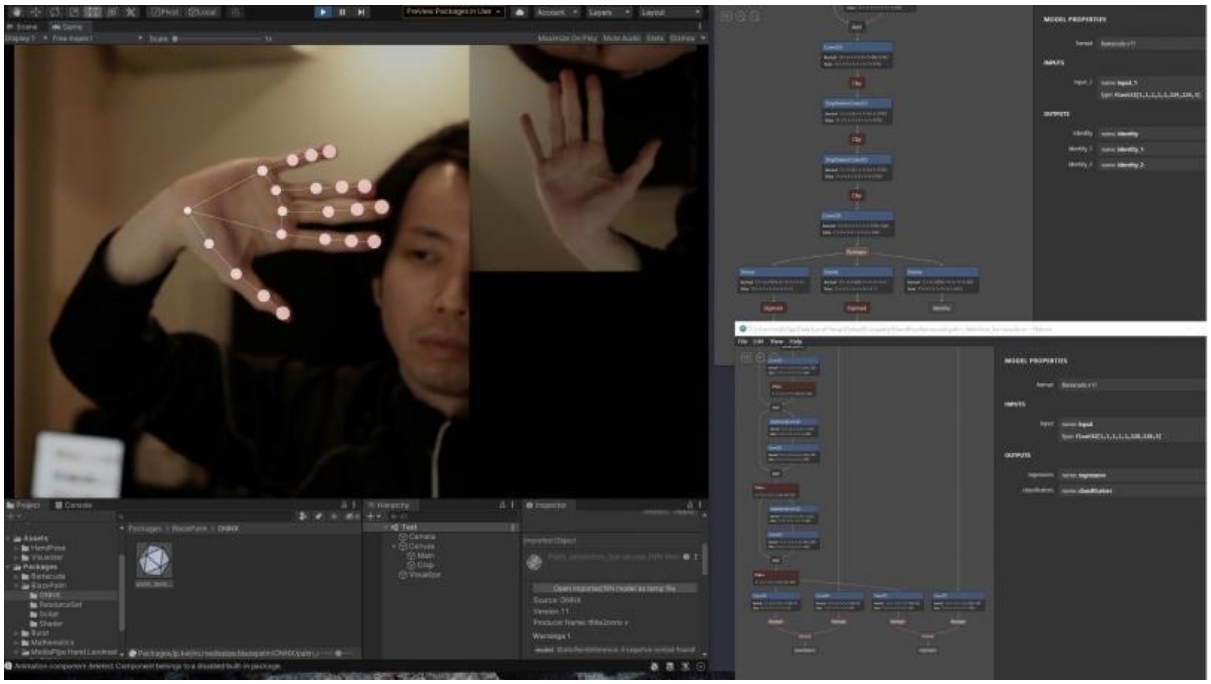

Рисунок 1.2 кадри застосування Unity Sentis

Гнучка реалізація моделі ШІ

Sentis дозволяє реалізовувати стандартизовані моделі ШІ, такі як передача стилю та розпізнавання мови. Моделі можна налаштовувати за допомогою вагових коефіцієнтів, шарів і ланцюжків висновків.

Висока продуктивність на пристроях користувачів

Sentis використовує обчислювальну потужність пристроїв кінцевих користувачів, а не хмари, що усуває складну хмарну інфраструктуру, мережеві затримки та постійні витрати на виведення (Рис. 1.3).

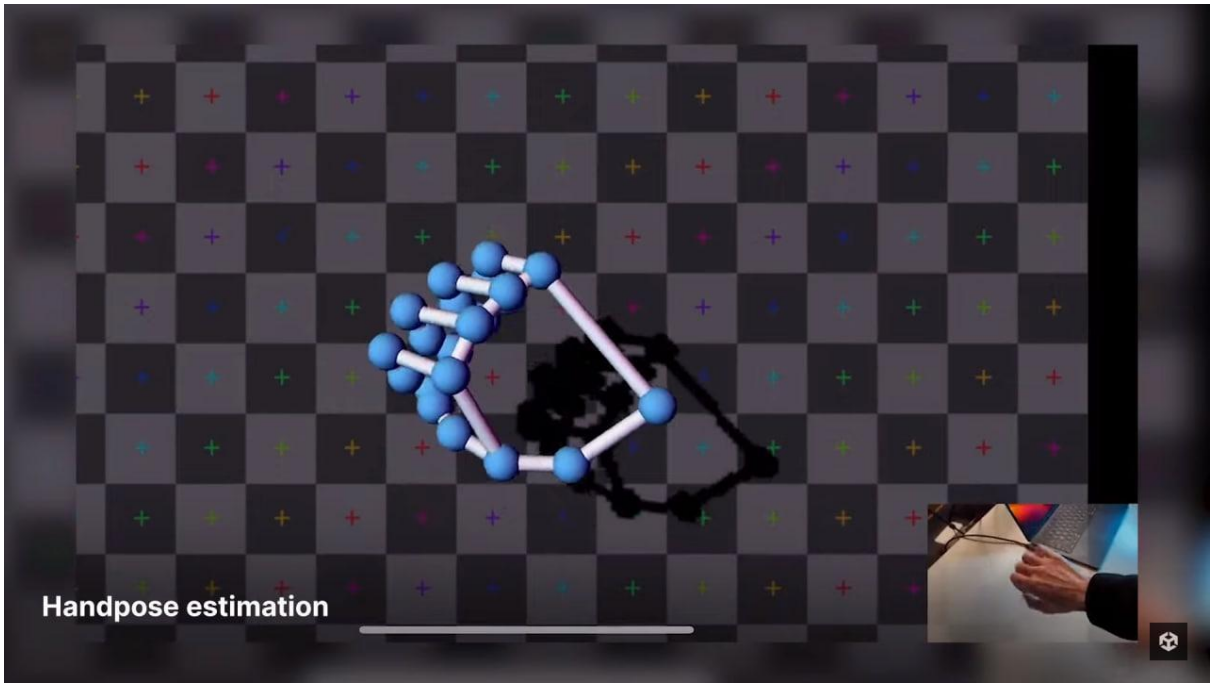

Рисунок 1.3 Захват рухів Unity Sentis

Технічні особливості:

- Доступний - Інтуїтивно зрозумілий API "plug-and-play" робить інтеграцію ШІ-моделей доступною за допомогою стандарту файлів моделей ONNX.
- Багатоплатформенна підтримка - Sentis працює з усіма платформами, що підтримують Unity, від мобільних до ПК та популярних ігрових консолей.
- Оптимізація - оптимізуйте модель ШІ для середовища виконання, а потім викликайте висновки зі швидкістю гри для плавного ігрового процесу.
- Оптимізація апаратного забезпечення - розгортайте висновки на найкращому доступному обладнанні (CPU або GPU) і використовуйте наявні обчислювальні системи та завдання Unity.
- Безпека - ШІ-моделі запускаються локально за допомогою Unity Sentis, дані не зберігаються і не передаються в хмару.

Варіанти використання:

Sentis - це фреймворк для імпорту та запуску сторонніх моделей штучного інтелекту, але він не містить власних моделей. Його можливості:

- Ідентифікація об'єктів - виявляйте, класифікуйте та сегментуйте об'єкти за допомогою ігрової камери або камери на пристрої.
- Налаштовувані ШІ-супротивники - створіть суперника в настільній грі з особливими правилами та власними кривими складності. Запустіть нейронну мережу, навчену на правилах гри, і визначайте ймовірність перемоги після кожного ходу.
- Розпізнавання рукописного тексту - розпізнавайте написані від руки цифри, літери та символи для унікальних ігрових взаємодій.
- Оцінка глибини - оцініть глибину об'єктів реального світу у вікні доповненої реальності, щоб закрити об'єкти в ігровій сцені.
- Розпізнавання мовлення - перетворюйте живу мову на ігровий текст за допомогою мовної моделі машинного навчання для природної мовної взаємодії між гравцями.
- Розумні NPC - Автоматизуйте діалоги та створюйте змістовну взаємодію між гравцями та NPC без обмежень ручного написання сценаріїв.

2 ДОСЛІДЖЕННЯ ОБЧИСЛЕНЬ ТА ГЕНЕРАЦІЇ 3Д МОДЕЛІ

2.1 Ознайомлення з підходом створення

Протягом століть люди виробили певне стійке уявлення про навколишній світ. Згідно з цим сприйняттям, легко описати складні структури за допомогою короткого пояснення. Наприклад, якщо попросити людину «уявити собі жовтий підводний човен, що летить у небі», людина матиме точний образ машини, про яку ви говорите. машину, про яку ви говорите, навіть не бачивши її раніше. Поряд з цим - штучний інтелект (ШІ) є швидко зростаючою сферою досліджень, і цілком можливо, що ШІ дозволить комп'ютерам сприйняття світу. Така здатність дозволила б комп'ютерам виконувати складні завдання комп'ютерам виконувати складні завдання, без того, щоб хтось керував кожною його дією. Таким чином, це значно полегшило б роботу дизайнерів, - графічні дизайнери та архітектори зможуть лише вказувати особливості, замість того, щоб витрачати багато часу на власне малювання.

Машинне навчання (ML) - це розділ штучного інтелекту, який використовується для того, щоб змусити системи автоматично вчитися на власному досвіді. Замість того, щоб програмувати їх на те, як саме виконувати певну роботу. Таким чином, його можна використовуватися для вирішення проблем без необхідності вказувати, як розв'язувати поставлену задачу. Створення 3D-моделей - це складна і трудомістка робота для всіх, хто не має достатнього досвіду в 3D-моделюванні. Це призводить до великою проблемою для ігрових рушіїв, оскільки розробники або повинні вивчати 3D-моделювання, наймати спеціального 3D-дизайнера або купувати сторонні моделі. Тому, якби моделі генерувалися обчислювальним шляхом, процес розробки став би набагато простішим і міг би призвести до більшої швидкісті роботи розробників. Для того, щоб зробити комп'ютерне 3D-моделювання обчислювальне 3D-моделювання придатним для використання, існують три фундаментальні вимоги:

1. Задовільна якість.

2. Швидкий час генерації.

3. Простота у використанні.

Мета штучного інтелекту полягає в тому, щоб зробити взаємодію та спілкування між людиною та системами штучного інтелекту більш природними. Як відомо, зір людського мозку, здається, дуже легко функціонує дуже просто. Нам не складно відрізнити собаку від кішки, прочитати відрізнити собаку від кота, прочитати слово або розпізнати людське обличчя. Але на відміну від людей - ці завдання є дійсно складними завданнями для вирішення за допомогою комп'ютера. Процес розпізнавання тільки здається легким, тому що людський мозок дійсно добре сприймає і, як наслідок, розуміє зображення. Протягом останніх років машинне навчання досягло значного прогресу у вирішенні цих складних завдань. Зокрема, модель під назвою - глибока згорткова нейронна мережа може досягти достатньої продуктивності у вирішенні складних завдань візуального розпізнавання, які збігаються або перевищують людську продуктивність у деяких аспектах.

У нечисленних публікаціях стверджується, що можна згенерувати 3D-модель, використовуючи єдине вхідне зображення та згорткових нейронних мереж (CNN) (Рис. 2.1). Можна припустити, що його можна вважати достатньо простим для третьої вимоги до споживача вимогу. Припущення, яке потребує перевірки є обмеження роздільної здатності на виході та часу генерації, що не було жодного разу не було вказано в жодній зі згаданих вище

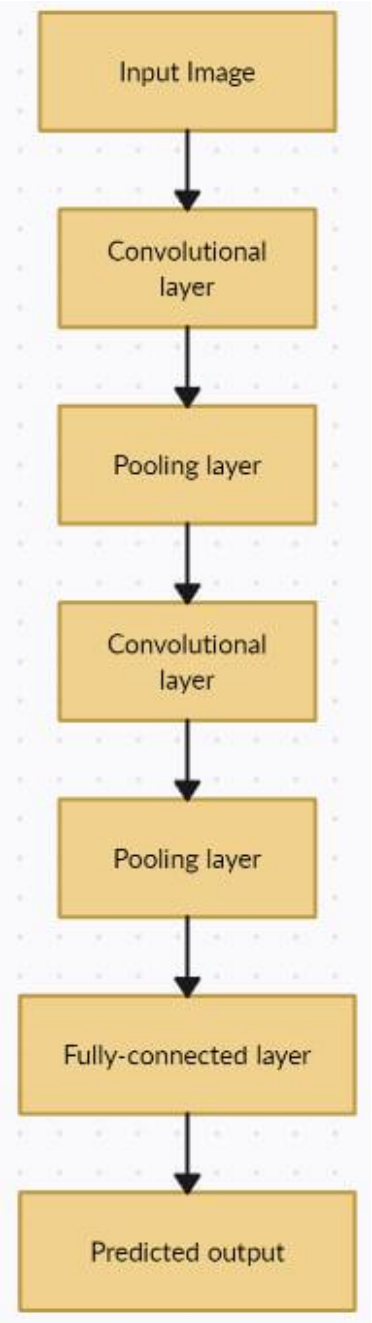

Рисунок 2.1 Візуальне представлення алгоритму згорткової нейронної мережі

Основною метою цього дослідження є визначення можливих обмежень використання CNN для генерації 3D моделей для генерації 3D моделей, беручи до уваги вихідну роздільну здатність та швидкості генерації. Крім того, ця робота

також фокусується на балансуванні між збільшенням якістю роздільної здатності та витратами часу на генерації вихідних даних. Згідно з усім

вищезазначеним питання дослідження може бути сформульоване наступним чином: «Як можна вплинути на швидкість генерації залежно від вихідної роздільної здатності під час генерації 3D-моделей з 2D-зображень за допомогою згорткових нейронних мереж?».

Як було зазначено вище - цей проект рішення фокусується на трьох напрямках (якість, швидкість та простота), щоб зробити програму генерації 3D-зображень задовільною для широкого використання.Тому, в свою чергу, щоб отримати відповідь на дослідницьке питання цієї роботи, необхідно вирішити наступні завдання необхідно вирішити наступні завдання:

1. Розробити модель машинного навчання для вилучення даних з 2D зображення.

2. Розробити генеративну 3D-модель машинного навчання

3. Розробити та зшити обидві моделі машинного навчання.

