Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Адаптивна система керування роботизованим маніпулятором на основі мультимодального глибокого навчання
(тема)

Виконав:
здобувач другого року навчання,
групи СШІм-23-1
Іван Науменко
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва освітньої програми)
Керівник доц. Олександр Шевченко
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
(_____) _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Науменку Івану Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Адаптивна система керування роботизованим маніпулятором на основі мультимодального глибокого навчання
затверджена наказом університету від 21 квітня 20 25 р. № 295Ст
2. Термін подання студентом роботи до екзаменаційної комісії 4 червня 20 25 р.
3. Вихідні дані до роботи Функція: розробка адаптивної системи керування роботизованим маніпулятором на основі мультимодального глибокого навчання. Взаємодія: симуляційне середовище із завданнями у формі текстових команд. Використані засоби: Linux (Arch), MuJoCo, LeRobot, Gymnasium, PyTorch, HuggingFace Transformers, TensorBoard. Вхідні дані: демонстраційний датасет, зображення з камери, кастомне симуляційне середовище, вектори положення суглобів, текстові інструкції природною мовою. Вихідні дані: ваги моделі, відео виконання завдань, графіки навчальних метрик і оцінок.
4. Перелік питань, що потрібно опрацювати в роботі
1) Актуальність та постановка задачі
2) Проєктування та розробка симуляційного середовища
3) Створення датасету та навчання моделі керування
4) Експериментальне дослідження та оцінка ефективності

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	21.04.2025	виконано
2	Вивчення науково-технічної літератури	25.04.2025	виконано
3	Постановка мети, задач та актуальності	27.04.2025	виконано
4	Аналіз сучасних підходів до керування	30.04.2025	виконано
	роботизованими маніпуляторами на основі ШІ		
5	Дослідження можливих програмних середовищ	02.05.2025	виконано
	для симуляції та навчання роботів		
6	Вибір архітектури нейронної мережі	05.05.2025	виконано
7	Проєктування та реалізація симуляційного	09.05.2025	виконано
	середовища у MuJoCo, інтеграція в LeRobot		
8	Налагодження конфігурації середовища,	14.05.2025	виконано
	визначення ознак спостереження та дій		
9	Збір або генерація навчального датасету	16.05.2025	виконано
10	Проведення процесу навчання моделі та	19.05.2025	виконано
	виправлення помилок		
11	Аналіз метрик навчання, оцінка якості політики	20.05.2025	виконано
12	Експерименти та тестування чутливості до змін у	21.05.2025	виконано
	середовищі		
13	Оформлення звіту до роботи	24.05.2025	виконано
14	Захист перед ЕК	04.05.2025	

Дата видачі завдання 21 квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Олександр Шевченко
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 52 с., 7 рис., 1 дод., 13 джерел.

АВТОМАТИЗАЦІЯ, АДАПТИВНЕ КЕРУВАННЯ, МУЛЬТИМОДАЛЬНЕ НАВЧАННЯ, НАВЧАННЯ З ПІДКРІПЛЕННЯМ, ОБРОБКА ПРИРОДНОЇ МОВИ, РОБОТИЗОВАНИЙ МАНІПУЛЯТОР, РОБОТОТЕХНІКА, СИМУЛЯЦІЙНЕ МОДЕЛЮВАННЯ, СИСТЕМА КОНТРОЛЮ РУХІВ, ШТУЧНИЙ ІНТЕЛЕКТ У РОБОТОТЕХНІЦІ, MJUCO.

Об’єкт дослідження – процеси керування роботизованим маніпулятором з використанням технологій штучного інтелекту для виконання завдань у динамічному середовищі.

Предмет дослідження – методи та алгоритми адаптивного керування роботизованим маніпулятором на основі мультимодального глибокого навчання з використанням візуальної інформації, кінематичних даних та текстових інструкцій.

Мета роботи – дослідження та розробка системи адаптивного керування роботизованим маніпулятором, яка здатна інтерпретувати природномовні інструкції, аналізувати дані з камер та датчиків положення суглобів для автономного виконання поставлених завдань без необхідності експліцитному програмуванні послідовності дій.

Методи дослідження – системний аналіз, імітаційне моделювання робототехнічних систем, глибоке навчання з підкріпленням, комп'ютерний зір, обробка природної мови, мультимодальні нейронні мережі, експериментальна перевірка алгоритмів у симуляційному середовищі з використанням фреймворку MuJoCo.

ABSTRACT

Master’s thesis contains: 52 pp., 7 fig., 1 ann., 13 references.

ADAPTIVE CONTROL, ARTIFICIAL INTELLIGENCE IN ROBOTICS, AUTOMATION, MOTION CONTROL SYSTEM, MUJOCO, MULTIMODAL LEARNING, NATURAL LANGUAGE PROCESSING, REINFORCEMENT LEARNING, ROBOTIC MANIPULATOR, ROBOTICS, SIMULATED MODELING.

Object of research – processes of controlling a robotic manipulator using artificial intelligence technologies for performing tasks in a dynamic environment.

Subject of research – methods and algorithms for adaptive control of a robotic manipulator based on multimodal deep learning using visual information, kinematic data, and textual instructions.

Purpose of the work – research and development of an adaptive control system for a robotic manipulator that can interpret natural language instructions, analyze data from cameras and joint position sensors for autonomous execution of assigned tasks without the need for explicit programming of action sequences.

Research methods – system analysis, simulation modeling of robotic systems, deep reinforcement learning, computer vision, natural language processing, multimodal neural networks, experimental verification of algorithms in a simulation environment using the MuJoCo framework.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект;

AI – Artificial Intelligence – штучний інтелект;

CPU – Central Processing Unit – центральний процесор;

FPS – Frames Per Second – кількість кадрів за секунду;

GPU – Graphics Processing Unit – графічний процесор;

LeRobot – Learning Environment for Robots – фреймворк для навчання керування роботами;

MuJoCo – Multi-Joint dynamics with Contact – симулятор фізики з підтримкою контактної динаміки;

OBS – Observation – спостереження (вхід моделі або середовища);

pi0 / π_0 – Multimodal Policy Zero – мультимодальна політика керування (назва моделі);

RAM – Random Access Memory – оперативна пам’ять;

SigLIP – Sigmoid Loss Image-Text Pretraining – енкодер зображень на основі сигмоїдальної втрати;

Sim2Real – Simulation-to-Reality Transfer – перенесення політики з симуляції у реальне середовище;

Swap – Swap Memory – віртуальна пам’ять (область жорсткого диска, що імітує RAM);

VLM – Vision-Language Model – візуально-мовна модель;

XML – eXtensible Markup Language – розширювана мова розмітки.

ВСТУП

В умовах стрімкого розвитку технологій роботизовані системи набувають все більшого поширення у різноманітних сферах людської діяльності. Промислова робототехніка, зокрема роботи-маніпулятори, становлять фундаментальну компоненту сучасного виробництва, логістики, медицини та наукових досліджень. Проте традиційні методи програмування роботизованих систем залишаються трудомісткими, вимагають висококваліфікованих фахівців і демонструють обмежену гнучкість у непередбачуваних або динамічних середовищах.

Сучасні досягнення в галузі штучного інтелекту відкривають нові горизонти для побудови інтелектуальних систем керування, здатних інтерпретувати різноманітні типи вхідних даних та автономно приймати рішення без необхідності в експліцитному програмуванні. Особливо перспективним напрямом є мультимодальне глибоке навчання, яке дозволяє системам інтегрувати кілька типів вхідної інформації – зображення з камер, сенсорні дані, поточний стан системи та природномовні інструкції.

Актуальність теми зумовлена зростаючою потребою у створенні більш універсальних, інтуїтивно керованих та адаптивних систем керування роботизованими маніпуляторами. Такі системи мають бути здатними швидко перебудовувати поведінку під нові завдання без перепрограмування, взаємодіяти з користувачем через природні інтерфейси (мову, жести, візуальні сигнали), а також демонструвати стійкість до варіацій у середовищі.

У роботі запропоновано інноваційний підхід до розробки адаптивної системи керування роботизованим маніпулятором, побудованої на основі мультимодальної трансформерної моделі Pi0. Модель отримує на вхід три основні модальності:

– зображення з камери;

– поточні положення суглобів маніпулятора;

– текстові команди природною мовою, після чого генерує відповідні дії без потреби у ручному програмуванні. Такий підхід дозволяє досягнути високого рівня узагальнення, гнучкості та інтуїтивної взаємодії.