4. Тестування та налаштування параметрів для досягнення найкращих результатів якості вихідних даних, а також найкращих результатів за часом витрат часу.

5. Досягнення висновку у вигляді фінальної 3D-моделі.

Коли проект цієї роботи буде завершено, результат повинен давати можливість виконувати 2D в 3D перетворення зображення в модель. Зображення має бути отримане з мобільної камери мобільного телефону і в подальшому перетворене на 3D-модель.

Метою цього розділу є з'ясування обмеження використання CNN для генерації 3D вокселізованих моделей, беручи до уваги продуктивності алгоритму. До характеристик продуктивності, що знижуються, можна віднести найбільш цінними вважаються наступні: швидкість генерації та роздільна здатність на виході. Крім того, в цьому розділі також буде приділено увагу можливий масштаб компромісу між покращенням роздільної здатності та збільшенням часу генерації вихідних даних. Результати цієї дослідницької роботи зосереджені на трьох підсферах, а саме: якість, швидкість та простота. Ці підсфери є найбільш цінними

для того, щоб зробити інструмент генерації 3D-зображень зручним для користувача. Коли цей проект буде завершено, мета полягає в тому, щоб уможливити 2D-3D перетворення об'єкта з малюнка, намальованого від руки, на 3D-об'єкт. Об'єкт можна буде імпортувати в інструмент 3d редактора наприклад, Blender та ігровий движок, наприклад, Unity 3D та Unreal Engine.

2.2 Методологія

Основний підхід у цьому розділі полягає у використання емпіричного методу розробки. Він полягає в тому, що деякі великі проблеми повинні бути розділені на менші завдання, використовуючи редукціонізм. Таким чином, генеративна модель спочатку розбивається на дві окремі нейронні мережі. Перша призначена для вилучення дані з вхідного зображення, а друга спрямована на використання результатів роботи першої нейронної мережі для створення 3D-моделі (Рис. 2.2). Обидві ці нейронні мережі були навчені та оптимізували окремо, використовуючи ітеративний, емпіричний дослідницьким методом. Невеликі параметри або набір даних вносяться зміни в нейронну мережу, яка підлягає перенавчати, щоб побачити, чи зменшує нейронна мережа середньоквадратичну похибку (MSE). Наукова назва нейронної мережі процес навчання нейронної мережі називається машинним навчанням. Це процес, який використовує статистику та ймовірність процес, який використовує статистику та ймовірність на наборі даних для класифікації. Можна порівняти ML з традиційним програмуванням і визначити, що звичайне програмування використовує дані разом з програмою для отримання певного результату. Але якщо поміняти місцями вихідні дані та програму, якщо поміняти місцями вихідні дані та програму. Це стане моделлю машинного навчання, яка виробляє програму замість того ж результату, що і як при звичайному програмуванні. Вхідні дані - це «дані», а «вихід» - це програма, яка може згенерувати подібний результат з іншого набору даних.

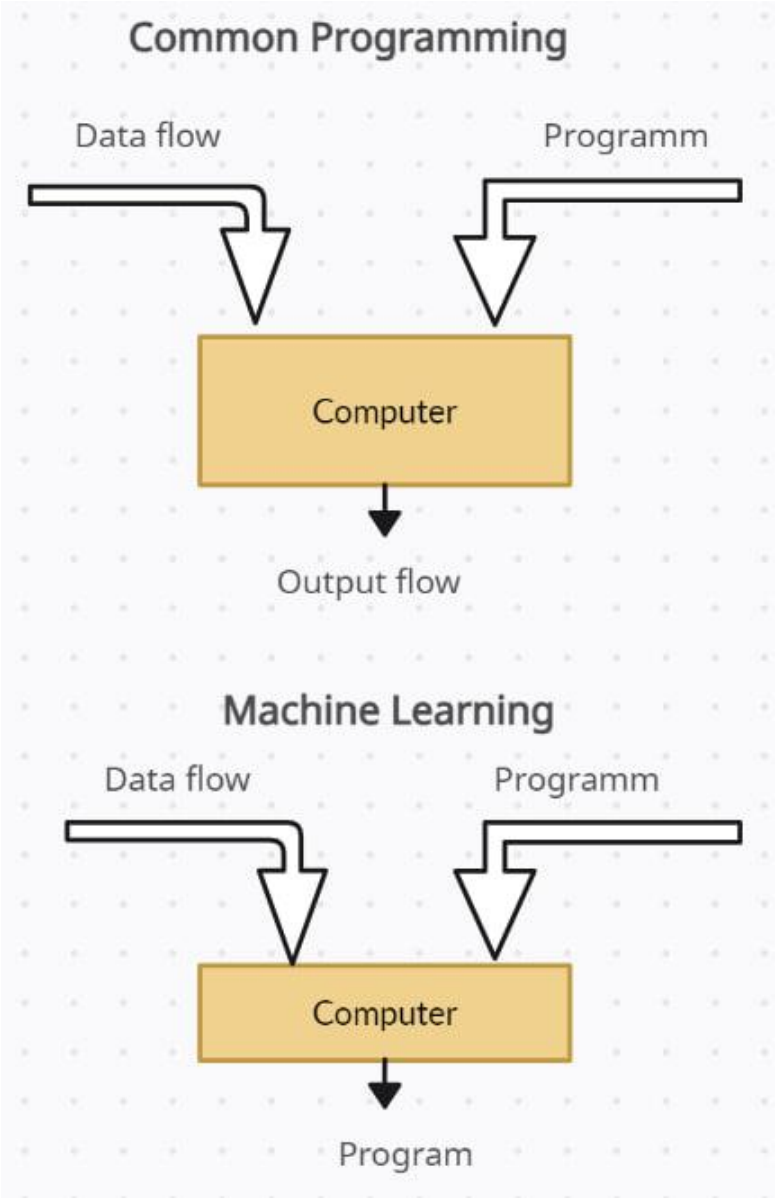

Рисунок 2.2 Приклад загального програмування та принципів побудови машинного навчання

Існує кілька різних варіантів підходів при роботі з машинним навчанням. Це керований підхід (широко відомий як «навчання з учителем»), неконтрольований підхід, напівконтрольований підхід та

підхід з підкріпленням. Підхід навчання під наглядом підхід до навчання може бути використаний і зазвичай використовується коли і вхідні, і вихідні дані доступні для розробнику. Його мета полягає в тому, щоб знайти відповідність між

вхідними та вихідними даними, це відображення може бути надалі використовувати для невидимих вхідних даних, щоб зіставити їх з аналогічним виходом. Прикладом може бути класифікатор, який може класифікувати відправлену веб-форму як «відправлену ботом або «подана людиною». Якщо розробник має набір даних, що складається з класифікованих дій надсилання веб-форм дій, то набір даних може використовувати поведінку у веб-формі, веб-форми як вхідні дані, а класифікацію - як вихідні. Це може створити «програму», яка може відрізнити різницю між ботом і людиною.

Підхід неконтрольованого навчання широко використовується для кластеризації даних. Прикладом може слугувати процес визначення того, як була побудована будь-яка мова побудована разом з дієсловами, іменниками або прикметниками. Підхід до навчання без нагляду також може бути використаний шляхом надання моделі великої кількості текстових даних, і вказати, що модель повинна групувати речення, які використовуються у схожий речення, які використовуються подібним чином. Якщо деякі вхідні дані було задано у попередньому параграфі, а також деякі вихідні дані, а також деякі вихідні дані, але не всі вихідні дані, можна використати дещо складніший підхід, який називається Генеративна змагальна мережа (Generative Adversarial Network) (Рис. 2.3). Це підхід, коли дві нейронні мережі працюють одна проти одної. Наприклад, ви створюєте програму, яка може діяти як бот у веб-формі і два блоки нейронної мережі, які конкуруютьодин проти одного. Один намагається діяти як бот, а інший намагається визначити, чи відправлена веб-форма ботом з іншої нейромережі чи це справжня веб-форма з правильного набору даних. Поки цей процес тренується, обидві мережі вдосконалюватимуться, що призведе до «кращої» ботоподібної поведінки та краще виявлення ботів. Цю модель також можна вважати напівконтрольованою, оскільки деякі дані були існували з самого початку, але нові дані також були створені в процесі навчання. Також можуть бути випадки, коли ні вхідні, ні вихідні дані не були задані, але дані про можливі дії та поточний стан середовища є.

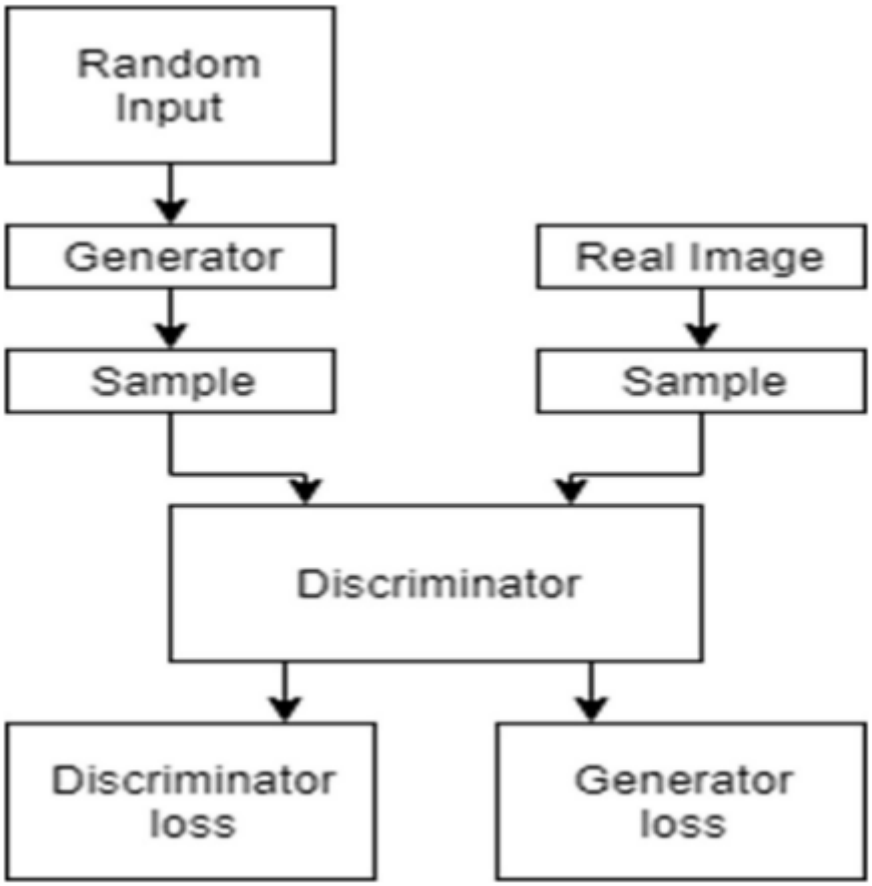

Рисунок 2.3 Візуальне представлення генеративної змагальної мережі

У такому випадку можна застосувати підхід навчання з підкріпленням можна застосувати підхід навчання з підкріпленням. Він починається з того, що випадкового вибору різних можливих дій і припущення про те, чи була ця дія правильною, в залежності від того, як змінилося середовище після того, як дія була виконана. Наприклад, процес польоту гелікоптера можна описати процес, коли гелікоптер намагається злетіти якомога високо, він швидко дізнається, що лопать ротора на хвості, що лопать гвинта на хвості не дає стільки висоти, як якщо б оберталася велика лопать пропелера на верхівці вертольота, обертається велика лопать пропелера у верхній частині гелікоптера.

Найкращим способом використання машинного навчання є використання нейронних мереж, які є біоподібною парадигма програмування. Тобто, це техніка,

яка дає комп'ютерам техніку, яка дає комп'ютерам можливість вчитися на основі спостережуваних даних.

У цьому розділі розглядаються два методи створення 3D-моделі за зображенням. Перший з них - це Генеративна змагальна мережа (Generative Adversarial Network), а другий автокодер на основі вісімкової структурованої архітектури. Набори даних, використані тут для навчальних цілей, були взяті з IKEA, який у свою чергу, був проаналізований вручну на предмет маркування та видалення пошкоджених даних для подальшого та видалення пошкоджених даних для подальшого використання.

Згорткова нейронна мережа (CNN) - це різновид мережа, натхненна зоровою корою головного мозку, успадкована від багатошарових персептронів (MLPs). CNN - це своєрідний механізм, який можна використовувати для представлення даних з невеликою кількістю параметрів, наприклад класифікацією. Щоб пояснити переваги CNN, існує публічний ресурс під назвою Large Scale Visual Recognition Challenge (LSVRC). Починаючи з середини 2013 року цей сервіс проводить щорічні змагання з комп'ютерних наук. Під час цих люди з усього світу намагаються розробити штучний інтелект, здатний штучний інтелект, здатний правильно класифікувати більшість зображень у п'ятнадцяти наборах даних під назвою ImageNet. На сьогоднішній день нейронні мережі, запропоновані в конкурсі, здатні досягти рівня помилок у 3,1%, тоді як ще кілька років тому цей показник становив показником у 20%. Такі результати дійсно показують ефективність нейронних мереж у виробничому використанні. Таким чином, CNN є новим надійним стандартом класифікації та розпізнавання зображень. CNN працює наступним чином обходу фільтрів через вхідні дані та виконуючи обчислення точкового добутку між фільтром і даними, результат цього завдання генерує нову так звану карту активації. Карта активації може бути оброблена, щоб стати вихідним продуктом або вона може стати частиною наступного прихованого шару. У свою

чергу, прихований шар буде оброблятися таким же або подібним чином, як і попередній.

Так чи інакше, згорткові нейронні мережі (CNN) перевершують будь-яку можливу людську швидкість розпізнавання. Однак, такі системи повинні вдосконалювати вручну. Інша проблема з такими системами полягає в тому, що вони потребують точних даних для навчання перед тим, як їх фактично розгортання. Важливо, щоб система була достатньо швидкою для розпізнавання зображень і щоб навчання повинно відбуватися без особливих труднощів, а також бути швидким.