Реалізація системи здійснювалася на базі відкритого фреймворку LeRobot, у межах якого було створено власне симуляційне середовище з роботизованим маніпулятором, адаптоване для мультимодального навчання. Для тестування та валідації розробленої системи керування було створено спеціалізоване симуляційне середовище на базі фреймворку MuJoCo, яке моделює реальні фізичні взаємодії робота-маніпулятора з об'єктами. Використання віртуального середовища дозволяє проводити експерименти з різними конфігураціями робота та завданнями, забезпечуючи безпечне та ефективне навчання моделі.

Наукова новизна дослідження полягає в поєднанні методів комп'ютерного зору, обробки природної мови та глибокого навчання в єдину систему керування, яка забезпечує семантичне розуміння завдань і автономне виконання маніпуляційних дій. Практична значущість обумовлена можливістю застосування розробленої системи в галузях, що потребують інтуїтивної взаємодії людини з машиною, зокрема в промисловості, логістиці, медичній робототехніці, тощо.

У процесі роботи над дослідженням проаналізовано ключові виклики та обмеження запропонованого підходу, зокрема питання генералізації навчених навичок на нові типи об'єктів та завдань, а також можливості перенесення системи з симуляційного середовища на реальні робототехнічні платформи. Таким чином, дослідження суттєво розширює теоретичні та практичні основи адаптивного керування роботизованими системами шляхом впровадження інноваційних методів обробки природномовних інструкцій.

1 АКТУАЛЬНІСТЬ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Сучасний стан розвитку систем керування роботизованими маніпуляторами

Сфера інтелектуального керування маніпуляційними роботами є однією з найбільш динамічних і перспективних галузей сучасної робототехніки, що інтегрує досягнення штучного інтелекту, машинного навчання, комп'ютерного зору та теорії керування. Традиційні методи, що базувалися на класичних алгоритмах програмування та жорстко визначених послідовностях дій, не забезпечують належної адаптивності у сучасних динамічних системах, що стимулює розвиток нових архітектур із самонавчанням і гнучкою інтерпретацією інструкцій для різноманітних сфер автоматизації. Значний внесок у розвиток інтелектуальних систем керування маніпуляційними роботами зробили дослідницькі групи Massachusetts Institute of Technology, Stanford University та провідні лабораторії компаній Google DeepMind, Boston Dynamics та NVIDIA. Їхні розробки демонструють принципово нові підходи до навчання роботів, засновані на глибокому машинному навчанні, методах навчання з підкріпленням та використанні мультимодальних нейронних мереж.

Сучасні дослідження в галузі інтелектуального керування маніпуляційними роботами здебільшого зосереджуються на мультимодальному навчанні, яке забезпечує можливість обробки та інтеграції інформації з різнорідних джерел, зокрема сенсорних даних і текстових інструкцій.

У свою чергу, дослідження класифікуються за двома типами:

– навчання з підкріпленням (reinforcement learning);

– імітаційне навчання (imitation learning).

Основа кожного підходу базується на власній парадигмі взаємодії інтелектуального агента з навколишнім середовищем та доступними джерелами інформації. Імітаційне навчання стало популярним завдяки можливості використання демонстрацій, виконаних людиною або іншою системою, для формування стратегій поведінки робота. У свою чергу, навчання з підкріпленням дозволяє системам оптимізувати свої дії через взаємодію з середовищем, що особливо корисно в задачах, де точні демонстрації важко отримати. Однак, як зазначається в літературі, навчання з підкріпленням часто стикається з проблемою високої обчислювальної складності та необхідності великої кількості ітерацій для досягнення стабільних результатів.

1.2 Аналіз існуючих мультимодальних систем керування роботами

Сучасний етап розвитку технологій та штучного інтелекту дозволяють маніпуляційним роботам зробити активний перехід від класичних програмованих рішень до гнучких, навчальних і мультимодальних систем, що поєднують глибоке навчання, комп’ютерний зір, обробку природної мови та сенсорну інформацію.

Провідні технологічні компанії та науково-дослідні центри почали пропонувати свої інноваційні рішення, які демонструють вагомі досягнення у цій галузі, пропонуючи інноваційні моделі, що поєднують різні модальності даних для підвищення автономності, гнучкості та адаптивності роботизованих систем.

Одним із головних гравців є компанія Google DeepMind, розробвши серію моделей RT (Robotics Transformer). Архітектура RT-1, представлена у 2022 році, поєднує трансформерний підхід із зоровими даними та природномовними інструкціями для керування роботами. В її основі лежить попередньо навчена модель обробки зображень EfficientNet-B3, яка

використовує FiLM-шари (Feature-wise Linear Modulation), що дає змогу адаптувати зорову інформацію відповідно до контексту завдання.

Експериментальні результати демонструють, що дана модель досягає 97% успішності на наборі з понад 700 навчених інструкцій, при цьому показуючи 76% успішності на абсолютно нових завданнях, 83% стійкості до відволікаючих об'єктів і 59% стійкості до змін фону.

Продовження серії моделей RT від Google DeepMind отримало своє втілення у RT-2 (Robotics Transformer 2), яка була представлена у 2023 році. Ця модель значно розширює можливості свого попередника за рахунок інтеграції масштабних візійно-мовних моделей (VLM), таких як PaLI-X та PaLM-E, які навчалися на зображеннях і текстових даних з інтернету. Головним новаторством RT-2 є об'єднання дієвого та лінгвістичного просторів, де роботизовані команди, такі як: координати переміщення, стан захоплювача, трансформуються у текстові токени. Це дозволяє системі опрацьовувати їх аналогічно до природномовних інструкцій. Така архітектура забезпечує моделі RT-2 не лише реалізацією базових операцій, а й успадкуванням семантичного потенціалу візуально-мовних моделей: розпізнавання символіки та елементарних логічних конструкцій. Це і стало ключовою перевагою даної моделі, набувши емерджентні можливості, які виникають через перенесення знань з інтернет-даних. На відміну від першої моделі (RT-1), яка демонструвала високу ефективність тільки у межах навчених інструкцій, друга версія успішно виконує завдання, які не були представлені в роботизованих даних. Наприклад, розміщення об’єктів біля чисел або іконок, а також вибір предметів на основі їхніх атрибутів, таких як (найменший предмет) або (найближчий предмет). Однак модель має обмеження, пов’язані з фізичною реалізацією дій: вона не здатна виконувати рухи та складні послідовності, відсутні в її тренувальних даних, а її продуктивність значною мірою залежить від обчислювальних ресурсів.

Одним із ранніх проривів у інтеграції візуального сприйняття з природномовними інструкціями для роботизованої маніпуляції стала модель CLIPORT, представлена у 2021 році. Її архітектура об’єднує два ключові елементи:

– просторову точність Transporter Networks;

– семантичне узагальнення моделі CLIP, попередньо навчену на великому корпусі зображень і текстів з Інтернету.

CLIPORT працює у форматі (pick-and-place), передбачаючи позиції для захоплення та переміщення об’єктів. Завдяки двопотоковій архітектурі модель здатна точно виконувати інструкції, задані природною мовою, навіть якщо об'єкти чи їхні атрибути не зустрічалися під час навчання.

Однією з переваг CLIPORT є її здатність до мультизадачного навчання: одна й та сама модель може вирішувати до 10 різних завдань, демонструючи продуктивність, аналогічну або вищу за спеціалізовані однозадачні політики. Під час випробувань у симуляційному середовищі та на реальному маніпуляторі Franka Panda система продемонструвала високу ефективність навіть у режимі few-shot навчання. Проте її застосування обмежується 2D-задачами на стільниці, а також простими рухами без складної кінематики чи багатопальцевого захоплення, що унеможливлює її пряме використання для маніпуляцій у складніших сценаріях.

1.3 Постановка задачі дослідження

Розробка роботизованого маніпулятора, здатного безпосередньо виконувати команди природною мовою на основі сприйняття візуального середовища, представляє міждисциплінарний науковий виклик, що потребує синтезу методів комп'ютерного зору, обробки природної мови та керування роботами в єдиній когнітивній архітектурі. Основною науково-прикладною проблемою є побудова ефективної нейронної моделі, що

поєднує обробку візуальної інформації, внутрішніх сенсорних параметрів системи та текстових команд користувача, і на основі цього формує узгоджену поведінку у вигляді послідовності моторних дій в умовах симуляційного або реального середовища. Особливість такої системи полягає в необхідності інтеграції гетерогенних типів даних, що традиційно оброблялися окремими підсистемами, у єдиному когерентному навчальному підході, який забезпечує інтерпретовану реакцію агента на високорівневі інструкції.

Реалізація такої системи потребує створення повноцінного програмного контуру, який включає модуль спостереження, обчислювальну модель політики, інтерфейс введення команд природною мовою, симулятор фізики та систему оцінки результатів виконання. Першочерговою задачею є аналіз сучасних програмних та апаратних засобів, що використовуються для моделювання робототехнічних систем, з акцентом на підтримку фізично достовірної динаміки, мультикамерного візуального вводу та підтримки користувацьких моделей. У цьому напрямку важливим є порівняння таких симуляційних середовищ, як MuJoCo, PyBullet, Webots, Isaac Sim, Genesis та інших, з урахуванням можливості інтеграції з фреймворками глибокого навчання і забезпечення детального керування взаємодією між роботом та об’єктами сцени.

Важливою складовою дослідження є вибір архітектури нейронної мережі, яка зможе об’єднати зображення з камер, пропріоцептивну інформацію про стан маніпулятора та текстові команди. У цьому аспекті необхідно здійснити порівняльний аналіз сучасних мультимодальних трансформерних моделей, що продемонстрували успішне поєднання мовних і зорових сигналів з моторною відповіддю у задачах маніпуляції. Вибрана модель повинна бути адаптована до специфіки створеного середовища: налаштована на відповідні формати вхідних та вихідних даних, параметризована згідно з конфігурацією робота та оптимізована для

навчання в обмежених обчислювальних умовах. Особливу увагу слід приділити узгодженню імен вхідних ознак, коректному визначенню їх розмірностей та забезпеченню стабільності моделі в процесі тренування. Крім того, необхідно враховувати обмеження, пов’язані з апаратним забезпеченням.

Окремий напрям задачі пов’язаний із побудовою власного симуляційного середовища, у якому змодельовано маніпулятор, камеру, об’єкт взаємодії та логіку постановки завдань. Таке середовище має бути сумісним з фреймворками для навчання політик, дозволяти програмну генерацію демонстрацій, підтримувати інтеграцію з інтерфейсом введення текстових інструкцій і збір метрик у процесі тренування. Необхідною умовою є наявність візуальної складової сцени, джерела стану робота та функціоналу для запису відео, що забезпечує подальший аналіз результатів. Особливу увагу необхідно приділити налагодженню конфігурацій взаємодії між середовищем та навчальною політикою, що вимагає точного узгодження форматів даних, назв ознак та способів їх передачі.

Підсумовуючи, завдання даного дослідження полягає у побудові функціонального експериментального контуру, що включає симуляційне середовище, адаптовану мультимодальну нейромережеву модель, систему навчання та інструменти моніторингу ефективності, з метою перевірки гіпотези про можливість створення універсального інтерфейсу керування роботизованими маніпуляторами на основі природної мови. Така система має дозволити уникнути жорсткого програмування окремих дій та забезпечити базовий рівень адаптивності до зміни умов середовища. Успішне вирішення поставлених задач створить підґрунтя для подальших досліджень у напрямі генералізації поведінки, багатоетапних маніпуляційних дій та розширення систем на фізичні робототехнічні платформи.

2 ПРОЄКТУВАННЯ ТА РОЗРОБКА СИМУЛЯЦІЙНОГО СЕРЕДОВИЩА

2.1 Вибір програмного забезпечення та фреймворків

Проєктування та розробка ефективного симуляційного середовища для навчання адаптивної системи керування роботизованим маніпулятором потребує ретельного аналізу та вибору оптимального програмного забезпечення та фреймворків. Вибір інструментарію визначає точність фізичного моделювання, швидкість симуляції, можливості візуалізації, гнучкість інтеграції з алгоритмами машинного навчання та перспективи подальшого переносу розроблених рішень на реальні робототехнічні системи.

Для реалізації даного проєкту проведено комплексний аналіз доступних на сьогодні програмних рішень фізичних рушіїв для моделювання роботизованих систем. Основні критерії оцінювання включали:

– точність фізичної симуляції;

– стабільність контактної взаємодії;

– обчислювальна ефективність;

– наявність готових моделей роботів-маніпуляторів;

– інтегрованість з фреймворками машинного навчання;

– підтримка паралельних симуляцій для прискорення навчання;

– документованість та активність спільноти розробників.

Згідно з результатами аналізу, в якості основного фізичного симулятора було обрано MuJoCo (Multi-Joint dynamics with Contact), спроєктований спеціально для робототехніки та дослідження у галузі штучного інтелекту. MuJoCo демонструє одну з найкращих продуктивність

у моделюванні динаміки твердих тіл та обробці контактних взаємодій, що критично важливо для задач маніпуляції. Ключовими перевагами MuJoCo є:

– розширена мова опису моделей на основі XML, що дозволяє детально специфікувати кінематичну структуру робота, властивості матеріалів, параметри актуаторів та сенсорів;

– інтегрована система візуалізації на основі OpenGL, що забезпечує можливості як для відображення процесу симуляції, так і для генерації візуальних даних з різних ракурсів;

– підтримка доступу до внутрішніх станів системи, зокрема положень суглобів, векторів сил і моментів;

– генерацію зображень з віртуальних камер, що дозволяє повністю реалізувати мультимодальний вхід до нейромережі.

Для побудови власного середовища в MuJoCo було використано інфраструктуру проєкту LeRobot, що надає зручну обгортку над MuJoCo для створення користувацьких середовищ. LeRobot дозволяє швидко конфігурувати робота-маніпулятор, додавати нові джерела сенсорних даних, змінювати структуру середовища, а також інтегрувати моделі нейронних мереж, зокрема мультимодальні трансформери. Інтеграція з бібліотеками PyTorch та HuggingFace Transformers забезпечує ефективне з’єднання симуляційного ядра з сучасними архітектурами глибокого навчання.

Таким чином, вибраний програмно-апаратний стек – MuJoCo + LeRobot + PyTorch – забезпечує баланс між точністю фізичної симуляції, зручністю розробки, гнучкістю моделювання середовища та повною підтримкою мультимодальних даних, що є критичним для навчання та тестування нейронних моделей, орієнтованих на керування роботизованими маніпуляторами в умовах часткової невизначеності.

2.2 Створення роботизованого маніпулятора та опис симуляційного середовища

2.2.1 Модель роботизованого маніпулятора та його кінематичні характеристики

Попри широкі можливості, які надає MuJoCo для гнучкої побудови кастомізованих робототехнічних систем, у зв’язку з обмеженими часовими ресурсами для цілей даного дослідження було обрано вже наявну модель маніпулятора (Interbotix VX300S) із офіційної бібліотеки роботів MuJoCo, яка за своїми кінематичними характеристиками та функціональною конфігурацією найбільш відповідала вимогам поставленого експерименту. Ілюстрація зовнішнього вигляду маніпулятора наведена на рисунку 2.1.