Згортковий шар завжди складається з трьох частин: власне згортки, функції активації та об'єднання. Множення точкового добутку між фільтром/вагами та вхідним шаром або прихованим шаром. Функція активації робить нейронну мережу нелінійною, щоб вона могла знаходити більш складні неочевидні закономірності. Об'єднання має на меті зменшити кількість даних, щоб видалити зайві створені нейрони і, таким чином, підвищити продуктивність.

Мета функції активації полягає в тому, щоб зробити нейронну мережу нелінійною. У свою чергу, це дає можливість визначати більш складні структури в межах наборів даних. Об'єднання має на меті зменшити фактичний розмір кожної карти активації, підвищити швидкість роботи нейронної мережі. Недоліком об'єднання є те, що воно позбавляє нейронну мережу можливості мережі визначати розташування моделей на зображенні, оскільки лише «найкраще» значення зображенні, оскільки зберігається лише «найкраще» значення. Дві широко використовувані функції об'єднання - це max-pooling та середнє об'єднання. Зазвичай вікно розміром 2x2 ковзає по матриці набору даних і якщо використовується максимальне об'єднання, зберігається лише найбільше значення у вікні. Середнє об'єднання збереже середнє значення всіх значень у вікні. Маючи масив даних розміром 2x2 розмір вікна, розмір карти активації можна зменшити на 75%. Алгоритм оптимізації спрямований на мінімізацію або максимізувати функцію втрат. Вона обробляється шляхом обчислення градієнта, який

використовується, щоб вирішити, чи потрібно ваги та зсуви мають бути збільшені або зменшити. Загалом, існує 3 підходи до оптимізації, які будуть розглянуті в цій роботі:

- Градієнтний спуск
- Адам
- Адаград

Вони були обрані через те, що вони, як було доведено, дають задовільний результат в інших подібних проектах, і вони є найпопулярніші підходи до оптимізації. Градієнтний спуск був обраний тому, що він розглядається як основа оптимізаційних підходів у цій галузі (Рис. 2.4). Адаптивна оцінка моменту, відома як Адам, є популярним підходом, який обчислює швидкість навчання для кожної ваги/зсуву. Він добре працює на практиці завдяки своїй здатності швидко збігатися і хорошій швидкості навчання. Адаград - це адаптивний градієнтний підхід, кожна ітерація якого зменшує швидкість самонавчання, що в теорії змусить його робити меншу кількість модифікацій

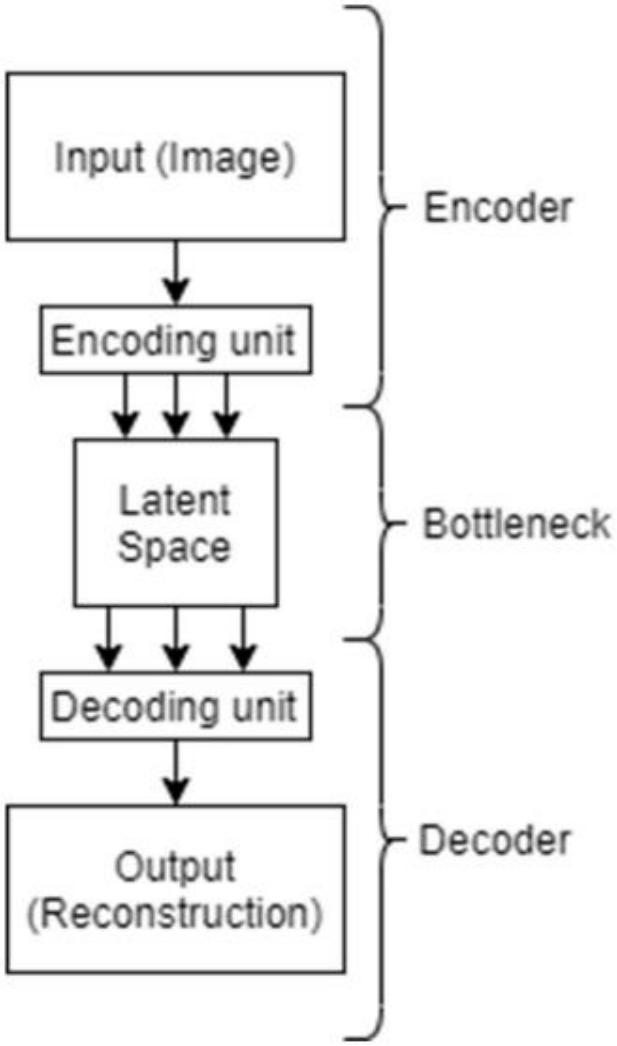

Рисунок 2.4 Візуалізація автокодера, Структура навчання

Автокодер - одне з найпопулярніших застосувань CNN. В основному, він зазвичай відомий своєю здатністю навчатися за допомогою легкого для розуміння, неконтрольованого методом навчання. Він використовує згорткові мережі для зменшення значення даних даного об'єкта в латентний простір. Далі він буде використовувати цей латентний простір для перетворення вихідного об'єкта. Це доводить, що автокодер здатен витягувати дані з об'єкта з метою його перетворення.

Автокодеру потрібно зменшити різницю між оригіналом та перетвореним об'єктом, з зворотним розповсюдженням. З яких основних структур складається

автокодер - кодер і декодер. Призначення кодера - зменшити кількість даних, необхідних для опису об'єкта, тоді як декодер намагатиметься перетворити об'єкт (Рис. 2.5).

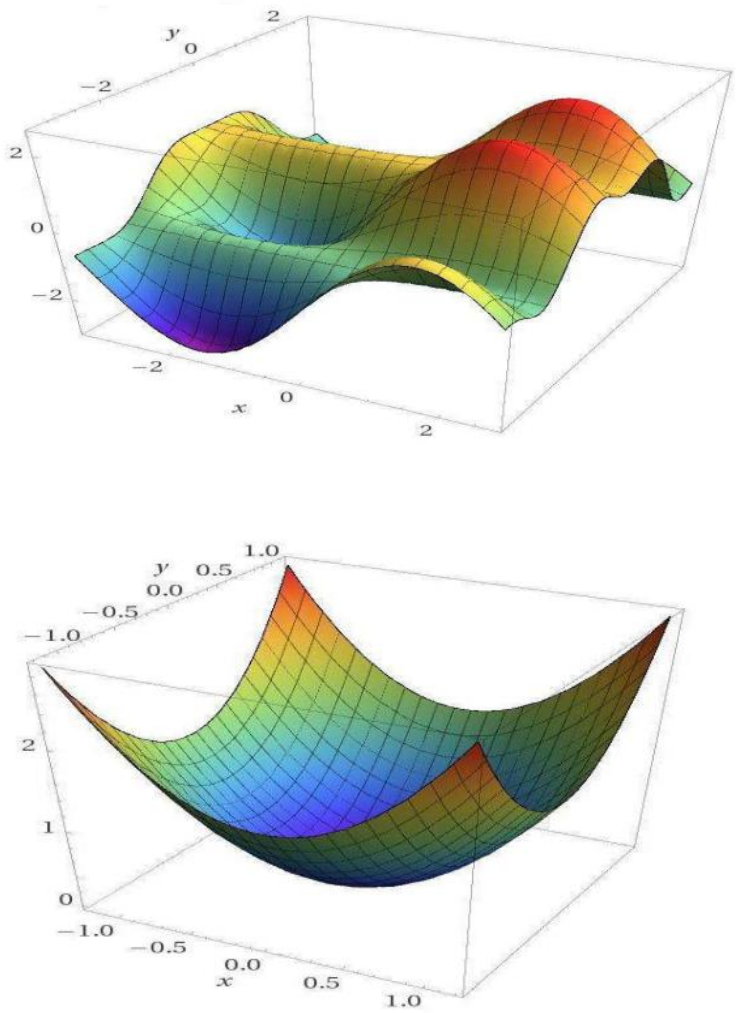

Рисунок 2.5 Візуальне представлення алгоритму градієнтного спуску

Генеративний ШІ зробив величезний прорив у 2014 році, коли представив нову модель нейронної мережі під назвою Generative Adversarial Network (GAN). Основна концепція Generative Adversarial мережі полягає в наступному: замість того, щоб навчати одну нейронну мережу на основі заздалегідь визначених відповідей (навчання з учителем), ця модель складається з двох нейронних мереж, дискримінатора та генератора. Завданням мережі-дискримінатора є правильно класифікувати, чи є вхідні дані частиною набору даних чи ні. Метою мережі-

генератора є обдурити дискримінатор, створивши дані, які він класифікує як «частину набору даних». Тренуючи обидві мережі одна проти одної, дискримінатор стане краще ідентифікувати об'єкти з набору даних, а генератор набору даних, навчиться краще створювати дані з атрибутами, які відповідають реальному набору даних.

Дані - один з найважливіших аспектів роботи зі штучним інтелектом. Щоб зменшити обсяг цієї роботи обсягу цієї роботи, в наборі даних розглядаються лише таблиці, які були представлені в наборі даних IKEA. Набір даних IKEA було обрано через те, що він є безкоштовним у використанні і що його відносно легко модифікувати та змінювати. Для того, щоб переконатися, що жодні пошкоджені дані не впливають на роботу нейронної мережі, було використано кілька наборів даних. нейронну мережу, розглядається декілька наборів даних, використовуючи як існуючі дані, так і генеруючи нові набір даних. Три підходи, описані нижче використовуються лише для модифікації вхідних даних, 3D-вихід є однаковий для всіх наборів даних зображень. Файли 3D-моделей з набору даних IKEA конвертуються у вокселізований python-формат за допомогою 3D-редактора Blender. Він використовує простий графічний інтерфейс, який легко зрозуміти, і графічний процесор локального комп'ютера, що робить перетворення швидкою та безперебійною. Рішення цієї роботи буде використовувати MSE для отримання кількісного порівняння на основі даних між базовими моделями та результатами. Нейронна мережу було покращено за рахунок зменшення MSE у попередній.

2.3 Впровадження

Глибокі згорткові нейронні мережі (CNN) досягли значних покращень у різних завданнях зору, включаючи класифікацію, виявлення та сегментацію. Однак, збільшення розміру моделі та обчислень ускладнює реалізацію DNN на вбудованих системах з обмеженими апаратними ресурсами. З метою відтворюваності, ця частина починається з налаштування середовища, за яким слідує архітектури CNN. Програмне та апаратне забезпечення, що використовується в цій роботі наступні:

1. Для експерименту, показаного у цій роботі, використовувався Python 3.5, але також у майбутніх експериментах буде C++ у випадку кращої можливої продуктивності за рахунок можливої низькорівневої реалізації ядра підходів за рахунок можливої низькорівневої реалізації ядра.

2. Tensorflow

3. Windows 10

4. GPU: NVIDIA GeForce RTX 2080 8GB

5. 32 GB RAM

6. Розмір набору даних: вхідних зображень ~ 4500, вихідних об'єктів ~ 20

Базову архітектуру було створено з двох згорткових нейронних мереж (CNN), перша називається «Convolutional Autoencoder», а друга - «Convolutional 3D Decoder». Робота першої ШНМ полягає в тому, щоб витягувати дані із зображення, які далі передаються як вхідні дані до другого ШНМ. Завдання другого CNN полягає у створенні вісімкової структури вокс-простору з латентного простору. Цей процес повторюється кілька разів поспіль, щоб збільшити роздільну здатність воксельного простору. У зв'язку з тим, що генератор 3D-моделей складно узагальнити, існували завдання вилучення даних, щоб зробити латентний простір схожим до тієї самої моделі, незалежно від кута нахилу моделі на вхідному зображенні.

Основна і єдина відмінність полягає в наборі даних. Для того, щоб зробити його більш поширеним при розпізнаванні моделей з різних ракурсів, всі зображення одного і того ж об'єкта одного і того ж об'єкта трансформуються в один і той самий результат. Таким чином, це також полегшить роботу 3D генеративної нейронної мережі, оскільки кут вхідного сигналу не має значення для 3D-моделі. Генерування 3D-моделей не є тривіальним завданням у здоровому глузді, тому було враховано два різних підходи було взято до уваги два різних підходи. Умовний Генеративна змагальна мережа (Generative Adversarial Network) була першим підходом, оскільки він був широко відомий у сфері генерації контенту і доведено, що він дає задовільний результат. Другий підхід - це умовний автокодер,

який використовує вісімкову структуру. На початку розвитку умовної генеративної змагальної мережі було показано, що вона має експоненціальне зростання використання пам'яті при збільшенні роздільної здатності. Це не є стабільним рішенням, тому був сфокусований підхід з умовним автокодером. Ідея структури умовного автокодера полягала в тому, щоб змусити нейронну мережу генерувати нейронну мережу генерувати модель поетапно. (Рис. 2.6). Фундамент натхненний процесом процесу ліплення, який в даному випадку починається з одного вокселя, який розбивається на менші частини, які перевіряються, щоб видалити непотрібні вокселів/просторів, що, в свою чергу, призводить до покращення роздільної здатності. Кожен крок навчався окремо, що дозволило нейронній мережі тренуватися на створенні якомога вищої роздільної здатності. Перший крок генерував 4x4x4 (21 воксель), наприклад, на цьому кроці можна було знайти пропорції моделі. Для моделі, яка вдвічі більша за ширину потрібно було б використати лише 64 вокселі (одна сторона куба). Щоб зрозуміти, наскільки це ефективно, уявімо собі припустимо, що ви хочете створити об'єкт з якістю 128x128x128 (x, y, z) вокселів. У цьому випадку на першому кроці буде відкинуто половину вокселів (128x128x128/2 = 1 048 576 перестановок вокселів), що дуже багато. Тому кожна ітерація генерації може покращити якість згенерованого стільця. Загальна кількість вокселів, необхідних для представлення таблиці у навчальному наборі даних, що відповідає 128x128x128, становить від ~5000 до ~70000 вокселів. Порівнюючи найгірший випадок (~70 000) з мутаціями для кожного вокселя у зменшить загальну кількість передбачуваних вокселів до ~70000 з 1 048 576 (~ 6.7%), а у найкращому випадку – до ~5000 прогнозів з 1048576 (~0.47%). Справа в тому, що відсоток загального воксельного простору зменшується з кожним LoD, отже, набагато більше виграшу від підвищеної якості. Результатом є те, що відсоток загального воксельного простору зменшується з кожним рівнем деталізації. Рівень деталізації, відсоток від загальної кількості вокселів, використаних у Таблиці для різних рівнів деталізації (Таблиця 2.1).

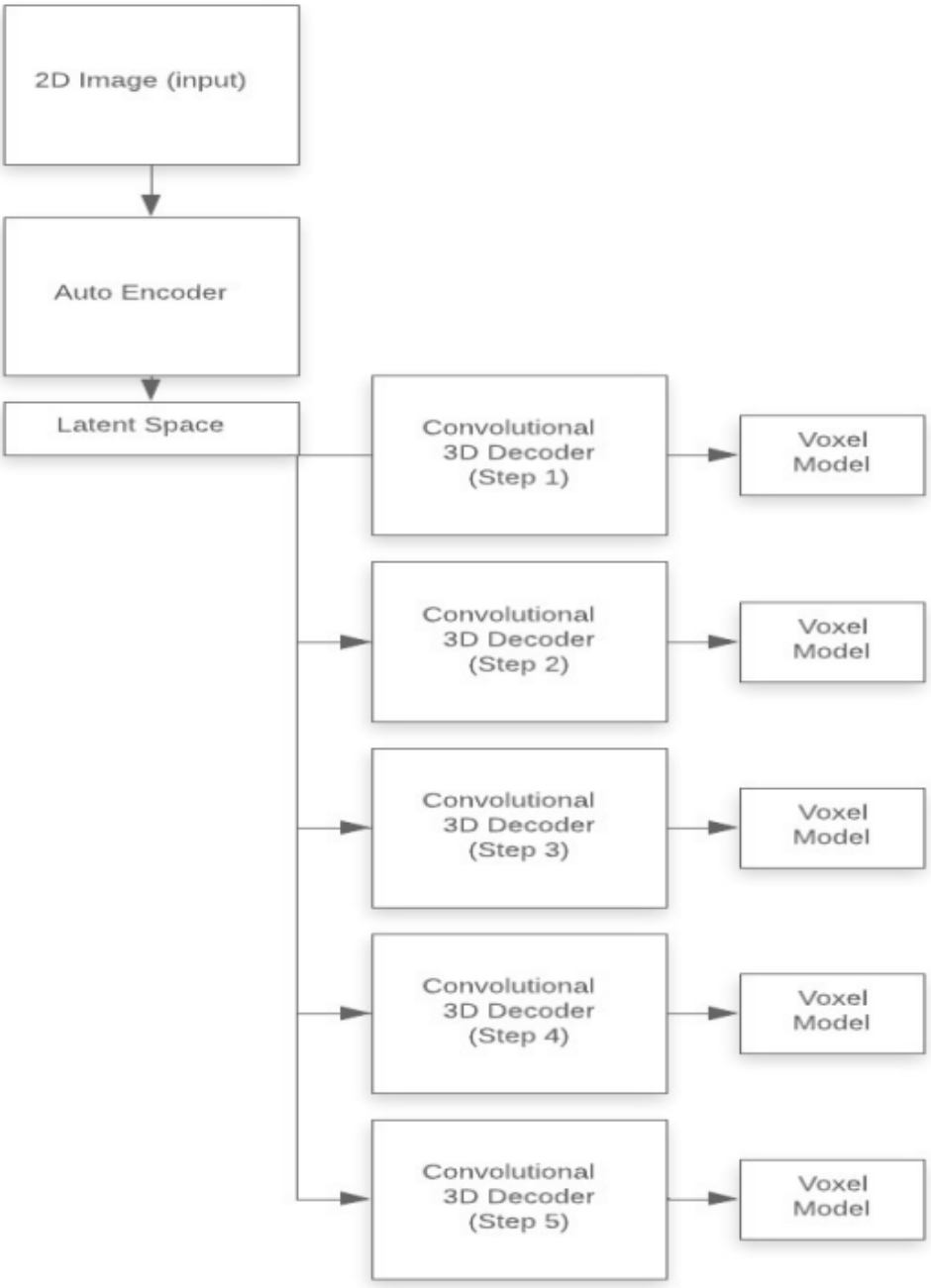

Рисунок 2.6 Архітектурна структура нейронних мереж, що використовуються під час генерації

Level of Detail	Height / Width / Depth	Percentage of used voxels
0	2/2/2	37.5 – 100%
1	4/4/4	21.4 – 81.8%
2	8/8/8	10.2 – 46.2%
3	16/16/16	4.2 – 22.1%
4	32/32/32	2.2 – 12.5%
5	64/64/64	1.1 – 6.5%
6	128/128/128	0.5 – 3.3%
7	256/256/256	0.1 – 1.8%

Таблиця 2.1 Відсоток вокселів, використаних для кожного рівня деталізації

Результати різних ітерацій навчання 3D покоління CNN були зібрані з процесу визначення оптимального відсіву, функції активації та функції втрат. Результати покращувалися зі збільшенням відсіву, завдяки через те, що в процесі вилучення даних була частина узагальнення вилучення даних, що було очікуваним результатом. Результат представлено в таблиці нижче (Таблиця 2.2).

Drop	0.5	0.75	0.9	0.95	0.99	1
Loss	0.25177	0.1588	0.04324	0.0283	0.0281	0.0042663

Таблиця 2.2 MSE від емпіричного підходу до оптимізації падіння

Тестування різних налаштувань для активації функцій було виконано шляхом тренування кожного з них на 500 епох. У тестах використовується однакова функція активації для вхідного шару та всіх прихованих шарів, вихідний шар може відрізнятися. Leaky ReLU для прихованих шарів і Tanh для вихідного шару дали найнижчий результат, як показано в таблиці нижче (Таблиця 2.3).

Hidden AF	Output AF	Loss
ReLU	None	0.2457
ReLU	ReLU	0.2986
ReLU	Sigmoid	0.05748
ReLU	Tanh	0.15502
lReLU	None	0.162
lReLU	lReLU	0.0586
lReLU	Sigmoid	0.0261
lReLU	Tanh	0.00656
lReLU	Softplus	0.00912

Таблиця 2.3 Результат MSE після 500 тренувань з використанням різних функцій активації

2.4 Подальші дослідження

Подальші дослідження та розробки в описаній галузі описаній галузі будуть розділені на дві окремі гілки. Перша гілка подальших досліджень буде зосереджена на системі і, зокрема, на модифікації алгоритму, щоб він міг обробляти більш високий рівень деталізації. Поточний підхід може обробляти лише 7-й рівень зі станами для 256x256x256 xyz-зразків. Подальший план полягає у наступному збільшити підтримуваний рівень деталізації до 9-го. Це означає, що він матиме можливість обробляти 512x512x512 та 1024x1024x1024 xyz-зразків відповідно. Однак, такий підхід може призвести до збільшення частоти падінь. Тому алгоритм слід модифікувати так, щоб мінімізувати це збільшення. Друга гілка є більш практичною і гілка побудови «екосистеми». Основна ідея цієї гілки полягає в тому, щоб об'єднати всі інструменти, які використовуються в один цілісний пакет та інтегрувати його з деякими рушіями, такими як Unity3D та Unreal Engine, які були

первинними ігровими рушіями, але у світлі сучаснихстають все більш універсальними інструментами, пропонуючи не лише створення ігор, але й різноманітних середовищ симуляції різноманітних середовищ. Обидва рушії Unreal та Unity підтримують 2D та повністю 3D візуалізацію. Обидва рушії використовують найновіші технології, серед яких об'ємне освітлення, фізично-базований рендеринг (PBR), постобробку, глобальне освітлення (GI), просунуті шейдери та багато іншого. Хоча розробники можуть досягти подібних результатів використовуючи будь-який з рушіїв, це, на жаль, область де Unity відстає. Unity був більш підходить для 2D та 3D розробки, тоді як Unreal від самого початку зосереджувався більше на графіці для 3D-розробки.

Інший спосіб - це використання методів кластеризації для побудови нейронної мережі. Метод кластеризації є одним з найважливіших інструментів для виявлення знань, під час якого вибірки поділяються на категорії, члени яких схожі між собою. Одне з найпоширеніших і найпоширеніших рішень для кластеризації рішень для кластеризації є кластеризація на основі розділів алгоритми, такі як K-середні та K-медоїди, які привернули багато уваги в області кластеризації клієнтів. Дані зображень у вигляді тексту є найпоширенішою формою порівняно з іншими типами даних, що зберігаються в пошукових системах типами даних, що зберігаються в пошукових системах при пошуку певного елемента серед колекції великої кількості документів. Існує кілька етапів, які передують пошуку зображень, найважливішим з яких є найважливішим з яких є кластеризація зображень. Кластеризація зображень - це процес упорядкування документів шляхом поділу їх на кілька окремих груп, кожна з яких група містить документи зі схожими тематикою і абсолютно не схожі на інші групи. Він відіграє важливу роль в інтелектуальному аналізі даних, таких як виявлення знань, розпізнавання образів і пошук інформації.

Наскільки значущою є обмін між генерацією швидкістю та вихідною роздільною здатністю при генерації 3D-моделей з 2D-зображення за допомогою згорткових нейронних мереж? На нашу користь, отримана роздільна здатність

зростає експоненціально з кожним рівнем. Деталізація зростає в геометричній прогресії, в той час як час генерації не є лінійним. Оскільки швидкість має кореляцію з кількістю мутацій, необхідних для переходу від одного рівня деталізації до іншого, а необхідна мутація отримує все менше відсоткове значення з кожним підвищенням рівня деталізації.

На закінчення, збільшення роздільної здатності також підвищує ефективність. Однак, загальна кількість мутацій все ще стрімко зростає, а це, в свою чергу, означає, що навіть якщо вища якість є технічно більш ефективною завдяки торгівлі, вона все ще не є життєздатною для створення моделей з високою роздільною здатністю, оскільки загальний час генерації буде занадто великим. Завдяки штучного інтелекту, який може допомогти у створенні 3D-середовищ, можна було б створювати ігри швидше, а малий бізнес отримав би кращу початкову позицію.

2.5 Дослідження технологій NVIDIA

Цифровий 3D-контент користується високим попитом у різних застосувань, включаючи ігри, розваги, архітектуру та моделювання робототехніки. Він поступово знаходить свій шлях у практично у всі можливі сфери: роздрібна торгівля, онлайн-конференції, віртуальна соціальна присутність, освіта тощо. Однак створення професійного 3D-контенту професійного 3D-контенту - справа не для будь-кого, вона вимагає величезної художньої та естетичної підготовки з досвідом 3D-моделювання. Розвиток цих навичок займає значну кількість часу та зусиль. Доповнення створення 3D-контенту природною мовою може значно допомогти демократизувати створення 3D-контенту як для початківців, так і для художників-експертів.

В стандарті представлено метод, який може синтезувати високодеталізовані 3D-моделі на основі текстових підказок за менший час обчислень. Зокрема, пропонується підхід до оптимізації від грубого до точного, який використовує кілька дифузійних дифузії з різною роздільною здатністю для оптимізації 3D-

представлення, що дозволяє генерувати як геометрію, що відповідає вигляду, так і деталі з високою роздільною здатністю. На першому етапі оптимізуємо грубе нейромережеве представлення, подібне до DreamFusion, але з ефективним використанням пам'яті та обчислювальних ресурсів на основі хеш-сітки. На другому етапі переходимо до оптимізації сіткового представлення, критично важливого кроку, який дозволяє нам використовувати попередні дифузії з високою роздільною здатністю 512 × 512. Оскільки 3D-сітки піддаються швидкій графіці рендери, які можуть рендерити зображення з високою роздільною здатністю в реальному часі, ми використовуємо ефективний диференційований растеризатор і використовуємо крупні плани камери для відновлення високочастотних деталей геометрії та текстури. В результаті, наш підхід створює високоякісний 3D-контент, який можна зручно імпортувати та візуалізувати у стандартному графічному програмному забезпеченні і робить це в 2 рази швидше, ніж DreamFusion. Крім того, буде продемонстровано різні творчі можливості керування процесом 3D-синтезу.

3D-синтезом, використовуючи досягнення, розроблені для додатків для редагування текстово-зображувального підходу, під назвою Magic3D, надає користувачам безпрецедентний контроль у створенні бажаних 3D-об'єктів за допомогою текстових підказок і еталонними зображеннями, що робить цю технологію ще на крок ближчою до демократизації створення 3D-контенту.

2.6 Перетворення тексту в зображення

Ми стали свідками значного прогрес у перетворенні тексту в зображення за допомогою дифузійних моделей за останні роки. Завдяки вдосконаленню моделювання та обробки даних дифузійні моделі можуть створювати складні семантичні концепції з текстових описів (іменники, прикметники, художні, стилів тощо) для створення високоякісних зображень об'єктів і сцен.

Вибірка зображень з дифузійних моделей займає багато часу. Для генерації зображень високої роздільної здатності, ці моделі або використовують каскад

моделей надвисокої роздільної здатності, або роблять вибірку з латентного простору з нижчою роздільною здатністю і декодують латенти у зображення з високою роздільною здатністю. Незважаючи на прогрес у створенні зображень з високою роздільною здатністю, використання мови для опису та керування властивостями 3D (наприклад, точки зору камери), зберігаючи при цьому точки зору камери), зберігаючи при цьому когерентність у 3D, залишається відкритою, складною проблемою.

2.7 Генеративні 3D-моделі

Існує велика кількість робіт з 3D генеративного моделювання, досліджуючи різні типи 3D-представлень, такі як 3D-воксельні сітки, хмароточкові, сітки, неявні або вісімкові представлення. Більшість з цих підходів спираються на навчальні дані у вигляді 3D-активів, які важко отримати в масштабі. Натхненні успіхом нейронного об'ємного рендерингу нейронних об'ємних зображень, нещодавні роботи почали інвестувати в синтез зображень з урахуванням 3D, який має перевагу у вивченні генеративних 3D-моделей безпосередньо з зображень, що є більш доступним ресурсом. Однак мережі об'ємного рендерингу, як правило, повільно реагують на запити, що призводить до компромісу між тривалим часом навчання і відсутністю узгодженості між декількома видами. EG3D частково пом'якшує цю проблему, використовуючи подвійний дискримінатор. Незважаючи на те, що ці роботи дають багатообіцяючі результати отримуючи багатообіцяючі результати, ці роботи залишаються обмеженими моделюванням об'єктів в межах однієї категорії об'єктів, таких як автомобілів, стільців або людських облич, що позбавляє їх масштабованості та творчого контролю, необхідних для створення 3D-контенту, творчого контролю, необхідного для створення 3D-контенту. У цьому розділі зосередження на синтезі текст-тривимірного зображення з метою створення тривимірного 3D-представлення сцени на основі текстової підказки.