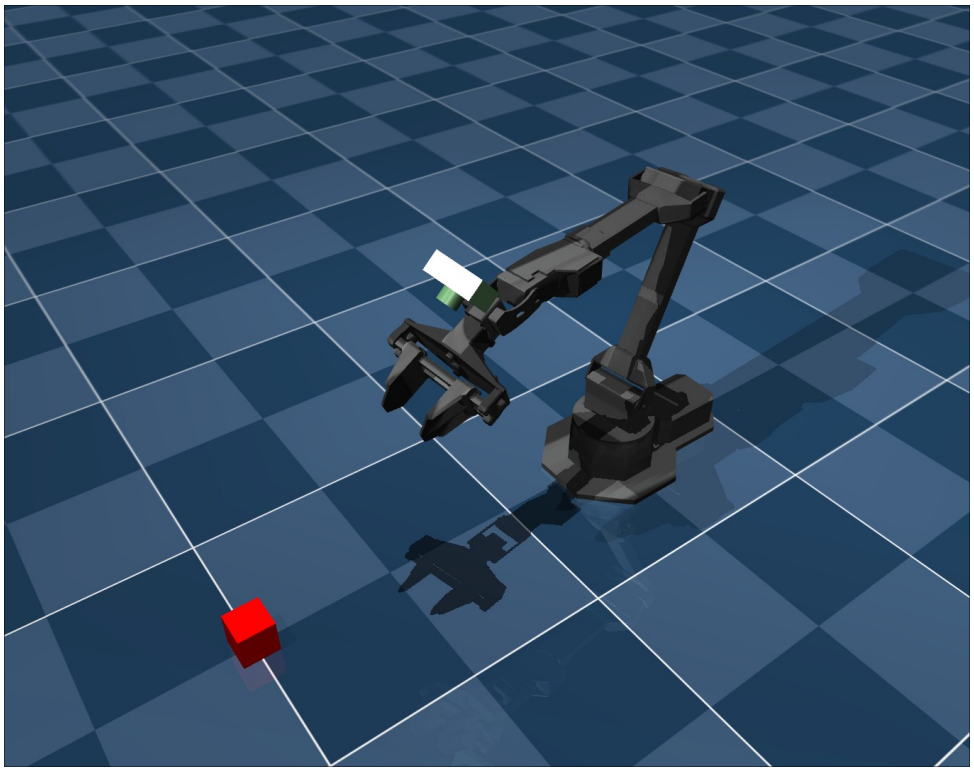

Рисунок 2.1 – Зовнішній вигляд маніпулятора

Дана модель була адаптована та доповнена для цілей мультимодального навчання і керування в умовах, наближених до реального середовища. Зокрема, було модифіковано конфігурацію симуляційного простору, встановлено додаткову камеру на кінці ефектора, що дозволяє отримувати візуальну інформацію безпосередньо з точки контакту з об’єктами, а також з метою реалізації навчальних сценаріїв було додано вільно рухомий об’єкт типу куб, розташований на платформі в центрі робочої зони маніпулятора. Для даного предмета дослідження прописані фізичні характеристики, такі як: маса, розмір, коефіцієнти тертя, які відповідають реалістичним параметрам для забезпечення коректного контакту під час захоплення. Візуалізація наведених змін представлена на рисунку 2.2.

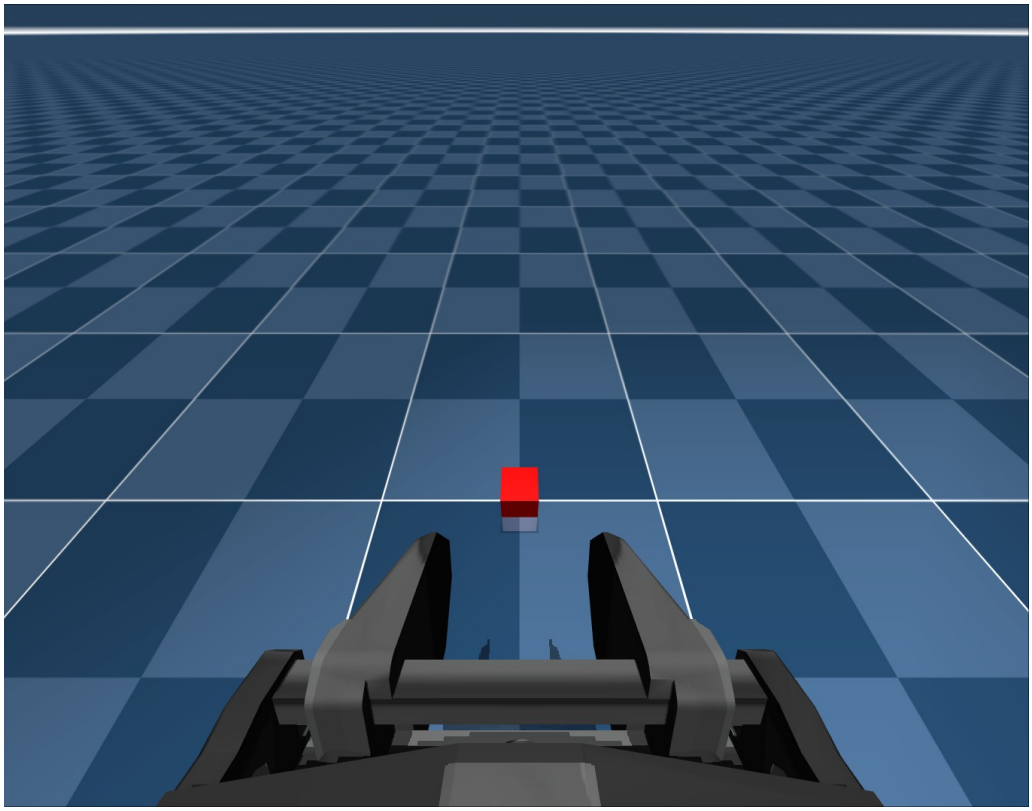

Рисунок 2.2 – Отримане зображення після додавання камери

Маніпулятор має шість ступенів свободи, що утворюють послідовний кінематичний ланцюг та забезпечують просторову рухливість кінцевого ефектора. Така конфігурація дозволяє виконувати широке коло маніпуляційних завдань, включаючи позиціювання, орієнтацію та захоплення об’єктів у тривимірному середовищі. Крім того, конструкція оснащена незалежним виконавчим механізмом для керування двопальцевим захватом паралельного типу, який реалізовано через пару лінійних приводів із симетричною механічною залежністю. Завдяки такій кінематичній схемі, маніпулятор забезпечує повноцінний доступ до тривимірного простору, дозволяючи виконувати широкий спектр задач захоплення та переміщення об’єктів з різною орієнтацією.

Таким чином, розроблена модель роботизованого маніпулятора вирізняється не лише високим рівнем фізичної достовірності, що досягається завдяки точному відтворенню кінематичних та динамічних характеристик у симуляційному середовищі MuJoCo, а й цілеспрямованою структурною адаптацією до вимог мультимодального навчання. Зокрема, архітектура моделі була спеціально оптимізована для обробки та інтеграції різнорідної сенсорної інформації, що включає візуальні сигнали з камери, пропріоцептивні дані про положення та зусилля в суглобах, а також природномовні інструкції, які задають мету або послідовність дій.

Інтегративна конфігурація забезпечує комплексну основу для проведення експериментів з навчання моделей, здатних до адаптивного керування в умовах часткової спостережуваності та високої варіативності вхідних даних. Такий підхід дозволяє здійснювати не лише тестування моделей у контрольованих сценаріях, а й проводити дослідження щодо їх здатності до генералізації – тобто переносу набутих стратегій поведінки на нові, раніше не представлені в навчальному датасеті, задачі та умови.

Зокрема, дослідницький інтерес становить вивчення ефективності обробки модальностей у різних комбінаціях та визначення ролі кожної з них

у забезпеченні стабільного виконання завдань маніпуляції в середовищі з розширеним сенсорним контекстом. У перспективі це відкриває можливості для побудови більш універсальних систем керування, здатних ефективно функціонувати у складних, динамічних та слабкоструктурованих середовищах.

2.2.2 Розробка симуляційного середовища керування роботизованим маніпулятором

Для забезпечення повноцінної інтеграції між фізичною моделлю роботизованого маніпулятора, реалізованою в середовищі MuJoCo, мультимодальною нейронною мережею та функціональним фреймворком LeRobot, необхідно створити спеціалізоване віртуальне оточення, здатне об’єднати ці дві складові в єдину навчальну систему. Таке середовище повинно виконувати роль посередника між симуляцією фізики, структурою сенсорних спостережень, мультимодальними інтерфейсами та інтерпретацією дій, які генерує модель. Як базовий інтерфейс для побудови подібного оточення використано бібліотеку Gymnasium – сучасну платформу для створення стандартних середовищ у стилі OpenAI Gym, яка забезпечує гнучку модульність, сумісність із MuJoCo та підтримку багатомодальних конфігурацій.

Розробка власного симуляційного середовища, яке б одночасно відповідало вимогам мультимодальної нейронної мережі та було б сумісним із тренувальним фреймворком LeRobot, виявилась нетривіальним завданням, що потребувало глибокого розуміння внутрішньої архітектури проєкту, оскільки на момент дослідженян та розробки була відсутня офіційна документація або чіткі інструкції щодо створення нового середовища у LeRobot. Для побудови працездатного коду було проаналізовано приклади вже інтегрованих середовищ – таких як aloha,

pusht, xarm – що входять до складу проєкту, кожне з яких було реалізоване для власного типу робота і мало індивідуальні особливості конфігурації. Озброївшись цими прикладами та великою кількістю експериментів, з використанням підказкazок та за допомогою великих мовних моделей, було поступово створено власне середовище (MyRobotEnv), адаптоване до специфіки модифікованого маніпулятора VX300S та побудоване на основі бібліотеки (gymnasium), що виступає базовим інтерфейсом між симуляцією MuJoCo та зовнішньою системою керування.