2.8 Генерування тексту в 3D

Завдяки нещодавнім успіхам у генеративному моделюванні текст-зображення в останні роки, генерація текст-тривимірних зображень також викликала неабиякий інтерес у навчальній спільноті. Більш ранні роботи, такі як CLIP-forge, синтезують об'єкти, вивчаючи нормалізуючу модель потоку для вибірки вбудовування фігур з текстового введення. Однак для цього потрібні 3D-активів у воксельних представленнях під час навчання, що робить його складно масштабувати за даними. DreamField і CLIPmesh пом'якшують проблему навчальних даних, покладаючись на попередньо навчену модель зображення-тексту для оптимізації базових 3D-представлення (NeRF і сітки), так що всі 2D-візуалізації досягають високих показників вирівнювання тексту і зображення. Хоча ці підходи уникають вимоги дорогого навчання на 3D-даних і здебільшого покладаються на попередні даних і здебільшого покладаються на попередньо навчені великомасштабні моделі зображення-текст, вони, як правило, створюють менш реалістичні 2D-візуалізації.

Нещодавно DreamFusion продемонстрував вражаючі можливості синтезу тексту в 3D, використавши потужну попередньо навчену модель дифузії текст-зображення як сильне зображення попереднього. Ми будемо спиратись на цю роботу і вдосконалювати кілька варіантів дизайну, щоб надати користувачам значно точніші 3D-моделі до рук користувачів зі значно меншим часом генерації.

2.9 DreamFusion

DreamFusion забезпечує перетворення тексту в 3D за допомогою двох ключових компонентів: нейронного представлення сцени, яке ми називаємо моделлю сцени, і попередньо навчена модель перетворення тексту в зображення генеративна модель на основі дифузії. Модель сцени - це параметрична функція x = g(θ), яка може генерувати зображення x при бажаному положенні камери. Тут g - це об'ємний рендер об'ємний рендеринг, а θ - це координатний MLP, що представляє 3D об'єм. Дифузійна модель ϕ постачається з вивченою функцією зашумлення єϕ(xt; y, t), яка передбачає дискретизований шум є на основі

зашумленого зображення xt, рівня шуму t та вбудовування тексту y. Вона визначає напрямок градієнта для оновлення θ таким чином, що всі відрендерингові зображення переміщуються до областей з високою ймовірністю густини, обумовленої вбудовуванням тексту, відповідно до дифузії. Зокрема, DreamFusion запроваджує функцію дискретизації за оцінкою Distillation Sampling (SDS), яка обчислює градієнт (2.1)

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)(\epsilon_\phi(x_t; y, t) - \epsilon)\frac{\partial x}{\partial \theta}\right] \tag{2.1}$$

де w(t) - вагова функція.

Ми розглядаємо модель сцени g та модель дифузії ϕ як модульні компоненти фреймворку, які можна вибирати. На практиці функція згладжування функцію єϕ часто замінюють іншою функцією є˜ϕ, яка використовує наведення без класифікатора, що дозволяє ретельно зважувати силу кондиціонування тексту. DreamFusion покладається на великі ваги безкласифікаторного наведення для отримання результатів кращої якості. DreamFusion використовує варіант Mip-NeRF 360 з явною моделлю затінення для моделі сцени та Imagen як модель дифузії (Рис. 2.8).

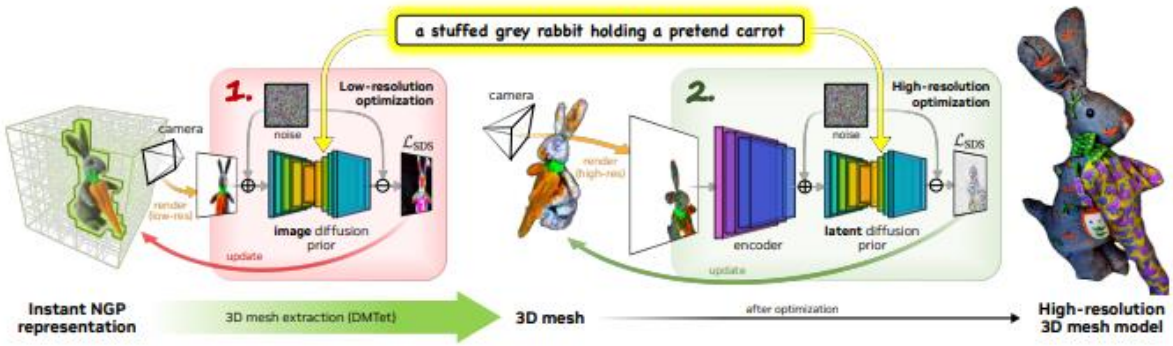

Рисунок 2.8 Огляд Magic3D. Генерується 3D-контент з високою роздільною здатністю з текстового запиту в режимі від грубого до точного. На першому етапі використовується попередня дифузія низької роздільної здатності та оптимізуємо нейронні поля (колір, щільність та нормальні поля), щоб отримати грубої моделі.

Далі диференційовано виділяється текстурована 3D-сітка з полів густини та кольору грубої моделі. Потім налаштовуємо її використовуючи модель латентної дифузії з високою роздільною здатністю. Після оптимізації модель генерує високоякісні 3D-сітки з деталізованими текстурами

Цей вибір призводить до двох ключових обмежень. По-перше, геометрія або текстури з високою роздільною здатністю не можуть бути отримані з високою роздільною здатністю, оскільки дифузійна модель працює лише на зображеннях 64×64. По-друге, використання великого глобального MLP для об'ємного рендерингу є як обчислювально дорогим, так і як і пам'ять, що робить цей підхід не дуже зручним.

2.10 Генерація 3D з високою роздільною здатністю

Magic3D - це двоетапний фреймворк від грубого до точного, який використовує ефективні моделі сцен, які дозволяють здійснювати синтез тексту в 3D з високою роздільною здатністю. Від грубої до тонкої дифузії Magic3D використовує дві різні попередні дифузії в режимі від грубої до точної для створення геометрії та текстур високої роздільної здатності текстур високої роздільної здатності. На першому етапі ми використовуємо базову модель дифузії описану в eDiff-I, яка схожа на базову модель дифузії Imagen, що використовується в DreamFusion. Цей дифузії використовується для обчислення градієнтів сцени моделі за допомогою втрат, визначених на зображенні з низькою роздільною здатністю 64x64. На другому етапі ми використовуємо модель латентної дифузії (LDM), яка дозволяє зворотне розповсюдження градієнтів на відрендерених зображеннях з високою роздільною здатністю 512x512; на практиці ми вирішили використовувати загальнодоступну модель Stable Diffusion модель Stable Diffusion. Незважаючи на створення зображень з високою роздільною здатністю, обчислення LDM є керованим, оскільки дифузія попередньо діє на латентний zt з роздільною здатністю 64x64 (2.2)

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)(\epsilon_\phi(z_t; y, t) - \epsilon)\frac{\partial z}{\partial x}\frac{\partial x}{\partial \theta}\right] \quad (2.2)$$

Збільшення часу обчислень в основному пов'язане з обчисленням ∂x/ ∂θ (градієнт зображення з високою роздільною здатністю) та ∂z/ ∂x (градієнт кодера в LDM).

2.11 Моделі сцен

Пропонується два представлення 3D-сцени: для грубих та точних зйомок з різною роздільною здатністю. Грубі моделі, такі як DreamFusion, використовують нейронні поля для прогнозування альбедо і щільності. Однак обчислювальні витрати можуть бути великими. Тому ми перейшли на кодування хеш-сітки від Instant NGP, що дозволяє представляти високочастотні деталі з меншими витратами. На етапі тонкої оптимізації ми використовуємо текстуровані 3D-сітки для рендерингу з дуже високою роздільною здатністю, що дозволяє ефективно виконувати оптимізацію.

Я пропоную два типи 3D-сцен: для грубих і точних зйомок з різною роздільною здатністю. Грубі моделі використовують нейронні поля, такі як DreamFusion, що прогнозують альбедо і щільність. Однак це дорого обчислювально. Ми перейшли на хеш-сітку Instant NGP, що зменшує витрати. На тонкій стадії оптимізації ми використовуємо текстуровані

3D-сітки для рендерингу з високою роздільною здатністю.

2.12 Оптимізація від грубої до точної

Опишемо нашу процедуру оптимізації від грубої до точної, яка спочатку оперує з грубим представленням нейронного поля а потім на текстурованій сітці з високою роздільною здатністю. Оптимізація нейронного поля. Подібно до Instant NGP, ми ініціалізуємо сітку заповнення з роздільною здатністю 2563 зі значенням 20, щоб стимулювати ріст фігур на ранніх стадіях оптимізації. Ми оновлюємо сітку кожні 10 ітерацій і генеруємо вісімницю для пропуску порожніх місць. Ми

розкладаємо сітку заповнюваність сітки на 0.6 при кожному оновленні і слідуємо миттєвому NGP з тими ж параметрами оновлення та порогових значень.

Замість того, щоб оцінювати нормалі за різницею густини, ми використовуємо МНК для передбачення нормальних значень. Зауважте, що це не порушує геометричних властивостей, оскільки використовується об'ємний рендер замість поверхневого, тому орієнтація частинок у неперервних позиціях не обов'язково має бути орієнтована на задану поверхню рівня. Це допомагає нам значно зменшити обчислювальні витрати на оптимізацію грубої моделі за рахунок уникнення використання скінченного диференціювання. Точні нормалі можна отримати на тонкому етапі оптимізації, коли ми використовуємо справжню модель рендерингу поверхні. Подібно до DreamFusion, ми також моделюємо фон за допомогою карти оточення MLP, яка прогнозує кольори RGB як функцію напрямку променів. Оскільки наша модель розрідженого представлення не підтримує репараметризацію сцени, як у Mip-NeRF 360, оптимізація має тенденцію до «обманювати», вивчаючи суть об'єкта за допомогою карти фонового оточення. Таким чином, ми використовуємо крихітний MLP для карти оточення (розмір прихованої розмірності 16) і зменшуємо швидкість навчання в 10 разів, щоб дозволити моделі сфокусуватися на геометрії нейронного поля. Оптимізація сітки. Для оптимізації сітки, отриманої при ініціалізації нейронного поля, ми перетворюємо (грубе) поле густини в SDF, віднімаючи його з ненульовою константою, отримуючи початкове значення si. Додатково ініціалізуємо поле об'ємної текстури безпосередньо з полем кольору, оптимізованим на грубому етапі. Під час оптимізації ми рендеримо витягнуту поверхню сітку у зображення з високою роздільною здатністю за допомогою диференційованого растеризатора. Ми оптимізуємо si та Δvi для кожної вершини vi за допомогою зворотного розповсюдження з використанням градієнта SDS високої роздільної здатності. При рендерингу мешу в зображення ми також відстежуємо 3D координати кожного відповідного пікселя проекції, які будуть використані для запиту кольорів у відповідному полі текстури для спільної оптимізації.

Під час рендерингу сіті ми збільшуємо фокусну відстань, щоб збільшити деталі об'єкта, що є важливим кроком на шляху до відновлення високочастотних деталей. Ми зберігаємо ту саму попередньо навчену карту оточення з грубої стадії оптимізації і компонуємо відрендерений фон з відрендереним об'єктом переднього плану об'єктом переднього плану за допомогою диференційованого згладжування. Для підвищення гладкості поверхні ми додатково впорядковуємо кутові різниці між сусідніми гранями на сітці. Це дозволяє нам отримати добре геометрію навіть за наявності сигналів спостереження з високою дисперсією, таких як градієнт SDS ∇θLSDS (Рис. 2.9).

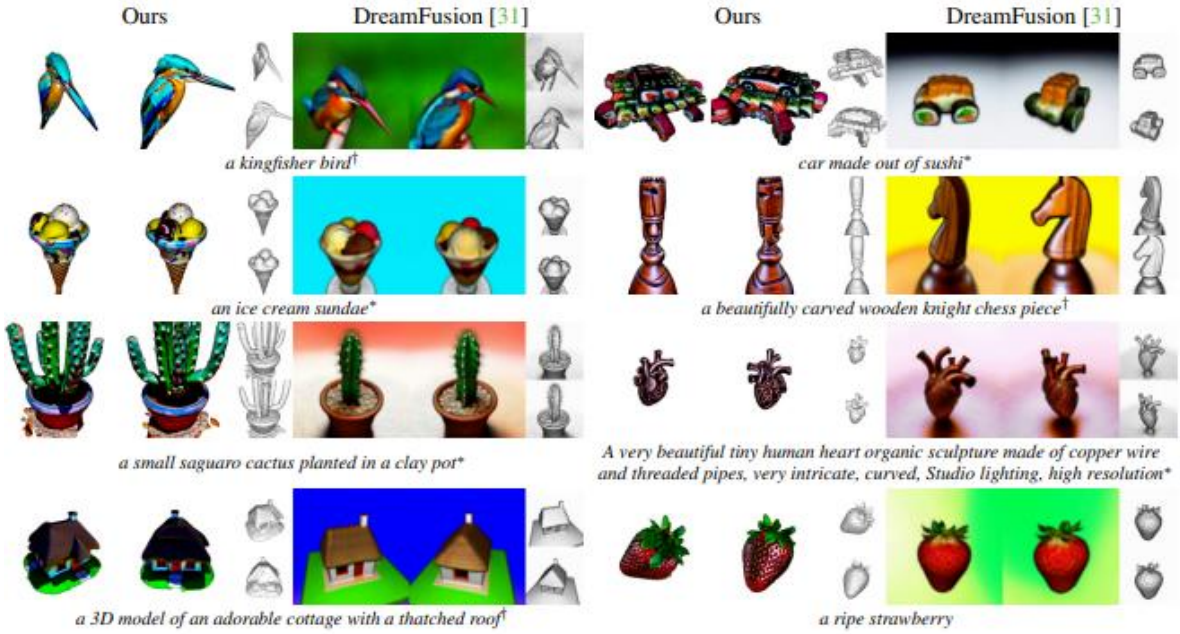

Рисунок 2.9 Порівняння якості Magic3D з DreamFusion

2.13 Керована генерація 3D

Оскільки певні стилі та концепції важко виразити словами, але легко за допомогою зображень, бажано мати можливість впливати на процес генерації словами, а також за допомогою зображень, бажано мати механізм впливу на генерацію 3D-моделі з тексту за допомогою зображень. Досліджуються різні методи обробки зображень, а також підхід до редагування на основі підказок.

2.14 Персоналізований текст у 3D

DreamBooth описано метод персоналізації моделей дифузії текст-зображення шляхом точного налаштування попередньо навченої моделі на декількох зображеннях об'єкта. Налагоджена модель може навчитися прив'язувати об'єкт до унікальним рядком ідентифікатора, позначеним як [V], і генерувати зображення об'єкта, коли [V] міститься у текстовій підказці.
У контексті перетворення тексту в 3D ми хотіли б створити 3D-модель предмета. Цього можна досягти, спочатку доопрацювавши наші попередні моделі дифузії за допомогою підходу DreamBooth, а потім використавши відрегульовані дифузійні пріори з ідентифікатором [V] як частину текстової підказки для навчання, щоб для надання навчального сигналу під час оптимізації 3D-моделі. Щоб продемонструвати застосовність DreamBooth у нашому фреймворку зібрано 11 зображень одного кота і 4 зображення однієї собаки. Налаштовано eDiff-I та LDM, прив'язавши текстовий ідентифікатор [V] до заданого об'єкта. Оптимізуємо 3D-модель з [V] у текстових підказках. Ми можемо успішно модифікувати 3D-моделі, зберігаючи об'єкти на заданих вхідних зображеннях (Рис. 2.10).

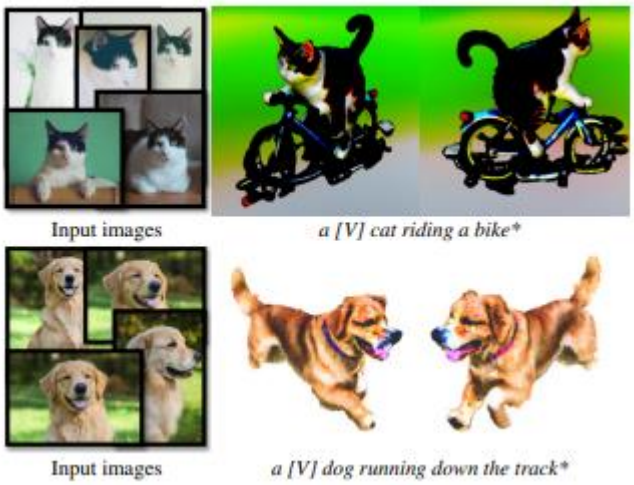

Рисунок 2.10 Magic3D з персоналізацією на основі DreamBooth.

2.15 Оперативне редагування на основі точного налаштування

Інший спосіб контролювати створений 3D-вміст - це точне налаштування вивченої грубої моделі за допомогою нової підказки. Наше редагування на основі підказок редагування на основі підказок складається з трьох етапів.

1. Ми навчаємо грубу модель
2. Ми змінюємо базову підказку і допрацьовуємо грубу модель за допомогою LDM. На цьому етапі ми отримуємо добре ініціалізовану NeRF модель для наступного кроку. Безпосередньо застосування оптимізації сіті до нової підказки створить високодеталізовані текстури, але може деформувати геометрію лише незначно деформує геометрію.
3. Оптимізуємо сіть за допомогою зміненої текстової підказки

Редагування на основі підказки може змінювати текстуру фігури або трансформувати геометрію і текстуру відповідно до тексту. Отримані моделі сцени зберігають розташування шарів і загальну структуру. Така можливість редагування робить створення 3D-контенту за допомогою Magic3D більш контрольованим.

На рисунку показано дві грубі NeRF-моделі, навчені за допомогою базової підказки базовою підказкою для "зайчика" та "білочки". Ми модифікуємо базову підказку, доопрацьовуємо нейронну модель у високій роздільній здатності та оптимізуємо сітку. Результати показують, що ми можемо налаштувати модель сцени відповідно до підказки, наприклад, змінити наприклад, замінивши ´зайчика на ´вітражного зайчика або ´металевого зайчика. ми отримуємо схожу геометрію, але з іншою текстурою (Рис 2.11).

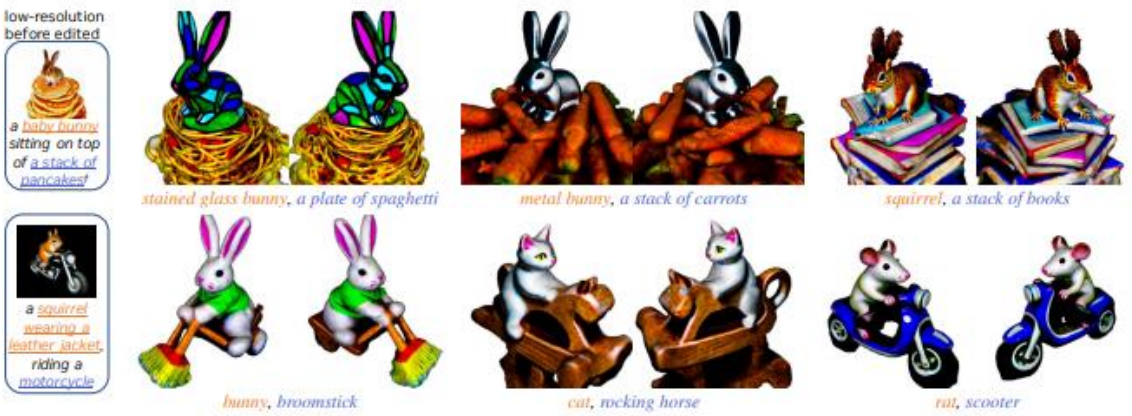

Рисунок 2.11 Magic3D з редагуванням на основі підказок. Маючи грубу модель (перший стовпчик), згенеровану за допомогою базової підказки, ми замінюємо підкреслений текст новим текстом і точно налаштовуємо NeRF, щоб отримати NeRF-модель з високою роздільною здатністю з LDM. Далі ми допрацьовуємо сітку з високою роздільною здатністю за допомогою NeRF-моделі. Такий метод редагування на основі підказок дає художникам кращий контроль над виходом 3D-генерації.

3 ІНСТРУМЕНТ ДЛЯ ГЕНЕРАЦІЇ

Інструмент для генерації 3Д-моделі буде створений на ігровому рушію Unity 2022.3.28.f1 LTS. Одним з ключових умов для подальшого створення інструменту - це розширення самого функціоналу робочої середи. В Unity для цього є всі можливості.

3.1 Graph Neural Networks

Сам функціонал створення буде відбуватись за допомогою нейромережі Meshy AI. Нейронна мережа Meshy використовує метод Graph Neural Networks (GNN), або графових нейронних мереж. GNN призначені для обробки даних, представлених у вигляді графів. Графи складаються з вузлів (вершин) та ребер (зв’язків між вузлами), що дозволяє моделювати складні структури даних.

Особливості та компоненти GNN

- Вузли та ребра:
  - Вузли (Nodes): Представляють сутності або об'єкти.
  - Ребра (Edges): Представляють зв’язки або відносини між сутностями.
- Передача повідомлень (Message Passing):
  - Один з основних механізмів GNN, де вузли обмінюються інформацією зі своїми сусідами. Це дозволяє вузлам отримувати інформацію про їхнє оточення.
- Агрегація інформації (Aggregation):
  - Після передачі повідомлень вузли агрегують отриману інформацію, щоб оновити свої власні ознаки. Існують різні методи агрегації, такі як сума, середнє або максимальне значення.
- Обробка сигналів на графах:
  - Використовується для аналізу сигналів, визначених на вузлах графу, що дозволяє обробляти складні залежності між даними.

Переваги використання GNN

- Врахування структури даних:

- GNN здатні враховувати складні взаємозв'язки між даними, що робить їх ефективними для задач, де дані природно представлені у вигляді графів.

- Гнучкість і масштабованість:
  - GNN можуть бути застосовані до різних типів графів, включаючи статичні та динамічні, спрямовані та неспрямовані.
- Потужність для складних задач:
  - Використовуються в задачах соціального аналізу, біологічних мереж, рекомендаційних системах, обробці молекулярних структур та інших.

Як працюють GNN

Досі глибинне навчання в основному зосереджувалося на зображеннях і тексті - типах структурованих даних, які можна описати як послідовності слів або сітки пікселів. Графіки, навпаки, неструктуровані. Вони можуть мати будь-яку форму або розмір і містити будь-які дані, включаючи зображення і текст. Використовуючи процес, який називається передачею повідомлень, GNN організовують графіки так, щоб алгоритми машинного навчання могли їх використовувати. Передача повідомлень вбудовує в кожен вузол інформацію про його сусідів. Моделі штучного інтелекту використовують вбудовану інформацію для пошуку закономірностей і прогнозування.

Наприклад, системи рекомендацій використовують форму вбудовування вузлів у GNN, щоб зіставити клієнтів з продуктами. Системи виявлення шахрайства використовують реберні вбудовування для пошуку підозрілих транзакцій, а моделі пошуку ліків порівнюють цілі графи молекул, щоб з'ясувати, як вони реагують одна на одну.

GNN унікальні ще в двох аспектах: Вони використовують розріджену математику, і моделі, як правило, мають лише два або три шари. Інші моделі ШІ зазвичай використовують щільну математику і мають сотні нейромережевих шарів (Рис. 3.1).

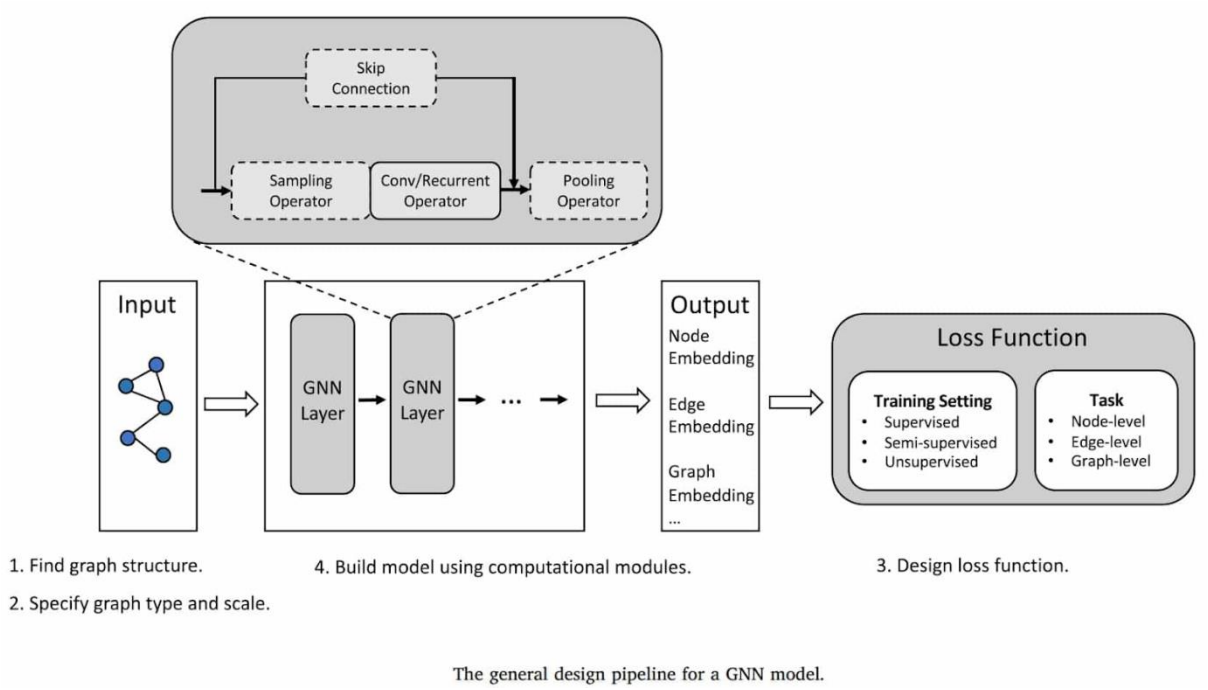

Рисунок 3.1 Схема візуалізації GNN

Meshy, з використанням GNN застосовується для аналізу та обробки даних, які мають складну внутрішню структуру і можуть бути представлені у вигляді графів. Це може включати задачі, пов’язані з аналізом мереж взаємодій, побудовою моделей для передбачення зв’язків або класифікації вузлів, а також іншими типами задач, де структура графу є критичною для досягнення високої точності та ефективності.

Отже, основним методом, який використовує нейромережа Meshy, є Graph Neural Networks (GNN), що дозволяє ефективно працювати з графовими структурами даних.

3.2 Інтеграція Meshy та створення інструменту

Для інтеграції функціоналу необхідно додати потрібні пакети через url або через package manager (Рис. 3.2).

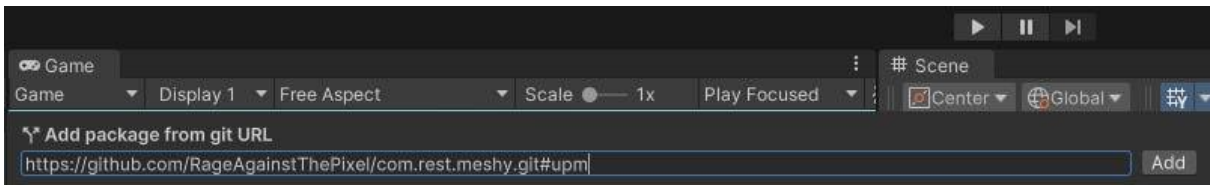

Рисунок 3.2 Додавання функціоналу Meshy

В самому проєкті нам необхідно створити папку MeshyAuthentication. В цій папці буде тільки один скрипт, який буде зберігати API та викликати його кожного разу, коли це потрібно (Рис. 3.3).

```
var api = new MeshyClient(new MeshyAuthentication("msy_apikey123"));
```

Рисунок 3.3 Зберігання API

Після додавання функціаналу Meshy в Unity з’явиться дашборд. Для того щоб дешборд не був “пустим” потрібно створити скрипт з необхідними командами.