Розроблене симуляційне середовище реалізоване у вигляді класу (MyRobotEnv), який успадковує функціонал від (MujocoEnv) з бібліотеки Gymnasium, що забезпечує інтеграцію з фреймворком (MuJoCo) для фізичної симуляції, та від (gym.utils.EzPickle) для підтримки серіалізації. Структура коду охоплює всі ключові етапи життєвого циклу епізоду:

– ініціалізацію;

– скидання (reset);

– побудову спостереження (_get_obs);

– рендеринг візуальної інформації (_render_camera, _get_camera_images);

– виконання дії (step);

– обчислення функції винагороди (_compute_reward).

У методі (__init__) відбувається конфігурація середовища: завантажується XML-модель робота, задається формат вихідних даних, список активних камер, тип винагороди та розміри зображення, після чого викликається конструктор базового середовища з передачею ключових параметрів.

Метод (reset_model) забезпечує відновлення робота у фіксовану початкову конфігурацію, яка визначена в XML-файлі через ключову позу (home); при цьому синхронізується положення маніпулятора та

об’єкта – кубика, який розташовується в одній з наперед визначених позицій на площині.

У методі (_get_obs) відбувається формування спостережень, в якому зчитуються значення положення всіх елементів середовища, вибираються останні вісім компонентів, щоб відфільтрувати тільки позицію суглобів робота, що згодом утворюють вектор (agent_pos), який і утворює вхідний вектор для моделі.

Отримання зображення з камери реалізований у ментоді (_get_camera_images). У ньому також відбувається обробка можливих винятків відсутності камер, і генерує словник зображень, приведених до формату (uint8) з розміром 224×224×3.

Метод (step) реалізує застосування дії, оновлення стану симуляції та обчислення винагороди.

Оцінка винагороди у (_compute_reward) включає евристичні компоненти: відстань до об’єкта, стан захоплення та висоту підйому, причому у режимі (dense) використовується згладжена функція з ваговими коефіцієнтами для точнішої градації проміжних станів.

Додатково, success_steps фіксує кількість послідовних кроків, коли виконуються умови досягнення мети – це дозволяє мінімізувати випадкові флуктуації та надає стійку метрику успіху епізоду.

У процесі реалізації виникала велика кількість помилок, пов’язаних передусім із невідповідністю назв та розмірностей параметрів, які передаються від середовища до моделі через інфраструктуру LeRobot. Зокрема, значні труднощі викликали розбіжності між назвами полів, що використовуються в середовищі та назвами, які очікуються політикою. Ще одну складність у процесі розробки становило визначення саме структури вхідних і вихідних параметрів, які передаються від середовища до моделі. Питання полягало не лише у визначенні їх розмірності, які формують стан агента, а й у тому, які саме компоненти з доступних у MuJoCo значень, таких

як qpos, qvel, позиції тіл, сенсори тощо, доцільно включити у вхідний вектор.

Ще однією важливою дією для забезпечення повної сумісності між новоствореним середовищем та тренувальною інфраструктурою LeRobot було необхідно внести цілеспрямовані зміни до конфігураційної підсистеми проєкту. Зокрема, ключовим кроком стало створення окремого конфігураційного класу (MyRobotEnv), який було зареєстровано як підклас базової абстракції (EnvConfig). Цей клас визначає унікальні параметри середовища:

– частоту оновлення (fps);

– шлях до XML-моделі робота;

– список камер;

– розміри зображень;

– тривалість епізоду;

– структуру вхідних і вихідних ознак (features) разом із мапінгом їхніх назв (features_map) на внутрішні ідентифікатори LeRobot.

Саме на цьому етапі виникли складнощі, пов’язані з точним визначенням імен параметрів та їхніх розмірностей, оскільки як мультимодальна модель, так і середовище LeRobot вимагають жорсткої відповідності між ключами словника спостережень та їхньою типізацією. У випадку будь-якої невідповідності – наприклад, якщо розмірність вектора стану агента (agent_pos) не збігалася з очікуваною моделлю або використовувався неправильний ключ, наприклад observation.image замість pixels – навчальний цикл завершувався з помилкою.

Ці виклики ускладнювалися також тим, що фреймворк LeRobot не містить ані офіційної документації для створення власного середовища, ані механізмів автоматичної валідації відповідності між env_config і policy_config, тому всі такі невідповідності доводилося виявляти вручну – шляхом аналізу трасувальних повідомлень, глибокого дебагінгу, а також

поетапної реконфігурації features і features_map. У багатьох випадках точне значення розмірностей не випливало з контексту, а визначалося експериментально – через аналіз стеку викликів і перевірку на етапах підключення спостережень до моделі.

Складність посилювалася ще й тим, що в окремих прикладах середовищ, які постачаються з LeRobot, спостерігалася неоднорідність у використанні назв ключів – зокрема, одні конфігурації використовували pixels, інші – observation.image, або ж agent_pos протиставлявся observation.state. Це вимагало не лише технічної перевірки, а й контекстного розуміння, які саме поля насправді використовуються у політиці й у коді завантаження даних. У підсумку, лише після ретельної синхронізації назв, структур, розмірностей та відлагодження методів вдалося забезпечити повну функціональну сумісність між середовищем, конфігураційним класом та нейронною мережею, що створило основу для подальшого етапу навчання.

3 СТВОРЕННЯ ДАТАСЕТУ ТА НАВЧАННЯ МОДЕЛІ КЕРУВАННЯ

3.1 Теоретичні основи мультимодального глибокого навчання

Мультимодальне глибоке навчання є напрямом штучного інтелекту, що досліджує способи одночасної обробки та інтеграції інформації з різних джерел або модальностей, таких як зображення, текст, аудіо, відео, сенсорні або пропріоцептивні дані. В основі такого підходу лежить ідея про те, що багатомодальне представлення дозволяє моделі формувати більш узагальнене уявлення про середовище, дію чи завдання, уникаючи обмежень, притаманних одній модальності. Наприклад, зображення дають просторовий контекст, текст описує інструкцію, а вектор стану робота забезпечує динамічну інформацію про його положення – поєднання цих сигналів дозволяє досягти більш природної, адаптивної поведінки системи.

Існує кілька принципових підходів до інтеграції модальностей у межах глибокого навчання. Найпростішим є раннє об’єднання (early fusion), коли всі модальності конкатенуються або проєктуються в спільний простір ще до основного обчислення. Більш поширеним є пізнє об’єднання (late fusion), коли кожен канал обробляється окремим енкодером, а потім результати поєднуються для прийняття рішення. Сучасні мультимодальні архітектури, зокрема CLIP, Flamingo, Perceiver та інші, застосовують більш складні стратегії, такі як крос-модальне увагове поєднання (cross-attention), яке дозволяє одній модальності фокусуватися на релевантних фрагментах іншої, формуючи глибші семантичні зв’язки між даними. Наприклад, у CLIP зображення та текст обробляються незалежними енкодерами, але через спільний простір векторних подань відбувається їх узгодження на рівні задачі.

Особливе місце в розвитку мультимодальних моделей посідають трансформерні архітектури, що завдяки механізму уваги (attention)

демонструють здатність гнучко інтегрувати неоднорідну інформацію. Цей підхід активно використовується і в сучасній робототехніці, де виникає потреба одночасно враховувати візуальний стан сцени, моторні сигнали та вербальні інструкції. Такі моделі здатні не лише прогнозувати дії, але й адаптуватися до нових завдань, демонструючи загальні риси так званих політик універсального призначення (generalist policies).

3.2 Вибір та структура нейронної мережі

Формування нейромережевої компоненти системи вимагало особливо ретельного підходу, оскільки саме ця частина відповідає за семантичну інтерпретацію завдань та прийняття рішень на основі мультимодального входу. У пошуку моделі, яка б поєднувала здатність до генералізації з ефективністю обробки реальних сенсорних даних, було розглянуто низку сучасних архітектур. У результаті порівняльного аналізу вибір зупинився на моделі pi0 – новітньому підході до побудови мультимодальних політик, який пропонує узагальнене уявлення про дії на основі візуальних, текстових та пропріоцептивних даних.