Перетворення з тексту до текстури

```
var api = new MeshyClient();
var textToTextureTasks = await api.TextToTextureEndpoint.ListTasksAsync(1, 12, SortOrder
    .Descending);

foreach (var meshyTask in textToTextureTasks)
{
    Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");
}
```

Рисунок 3.4 Перерахування тексту до завдань текстури

Після отримання в дешборді значень або прописування параметрів вручну (як показано на рисунку нижче) відправляється запит на генерацію текстури (Рис. 3.5).

```
var sphere = GameObject.CreatePrimitive(PrimitiveType.Sphere);
var request = new TextToTextureRequest(sphere, "Basketball", "game assets", enableOriginalUV:
    false, resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);
var taskResult = await MeshyClient.TextToTextureEndpoint.CreateTextToTextureTaskAsync(request,
    new Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}:
    {progress.PrecedingTasks ?? progress.Progress}")));
Assert.IsNotNull(taskResult);
Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult
    .ExpiresAt}");
```

Рисунок 3.5 Прописаний вручну запит на генерацію текстури

Створення нового тексту для завдання текстурування і буде повідомляє про прогрес, поки завдання не буде завершено

Перетворення з тексту до 3Д-моделі

Необхідні параметри отримуються через дешборд або прописуються вручну (так як показано перетворення до текстури) (Рис3.6)

```
var textTo3DTasks = await MeshyClient.TextTo3DEndpoint.ListTasksAsync(1, 12, SortOrder
    .Descending);

foreach (var meshyTask in textTo3DTasks)
{
    Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");
}
```

Рисунок 3.6 Перерахування тексту до завдань генерації моделі

Створення нового тексту до 3D-завдання (Рис. 3.7)

```
var request = new TextTo3DRequest("a treasure chest", "realistic, wooden, carved, highest
    quality", resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);
var taskResult = await MeshyClient.TextTo3DEndpoint.CreateTextTo3DTaskAsync(request, new
    Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress
    .PrecedingTasks ?? progress.Progress}")));
Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult
    .ExpiresAt}");
```

Рисунок 3.7 Прописаний вручну запит на генерацію моделі

З картинки до моделі

Для генерації 3Д-моделі з картинки потрібно зробити схожі дії, але змінити запит (Рис. 3.8)

```
var api = new MeshyClient();
var imageTo3dTasks = await api.ImageTo3DEndpoint.ListTasksAsync(1, 12, SortOrder.Descending);

foreach (var meshyTask in imageTo3dTasks)
{
    Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");
}

var imageUrl = "https://raw.githubusercontent.com/KhronosGroup/glTF-Sample-Assets/main/Models
    /Fox/screenshot/screenshot-x150.jpg";
var request = new ImageTo3DRequest(imageUrl);
var taskResult = await api.ImageTo3DEndpoint.CreateImageTo3DTaskAsync(request, new Progress
    <TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress
    .PrecedingTasks ?? progress.Progress}")));
Assert.IsNotNull(taskResult);
Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult
    .ExpiresAt}");
```

Рисунок 3.8 Прописаний вручну запит на генерацію моделі

3.3 Показ інструменту

Через дешборд можна обрати спосіб перетворення до 3Д моделі. Також можна обрати стиль створення: реальність, мультики, low-poly.

Після створення моделі її можна перестворити з різної інтенсивністю змін (Рис. 3.9)

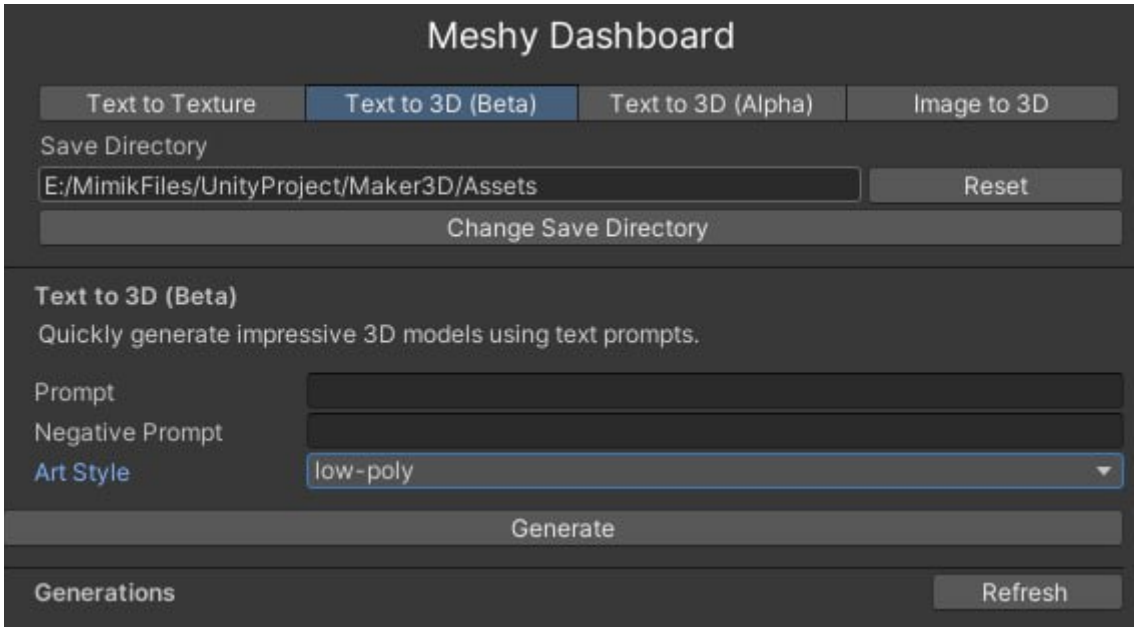

Рисунок 3.9 Інтерфейс дешборду

Після вдалого створення моделі її можна додати в проєкт і використовувати як 3Д-модель (Рис. 3.10)

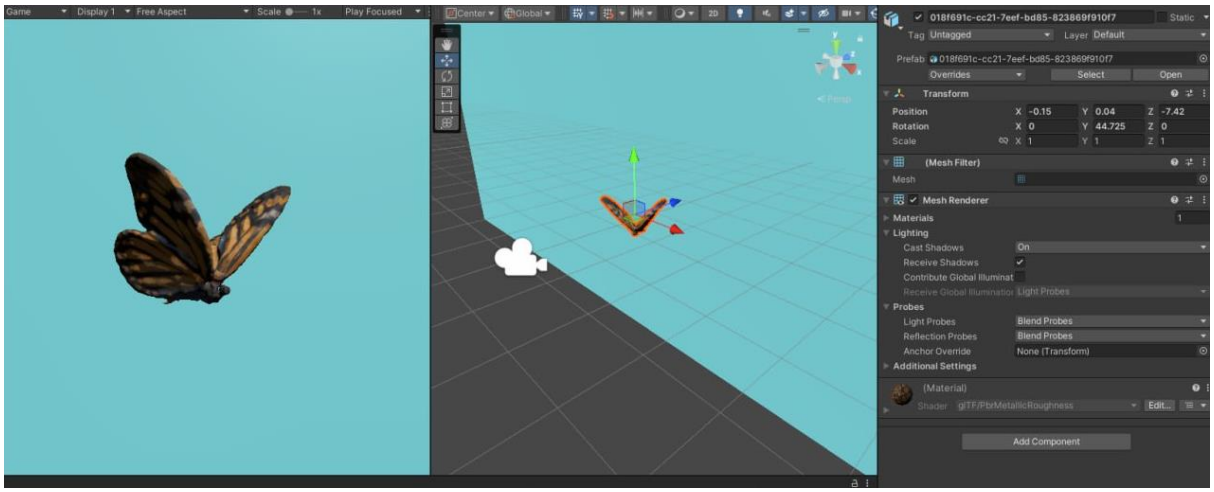

Рисунок 3.10 Демонстрація 3Д-моделі в редакторі

Якщо була вибрана опція, щоб створити 3Д-модель, то для неї також створюються всі потрібні компоненти (матеріал, меш-сітка, текстури), при необхідності їх можна замінити (Рис. 3.11)

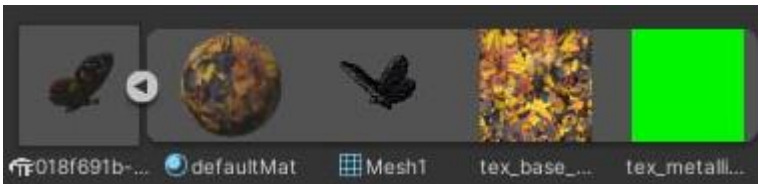

Рисунок 3.11 Демонстрація всього необхідного набору 3Д-моделі

ВИСНОВОК

Дивлячись на результат, можна припустити ,що в майбутньому подібна система може бути використана для генерації 3D моделей. Основна ідея цієї роботи полягала в отриманні знань про різні проблемні питання генерації 3D моделей з 2D зображень. Однак, при поточній реалізації, вихідні дані є надто низькими та повільними для високої роздільної здатності.

Для традиційного створення 3Д-моделі в професійній команді потрібно 3 людини:

- дизайнер,який намалює арт моделі
- 3Д-художник, який зробить цю 3Д-модель
- програміст, який буде в подальшому з нею працювати

На розробку моделі може піти не один день. Розроблений мною модуль зменшує часові затрати до декількох хвилин. При розробці , наприклад, гри часто необхідно маленькі деталі на пошук яких виділяється дорогоцінний час, при цьому сама розробка гри з місця не зрушується . Згенеровані моделі можна використовувати одразу або віддати на допрацювання відповідним спеціалістам.

ПЕРЕЛІК ПОСИЛАНЬ

1. Bebesko B., Khorolska K., Kotenko N. 3d modelling by means of artificial intelligence. Kyiv, 2021. P. 1296–1308.
2. Chang A. Learning spatial knowledge for text to 3D scene generation. *Proceedings of empirical methods in natural language processing*. Qatar, 2020. P. 2028–2038.
3. Magic3D: High-Resolution Text-to-3D Content Creation / C.-H. Lin et al. *IEEE conference on computer vision and pattern recognition*. 2023. P. 300–309. URL: https://doi.org/10.48550/arXiv.2211.10440 (date of access: 28.05.2024).
4. MashyAI. Meshy. MeshyAI. Version 1.0.0. URL: https://github.com/mechanistry/meshy (date of access: 28.05.2024).
5. Merritt R. What are graph neural networks?. *NVIDIA Blog*. URL: https://blogs.nvidia.com/blog/what-are-graph-neural-networks/ (date of access: 28.05.2024).
6. Sanchez-Lengeling B. A gentle introduction to graph neural networks. *Distill*. URL: https://distill.pub/2021/gnn-intro/ (date of access: 28.05.2024).
7. Shaikhnag A. Generating 3D models from text with Nvidia's Magic3D - 3D Printing Industry. *3D Printing Industry*. URL: https://3dprintingindustry.com/news/generating-3d-models-from-text-with-nvidias-magic3d-220520/ (date of access: 28.05.2024).
8. Viviani M. How ai-generated 3D models are transforming 3D pipelines. *Medium*. URL: https://medium.com/@nvidiaomniverse/how-ai-generated-3d-models-are-transforming-3d-pipelines-d4245566a289 (date of access: 28.05.2024).
9. Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. In Advances in Neural Information Processing Systems (NeurIPS), 2022
10. Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 18603-18613, 2022.
11. Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. arXiv preprint arXiv:2205.11487, 2022.
12. Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. arXiv preprint arXiv:2208.12242, 2022

13. Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 10684-10695, June 2022.
14. Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 2022.
15. Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In Advances in Neural Information Processing Systems (NeurIPS), 2022.
16. Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11453±11464, 2021.
17. Zekun Hao, Arun Mallya, Serge Belongie, and Ming-Yu Liu. GANcraft: Unsupervised 3D Neural Rendering of Minecraft Worlds. In ICCV, 2021.
18. Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for highresolution image synthesis. In International Conference on Learning Representations, 2022.