Модель π_0 (pi-zero, pi0) була розроблена дослідницькою командою компанії Physical Intelligence у 2024 році як універсальний фреймворк для управління роботами на основі алгоритмів штучного інтелекту. Авторський колектив сформували відомі дослідники в галузі машинного навчання і робототехніки, які мають значний досвід у розробці великих мультимодальних моделей. Основною метою цього проєкту було створення масштабованої архітектури, здатної інтегрувати візуальні, мовні та моторні сигнали для виконання складних маніпуляційних завдань у реальному світі.

Архітектура моделі включає низку компонентів, що сприяють досягненню високої продуктивності. У якості семантичного ядра модель π_0 використовує передтреновану візуально-мовну модель PaliGemma, яка

являє собою складну гібридну архітектуру, що інтегрує візуальний енкодер SigLip та мовну модель Gemma.

Структура взаємодії основних функціональних компонентів мультимодальної моделі керування π_0, а також поетапний процес обробки вхідних даних і генерації моторних команд, схематично представлено на рисунку 3.1. Ця схема виконує роль візуального узагальнення архітектури системи, ілюструючи весь ланцюг трансформації вхідної інформації до кінцевого результату – дії маніпулятора. Згідно з представленою структурою, модель приймає на вхід три типи даних: візуальні спостереження (у вигляді RGB-зображень із камер, закріплених на або навколо робота), природномовну інструкцію (що визначає поставлену задачу) та пропріоцептивну інформацію у вигляді векторного представлення поточного стану робота (наприклад, положення суглобів, швидкості, сили тощо).

Усі ці входи попередньо нормалізуються та подаються до візуально-мовної моделі (VLM) – заздалегідь натренованої нейронної архітектури, яка виконує функцію спільного кодування мультимодальної інформації. VLM інтегрує зображення, текстову інструкцію та вектор стану у єдиний латентний простір, створюючи спільне векторне представлення поточного контексту завдання.

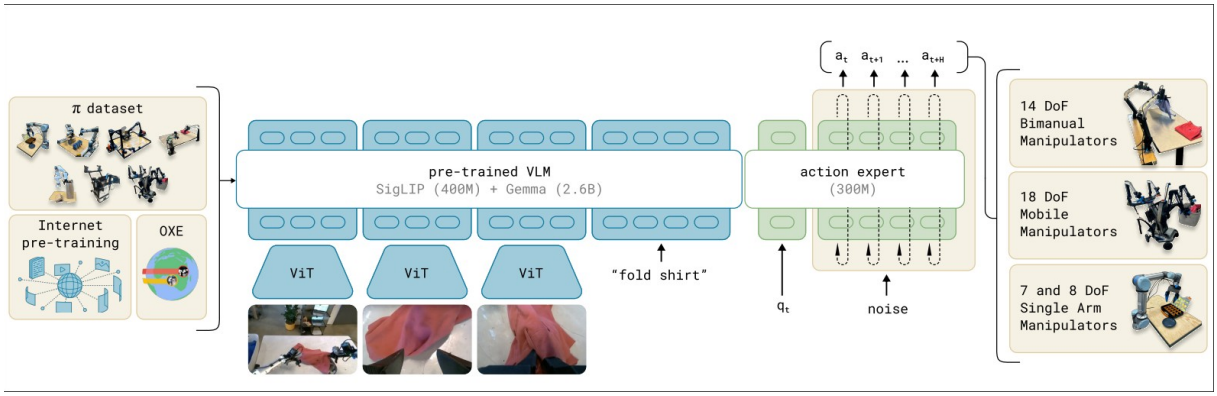

Рисунок 3.1 – Архітектура нейронної мережі π_0

SigLip (Sigmoid Loss for Language-Image Pre-training) є спеціалізованим візуальним трансформером об'ємом 400 мільйонів параметрів, оптимізованим для ефективного спільного навчання з текстовими представленнями. Особливістю даного трансформера є використання модифікованої функції втрат на основі сигмоїди, яка значно покращує здатність моделі до узагальнення в умовах обмежених даних.

Мовна складова PaliGemma побудована на базі архітектури Gemma 2B – оптимізованої версії великої мовної моделі від Google, що містить 2.6 мільярда параметрів.

Також до базової VLM-архітектури додано спеціалізований (action expert) об'ємом 300 мільйонів параметрів, який відповідає за генерацію послідовностей дій для робота. На відміну від класичних autoregressive або токенізованих підходів до генерації дій, цей модуль використовує інноваційний підхід потокового узгодження (flow matching) – варіант дифузійного навчання, який дозволяє моделі ефективно апроксимувати розподіли безперервних дій у часі. Дія на кожному кроці моделюється як частина action chunk, що складається з послідовності дій для робота, розрахованої на фіксований інтервал у 50 кроків наперед, які згодом об'єднуються у векторну послідовність.

Модель було обрано для реалізації проєкту завдяки її сучасним архітектурним рішенням, що поєднують інноваційні підходи з високою ефективністю та практичною доступністю. Її архітектура дозволяє досягати високої продуктивності при помірних обчислювальних витратах, що робить її кращим вибором серед інших варіантів.

3.3 Формування навчального датасету для симульованого роботизованого середовища

У даному дослідженні як основу було використано проєкт lerobot, що

зумовило необхідність створення датасету у структурованому форматі LerobotDataset, який забезпечує оптимальну сумісність з архітектурою проєкту та коректну інтеграцію з навчальними моделями.

LerobotDataset вирізняється чіткою організацією навчальних даних та їх узгодженістю з вимогами модельної архітектури. Цей формат передбачає зберігання часових рядів, спостережень, виконаних дій та відповідних нагород, організованих у послідовні епізоди функціонування робота у симуляційному середовищі. Кожен запис в отриманому датасеті містить наступні обов'язкові компоненти:

– стан спостереження (observation.state), що включає поточні позиції суглобів робота;

– вектор дії (action), який представляє керуючі сигнали для суглобів та грипера;

– часову мітку (timestamp), що фіксує момент виконання дії;

– індекси епізоду та кадру для організації послідовностей;

– інформацію про нагороду (next.reward) для оцінки успішності дії;

– прапорець завершення епізоду (next.done).

Через внесені зміни до структури середовища робота відсутні готові дані для даної моделі в загальнодоступних джерелах, тому було прийнято рішення створити власний невеликий датасет для навчання. З метою реалізації було написано код на мові Python, який взаємодіє з середовищем та роботом через бібліотеку Gymnasium. Він реалізує клас (DataCollector), який відповідає за ініціалізацію середовища, генерацію дій робота, збір даних та їх збереження у відповідному форматі. Структура зберігання даних передбачає організацію файлів у кілька категорій: фактичні дані станів та дій у форматі Parquet, відеозаписи з камер робота у форматі MP4, а також метадані у формати JSON, що описують структуру та статистичні характеристики датасету.

Для запису даних було реалізовано програмні інструкції руху робота, які відтворюють послідовності дій для підняття кубика на певну висоту, що є пріоритетним завданням на поточному етапі. Функціональна реалізація настанов розділена на декілька фаз руху:

– наближення до об'єкта;

– захоплення кубика;

– підняття його на задану висоту;

– утримання у стабільному положенні.

Паралельно з фіксацією даних про стан та дії робота здійснювався запис відеоданих з камери, встановленої на грипері робота. Ці відеодані зберігаються у структурованому вигляді та синхронізуються з іншими даними за допомогою часових міток. Такий підхід дозволяє моделі навчатися зіставляти візуальну інформацію з відповідними механічними діями робота.

У зв'язку з часовими обмеженнями дослідження та з метою оптимізації процесу навчання моделі було прийнято рішення зосередитися на записі всього однієї конкретної дії – підняття кубика на визначену висоту. Такий вибір не був випадковим, а зумовлений практичними обмеженнями – насамперед браком часу на формування повноцінного, різноманітного датасету, а також обмеженнями доступних обчислювальних ресурсів, що не дозволяли ефективно тренувати модель на великій кількості прикладів. Основною метою дослідження було не навчання універсального агента, а перевірка працездатності мультимодальної моделі π_0 у контексті взаємодії з кастомним середовищем та з’ясування потенціалу фреймворку LeRobot як основи для подальших інтерактивних експериментів.