ДОДАТКИ

ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛЯ

```
var api = new MeshyClient(new MeshyAuthentication("msy_apikey123"));


var api = new MeshyClient();

var textToTextureTasks = await api.TextToTextureEndpoint.ListTasksAsync(1, 12, SortOrder.Descending);

foreach (var meshyTask in textToTextureTasks)

{ Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}"); }

var sphere = GameObject.CreatePrimitive(PrimitiveType.Sphere);

var request = new TextToTextureRequest(sphere, "Basketball", "game assets", enableOriginalUV: false, resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);

var taskResult = await MeshyClient.TextToTextureEndpoint.CreateTextToTextureTaskAsync(request, new Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress.PrecedingTasks ?? progress.Progress}")));

Assert.IsNotNull(taskResult);

Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult.ExpiresAt}");


var api = new MeshyClient();

var textTo3DTasks = await MeshyClient.TextTo3DEndpoint.ListTasksAsync(1, 12, SortOrder.Descending);

foreach (var meshyTask in textTo3DTasks)

{

   Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");

}
```

```
var request = new TextTo3DRequest("a treasure chest", "realistic, wooden, carved, highest quality", resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);

var taskResult = await MeshyClient.TextTo3DEndpoint.CreateTextTo3DTaskAsync(request, new Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress.PrecedingTasks ?? progress.Progress}")));

Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult.ExpiresAt}");


var api = new MeshyClient();

var imageTo3dTasks = await api.ImageTo3DEndpoint.ListTasksAsync(1, 12, SortOrder.Descending);

foreach (var meshyTask in imageTo3dTasks)

{

   Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");

}

var api = new MeshyClient();

var imageUrl = "https://raw.githubusercontent.com/KhronosGroup/glTF-Sample-Assets/main/Models/Fox/screenshot/screenshot-x150.jpg";

var request = new ImageTo3DRequest(imageUrl);

var taskResult = await api.ImageTo3DEndpoint.CreateImageTo3DTaskAsync(request, new Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress.PrecedingTasks ?? progress.Progress}")));

Assert.IsNotNull(taskResult);

Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult.ExpiresAt}");
```

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

КВАЛІФІКАЦІЙНА РОБОТА
НА ТЕМУ :"РОЗРОБКА UNITY-МОДУЛЯ ГЕНЕРАЦІЇ 3D-МОДЕЛЕЙ ПО ТЕКСТОВОМУ ЗАПИТУ ЗА ДОПОМОГОЮ МЕТОДІВ ШТУЧНОГО ІНТЕЛЕКТУ "

Виконав: здобувач вищої освіти гр. ШІД-41
Максим ПАВЛЮК
Керівник: старший викладач Тетяна КИСІЛЬ

2024 рік

Рисунок 5.1 Слайд 1

Мета роботи: прискорення процесу створення 3Д-моделей в ігровому рушію Unity за допомогою штучного інтелекту

Об'єкт дослідження: процес прискорення створення 3Д-моделей

Предмет дослідження: штучний інтелект та нейромережі

Основні завдання кваліфікаційної роботи:

1. аналіз можливостей штучного інтелекту в розробці ігор
2. Дослідження обчислень та генерації 3Д-моделі
3. Модуль генерації 3Д-моделі

Рисунок 5.2 Слайд 2

БАГАТОПЛАТФМОРМНИЙ ІНСТРУМЕНТ ТА ІГРОВИЙ РУШІЙ UNITY

1

- UNITY - ЦЕ КРОСПЛАТФОРМНИЙ ІГРОВИЙ РУШІЙ, РОЗРОБЛЕНИЙ КОМПАНІЄЮ UNITY TECHNOLOGIES, ВПЕРШЕ ВИПУЩЕНИЙ У 2005 РОЦІ. В ОСНОВІ UNITY ЛЕЖИТЬ ЗРУЧНИЙ ІНТЕРФЕЙС У ПОЄДНАННІ З ПОТУЖНИМИ ІНСТРУМЕНТАМИ ТА ФУНКЦІОНАЛОМ, ЩО РОБИТЬ ЙОГО ДОСТУПНИМ ЯК ДЛЯ ДОСВІДЧЕНИХ РОЗРОБНИКІВ, ТАК І ДЛЯ ПОЧАТКІВЦІВ.

Рис. 1 Ігровий рушій Unity

Рисунок 5.3 Слайд 3

АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

2

MIDJOURNY: ЗА ДОПОМОГОЮ MIDJOURNY МОЖНА СТВОРЮВАТИ РІЗНІ МЕТОДИ ГЕНЕР АЦІЇ, ЯКІ ДОПОМАГАЮТЬ АВТОМАТИЗУВАТИ ПРОЦЕС СТВОРЕННЯ ВМІСТУ. ЗОКРЕМА, MI DJOURNY ВИКОРИСТОВУЄ ТАКІ МЕТОДИ ГЕНЕРАЦІЇ -
DO DINMICO, LLUIS LLACH, LOLA FLORES, MARISOL, JULIO IGLESIAS, NINO BRAVO, C ARLOS GARDEL

UNITY MUSE: ПРАЦЮЄ НА БАЗІ ШТУЧНОГО ІНТЕЛЕКТУ НОВОГО ПОКОЛІННЯ, ДОЗВОЛЯ Є ТВОРЦЯМ УСІХ РІВНІВ КВАЛІФІКАЦІЇ БЛИСКАВИЧНО РОЗРОБЛЯТИ ІГРИ NA 3D-ДОДАТКИ В РЕАЛЬНОМУ ЧАСІ ЗА ДОПОМОГОЮ ПРОСТИХ ТЕКСТОВИХ ПІДКАЗОК.

UNITY SENTIS: ЦЕ БІБЛІОТЕКА ДЛЯ ІНФЕРЕНСУ НЕЙРОННИХ МЕРЕЖ В СЕРЕДОВИЩІ UNITY. ВОНА ДОЗВОЛЯЄ ІМПОРТУВАТИ НАВЧЕНІ МОДЕЛІ НЕЙРОННИХ М ЕРЕЖ В UNITY ТА ВИКОНУВАТИ ЇХ В РЕАЛЬНОМУ ЧАСІ НА БУДЬ-ЯКІЙ ПЛАТФОРМІ, ЯКУ ПІДТРИМУЄ UNITY, ВКЛЮЧАЮЧИ РЕДАКТОР

Рисунок 5.4 Слайд 4

3

Машинне навчання (ML)

це розділ штучного інтелекту, який використовується для того, щоб змусити системи автоматично вчитися на власному досвіді. Замість того, щоб програмувати їх на те, як саме виконувати певну роботу. Таким чином, його можна використовуватися для вирішення проблем без необхідності вказувати, як розв'язувати поставлену задачу.

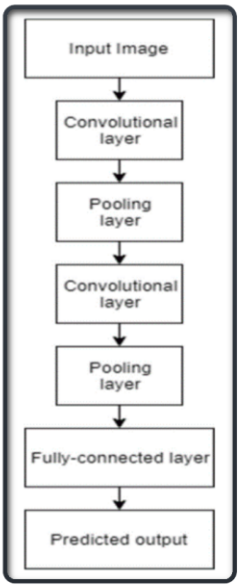

Рис.2 Візуальне представлення алгоритму згорткової нейронної мережі

Рисунок 5.5 Слайд 5

4

GENERATIVE ADVERSARIAL NETWORK

Згорткова нейронна мережа (CNN) - це різновид мережа, натхненна зорово ю корою головного мозку, успадкован а від багатошарових персептронів (MLP S).

CNN це своєрідний механізм, який можна використовувати для представл ення даних з невеликою кількістю параметрів, напри клад класифікацією.

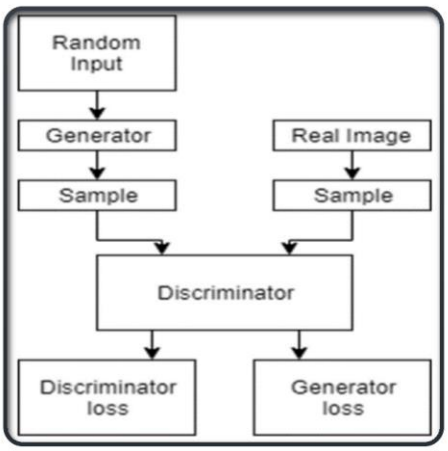

Рис. 3 Візуальне представлення генеративної змагальної мережі

Рисунок 5.6 Слайд 6

DREAMFUSION ТА MAGIC3D

5

DREAMFUSION ЗАБЕЗПЕЧУЄ ПЕРЕТВОРЕННЯ ТЕКСТУ В 3D ЗА ДОПОМОГОЮ ДВОХ КЛЮЧОВИХ КОМПОНЕНТІВ: НЕЙРОННОГО ПРЕДСТАВЛЕННЯ СЦЕНИ, ЯКЕ МИ НАЗИВАЄМО МОДЕЛЛЮ СЦЕНИ, І ПОПЕРЕДНЬО НАВЧЕНА МОДЕЛЬ ПЕРЕТВОРЕННЯ ТЕКСТУ В ЗОБРАЖЕННЯ ГЕНЕРАТИВНА МОДЕЛЬ НА ОСНОВІ ДИФУЗІЇ.

MAGIC3D - ЦЕ ДВОЕТАПНИЙ ФРЕЙМВОРК ВІД ГРУБОГО ДО ТОЧНОГО, ЯКИЙ ВИКОРИСТОВУЄ ЕФЕКТИВНІ МОДЕЛІ СЦЕН, Я КІ ДОЗВОЛЯЮТЬ ЗДІЙСНЮВАТИ СИНТЕЗ ТЕКСТУ В 3D З ВИСОКОЮ РОЗДІЛЬНОЮ ЗДАТНІСТЮ.

Рисунок 5.7 Слайд 7

ІНТЕГРАЦІЯ MESHY ТА СТВОРЕННЯ ІНСТРУМЕНТУ

6

Для інтеграції функціоналу необхідно додати потрібні пакети через url або через package manager

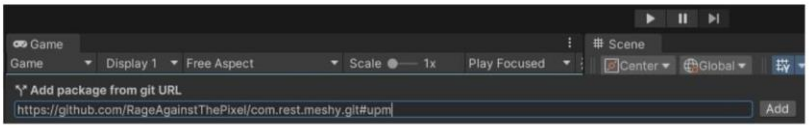

Рис. 4 Додавання функціоналу Meshy

В самому проєкті нам необхідно створити папку MeshyAuthentication. В цій папці буде тільки один скрипт, який буде зберігати API та викликати його кожного разу, коли це потрібно

```
var api = new MeshyClient(new MeshyAuthentication("msy_apikey123"));
```


Рис. 5 Зберігання API

Русинок 5.8 Слайд 8

ПЕРЕТВОРЕННЯ З ТЕКСТУ ДО ТЕКСТУРИ

7

```
var api = new MeshyClient();
var textToTextureTasks = await api.TextToTextureEndpoint.ListTasksAsync(1, 12, SortOrder
    .Descending);

foreach (var meshyTask in textToTextureTasks)
{
    Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");
}
```

```
var sphere = GameObject.CreatePrimitive(PrimitiveType.Sphere);
var request = new TextToTextureRequest(sphere, "Basketball", "game assets", enableOriginalUV:
    false, resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);
var taskResult = await MeshyClient.TextToTextureEndpoint.CreateTextToTextureTaskAsync(request,
    new Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}:
    {progress.PrecedingTasks ?? progress.Progress}")));
Assert.IsNotNull(taskResult);
Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult
    .ExpiresAt}");
```

Рис. 6 Запит на генерацію текстури

Після отримання в дешборді значень або прописування параметрів вручну відправляється запит на генерацію текстури. Створення нового тексту для завдання текстурування і буде повідомляє про прогрес, поки завдання не буде завершено

Рисунок 5.9 Слайд 9

ПЕРЕТВОРЕННЯ З ТЕКСТУ ДО 3Д-МОДЕЛІ

8

```
var textTo3DTasks = await MeshyClient.TextTo3DEndpoint.ListTasksAsync(1, 12, SortOrder
    .Descending);

foreach (var meshyTask in textTo3DTasks)
{
    Debug.Log($"{meshyTask.Id} | created_at: {meshyTask.CreatedAt}");
}
```

```
var request = new TextTo3DRequest("a treasure chest", "realistic, wooden, carved, highest
    quality", resolution: Resolutions.X1024, artStyle: ArtStyles.Realistic);
var taskResult = await MeshyClient.TextTo3DEndpoint.CreateTextTo3DTaskAsync(request, new
    Progress<TaskProgress>(progress => Debug.Log($"[{progress.Id}] {progress.Status}: {progress
    .PrecedingTasks ?? progress.Progress}")));
Debug.Log($"{taskResult.Id} | created_at: {taskResult.FinishedAt} | expires_at: {taskResult
    .ExpiresAt}");
```

Рис. 7 запит на генерацію моделі

Необхідні параметри отримуються через дешборд або прописуються вручну

Рисунок 5.10 Слайд 10

МОДУЛЬ ДЛЯ ГЕНЕРАЦІЇ

9

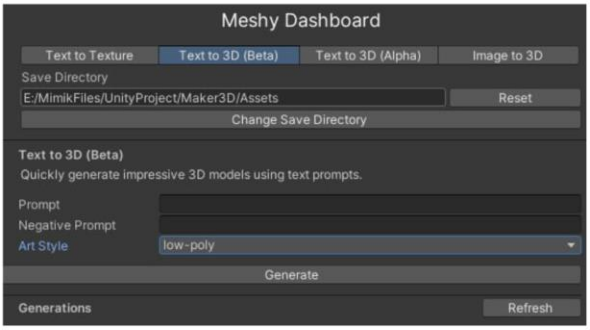

Рис. 8 Інтерфейс дешборду

Через дешборд можна обрати спосіб перетворення до 3Д моделі. Також можна обрати стиль створення: реальність, мультики, low-poly.
Після створення моделі її можна перестворити з різної інтенсивністю змін. Після вдалого створення моделі її можна додати в проєкт і використовувати як 3Д-модель

Рисунок 5.11 Слайд 11

ДЕМОНСТРАЦІЯ ГЕНЕРАЦІЇ

10

Якщо була вибрана опція, щоб створити 3Д-модель, то для неї також створюються всі потрібні компоненти (матеріал, меш-сітка, текстури), при необхідності їх можна замінити

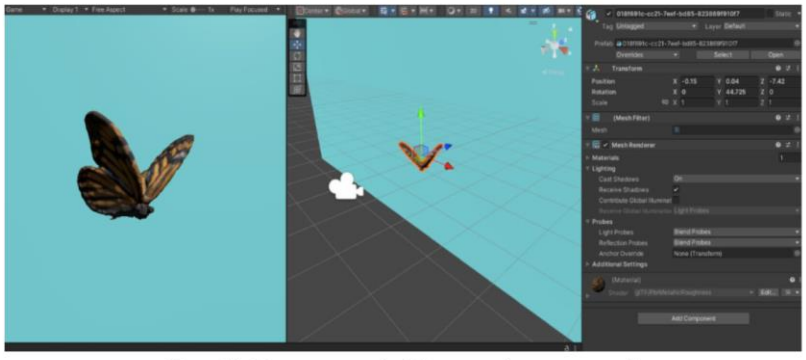

Рис. 9 Демонстрація всього необхідного набору 3Д-моделі

Рис. 10 Демонстрація 3Д-моделі в редакторі

Рисунок 5.12 Слайд 12

ВИСНОВКИ

11

Для традиційного створення 3Д-моделі в професійній команді потрібно 3 людини:

- дизайнер,який намалює арт моделі
- 3Д-художник, який зробить цю 3Д-модель
- програміст, який буде в подальшому з нею працювати

На розробку моделі може піти не один день. Розроблений мною модуль зменшує часові затрати до декількох хвилин. При розробці , наприклад, гри часто необхідно маленькі деталі на пошук яких виділяється дорогоцінний час, при цьому сама розробка гри з місця не зрушується . Згенеровані моделі можна використовувати одразу або віддати на допрацювання відповідним спеціалістам.

Рисунок 5.13