3.4 Методи та алгоритми навчання адаптивної системи керування

У межах проєкту Lerobot вже реалізовано необхідні програмні засоби

для навчання моделей, зокрема й архітектури pi0. Проте, оскільки як конфігурація робота, так і симуляційне середовище були розроблені самостійно, без відповідної документації та офіційних інструкцій, виникла потреба у попередній багатоступеневій підготовці. Це включало адаптацію середовища, забезпечення його сумісності з наявними скриптами, а також їх модифікацію. Процес навчання було умовно розподілено на два основних етапи:

– вирішення проблем сумісності та налаштування конфігурації моделі;

– навчання моделі з поступовим усуненням виявлених недоліків.

3.4.1 Вирішення проблем сумісності та налаштування конфігурації моделі

Наступним за складністю та часовими витратами завданням було забезпечення повної сумісності між структурою вхідних і вихідних даних симуляційного середовища та очікуваним форматом моделі. Під час інтеграції моделі Pi0 із власноруч розробленим середовищем (myrobot_env) було зафіксовано низку типових помилок, пов’язаних із невідповідністю розмірностей тензорів, неправильним форматуванням каналів зображення, а також розбіжністю між фактичними та очікуваними структурами словників спостережень. Усі ці проблеми вимагали системного аналізу та тонкого налаштування як конфігурації середовища, так і параметрів самої моделі.

У розробленій специфікації (MyRobotEnv), яка реалізує реєстрації та специфікацію для мого середовища та моделі в проєкті Lerobot, прописані ключові параметри для визначення розміру зображення (pixels), кількість вхідних ознак (agent_pos) та розмірність простору дій (action). Зазначені параметри визначаються технічними характеристиками розробленого робота і повинні точно відповідати конфігурації, з якою ініціалізується

політика Pi0. Однак особливе ускладнення на етапі інтеграції спричинили розбіжності у назвах ключів для мультимодальних ознак між різними компонентами проєкту. Оскільки LeRobot не має централізованої або формалізованої документації, більшість рішень щодо правильного формату конфігураційних параметрів ґрунтується на фрагментарних поясненнях у коді, коментарях розробників, прикладах у тестових скриптах або обговореннях у спільноті. Наприклад, у деяких фрагментах обговорень згадувалося, що для зображення варто використовувати ключ (observation.image), однак на практиці модель π_0 коректно обробляє лише поле з назвою (pixels), і будь-яка відмінність у назві призводила до помилок при зчитуванні або ініціалізації вхідних конфігурацій. Майже аналогічна проблема виникала з передачею пропріоцептивного стану робота: у конфігурації середовища це поле називалося (agent_pos), тоді як у конфігурації моделі π_0 воно має бути позначене тільки як observation.state. Відсутність уніфікованих схем відповідності між цими назвами та відсутність документації призводила до типових критичних помилок, таких як (KeyError), (shape mismatch) або винятків при спробі формування батчу для навчання. Для вирішення цього класу проблем було здійснено ручний аналіз та покроковий дебагінг модулів, відповідальних за обробку конфігурацій.

Таким чином, саме за рахунок емпіричного аналізу, дослідження зчитування конфігів на рівні вихідного коду та перевірки відповідності імен на всіх етапах вдалося встановити набір дійсно зарезервованих імен, що проходять усі перевірки та підтримуються усіма рівнями системи. Це дозволило уніфікувати конфігурацію, зробивши її сумісною як з політикою, так і з середовищем, не вдаючись до глибокої модифікації внутрішнього коду проєкту Lerobot.

3.4.2 Навчання моделі з поступовим усуненням виявлених недоліків

Після завершення повного циклу підготовки, що включав модифікацію симуляційного середовища, створення мультимодального датасету та адаптацію конфігурації моделі π_0 до обраних форматів вхідних і вихідних даних, було здійснено запуск навчання керуючої політики відповідно до заданих архітектурних параметрів.

Навчання моделі проводилося на комп'ютері з використанням OS Linux та без застосування GPU, виключно на CPU з 32 ГБ фізичної оперативної пам'яті. На етапі тестового запуску навчання було виявлено, що наявна конфігурація апаратного забезпечення є недостатньою для стабільного проходження повного навчального циклу, оскільки в пікові моменти споживання оперативної пам'яті перевищувало доступні 32 ГБ. У результаті виникала потреба у розширенні доступної пам'яті для уникнення аварійного завершення процесу. Для цього було створено додаткові swap-розділи за рахунок SSD-накопичувача, що сприяло збільшенню пам'яті ще на 30 ГБ. У підсумку система оперувала з приблизно 60 ГБ доступної пам'яті, з яких під час навчання було задіяно близько 50 ГБ. Також у зв'язку з даними обмеженнями навчання проводилося з розміром батчу, рівним одиниці (batch_size = 1), що значно знижує ефективність батчевої нормалізації та швидкості навчання, однак дозволяє зменшити кількість експлуатованої пам'яті.

Окремим викликом стало те, що у вихідному коді (train.py) із проєкту LeRobot не було передбачено логування основних метрик навчання, таких як значення втрати (loss), норми градієнта (grad_norm), точність, стабільність генерації дій або якість відповідності демонстраціям. Це значно ускладнювало аналіз динаміки навчання та не дозволяло відстежити прогрес у зручній формі. Для вирішення цієї проблеми було внесено власні зміни до тренувального скрипта: додано ініціалізацію логгера

TensorBoard (SummaryWriter), а також функціонал запису основних показників у кожному навчальному кроці.

Після завершення всіх модифікацій і верифікації конфігурацій було розпочато фінальне навчання моделі π_0. У цій версії запуску було усунено всі попередні помилки та недоліки. Було чітко визначено інтервали для збереження контрольних точок моделі – кожні 5 000 навчальних кроків, що дозволяло, у разі потреби, відновлювати навчання з останньої стабільної фази. Крім того, кожні тисячу кроків проводилося автоматичне генерування rollout-відео – створення коротких симуляцій виконання завдання з використанням поточних параметрів моделі. Такі відео дозволяють візуально оцінити прогрес навчання, якість генерації дій, реакцію системи на природномовні інструкції, узгодженість маніпуляцій з поставленим завданням та виявленню потенційних збоїв під час навчання.

У результаті було отримано функціонально натреновану реалізацію моделі pi0, навчену виконувати, через обмежені ресурси, всього одну дію, а саме підняття кубика перед собою. Отримані ваги, лог-файли TensorBoard, проміжні відео та збережені контрольні точки слугували базою для подальшого аналізу результатів навчання та підтвердження ефективності мультимодального підходу у формуванні гнучкої та інтуїтивної системи керування роботизованим маніпулятором.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ

4.1 Оцінка моделі та експериментальні дослідження

Після проходження приблизно 5–6 тисяч кроків навчання, що сумарно тривало близько 14 годин, було здійснено попередню оцінку рівня сформованої політики. Для моніторингу динаміки навчання модель автоматично генерувала відеозаписи виконання завдання кожні тисячу кроків, що дозволяло відстежувати поступову еволюцію поведінки агента та своєчасно фіксувати якісні зміни. На основі аналізу цих візуальних спостережень було встановлено, що модель pi0, за умови використання лише одного прикладу дії в навчальному наборі, досягає прийнятного рівня виконання приблизно після 5 тисяч ітерацій оптимізації. Після завершення процесу навчання було проведено ґрунтовний аналіз отриманих метрик, а також серію цільових експериментів, покликаних оцінити стабільність поведінки моделі, узгодженість її дій із поставленими завданнями та загальну ефективність сформованої політики в умовах симуляційного середовища. У процесі оцінювання використовувалися як числові показники, отримані під час навчання та валідації, так і автоматично згенеровані візуальні матеріали (rollout-відео), що демонстрували реальну реакцію моделі на вхідні інструкції та дозволяли здійснювати якісний аналіз її поведінки.

Графіки метрик, зібрані через TensorBoard, демонструють загалом стійке та поступове покращення як у фазі навчання (рисунок 4.1), так і під час оцінювання (рисунок 4.2). У розділі train спостерігається чітка тенденція зниження функції втрат (train/loss) із початкового значення понад 0.6 до рівня менше ніж 0.15 на 5000-му кроці, що свідчить про успішне пристосування політики до навчального розподілу демонстрацій. Норма

градієнта (train/grad_norm) демонструє плавне зниження, що відповідає стабілізації процесу оптимізації та зменшенню коливань у просторі параметрів.

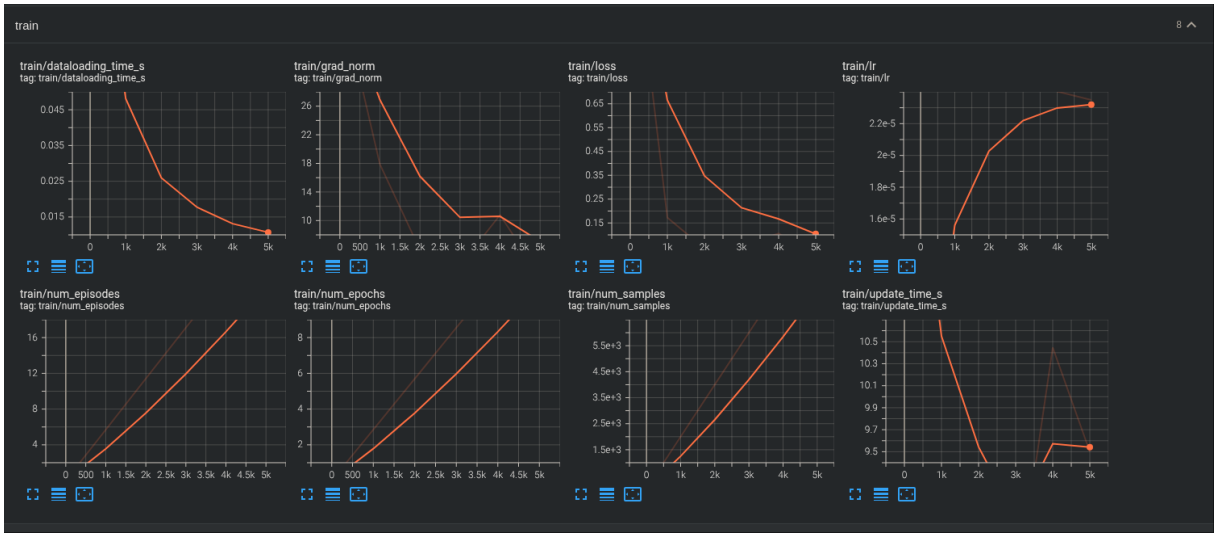

Рисунок 4.1 – Динаміка тренувальних метрик моделі під час навчання

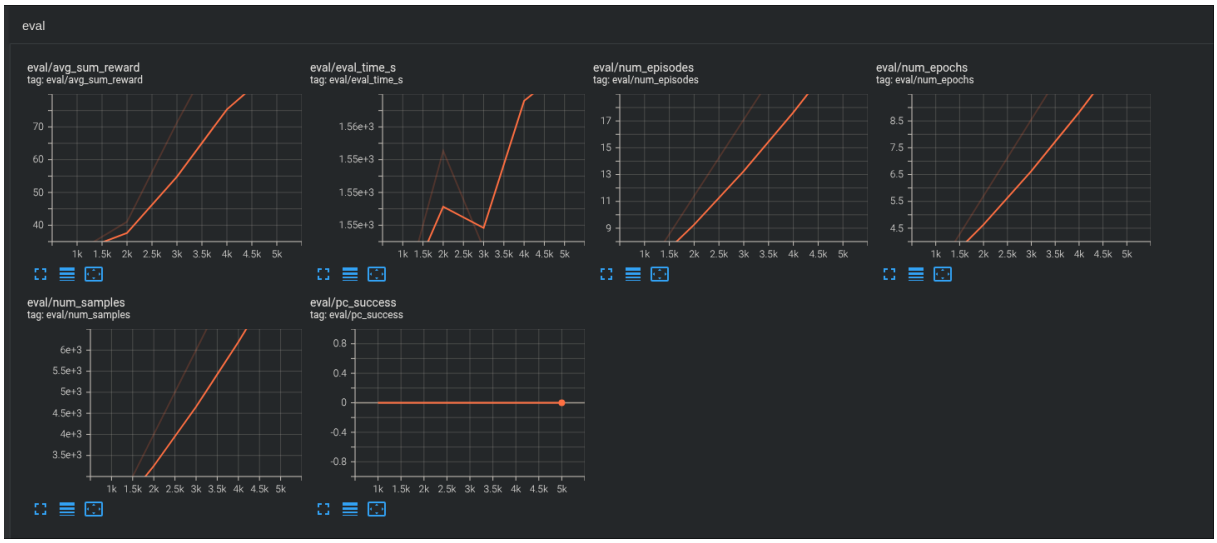

Рисунок 4.2 – Динаміка метрик оцінки (evaluation) моделі на валідаційних епізодах під час навчання

У фазі оцінювання (eval) зафіксовано систематичне покращення показника avg_sum_reward, який із початкових значень ~35 поступово зріс до понад 90. Це свідчить про зростання загальної ефективності стратегії виконання завдань.

Хоча кількісні метрики відіграють важливу роль в об'єктивному аналізі якості навчання моделі, не менш значущим є візуальне оцінювання поведінки системи у процесі виконання завдань, що дозволяє глибше інтерпретувати результати. Саме тому було згенеровано серію відео, які слугують наочним підтвердженням того, наскільки ефективно модель змогла засвоїти одну єдину маніпуляційну дію, представлену у навчальному датасеті, а саме підхід до об’єкта, його захоплення та підняття на задану висоту. Ці візуальні спостереження дозволяють не лише проаналізувати динаміку навченої поведінки, але й оцінити здатність моделі до просторової адаптації. Наприклад, коли кубик був злегка зміщений відносно тієї позиції, яку він займав під час запису демонстрацій у датасеті.

Таким чином згенеровані демонстраційні відео дають змогу наочно оцінити якість поведінки моделі π_0 у завданні, що повністю відповідало прикладу, представленому в навчальному датасеті. Як показали результати спостережень, у переважній більшості випадків (близько 93%) модель успішно відтворює необхідну послідовність дій: маніпулятор підводиться до об'єкта, виконує його захоплення та підняття на задану висоту. При цьому в ряді епізодів спостерігалося незначне відхилення від очікуваної траєкторії, зокрема, траплялися випадки повільного або не зовсім прямолінійного руху, а також легке смикання маніпулятора під час виконання окремих фаз. Тим не менш, загальна якість виконання дії свідчить про те, що модель успішно засвоїла дані дії, як шаблон поведінки та здатна його стабільно відтворювати.

Водночас, результати тестування на завданні з незначною модифікацією умов, наприклад у випадку зміщення кубика відносно його

початкової позиції, яка була використана під час запису демонстрацій, спостерігається дуже обмежену здатність моделі до адаптації. У цих епізодах робот продовжував орієнтуватися на фіксовану позицію об’єкта, з якою був ознайомлений у навчальному прикладі, і не не дуже хотів адаптувати свої рухи до нового розташування кубика. Це призводило до хибних дій, повного ігнорування об'єкта та постійних спроб захопити його на старому місці, що вказує на відсутність узагальнення поведінки поза межами побаченого прикладу. Таким чином, модель, навчена лише на одному варіанті завдання, продемонструвала здатність до точного відтворення конкретної ситуації, але не сформувала стійкої залежності між візуальним входом та положенням цілі в більш загальному сенсі.

Візуальний аналіз результатів навчання моделі $\pi 0$ проводився також за допомогою автоматично згенерованих відеозаписів (rollout), що фіксували поведінку агента протягом валідаційних епізодів. На основі цих матеріалів можна зробити висновки про узгодженість дій робота з навчальною демонстрацією, а також про обмеження моделі щодо адаптації до нових конфігурацій середовища. Як видно на рисунку 4.3, у випадках, коли кубик перебуває у тій самій позиції, що й під час тренування, модель з високою ймовірністю успішно виконує дане завдання – підносить гріпер до кубика, захоплює його і піднімає на цільову висоту. Водночас, при навіть незначному зсуві положення кубика, як показано на рисунку 4.4, модель демонструє неузгоджену поведінку: маніпулятор здійснює рух до колишньої позиції об’єкта, і дія не завершується успішно. Це свідчить про відсутність здатності до просторової генералізації, що є прямим наслідком використання вузького та одномодального навчального прикладу. Така поведінка також вказує на те, що модель не формує стійкого просторового уявлення про розташування об’єкта, покладаючись переважно на патерни, характерні лише для навчального епізоду.

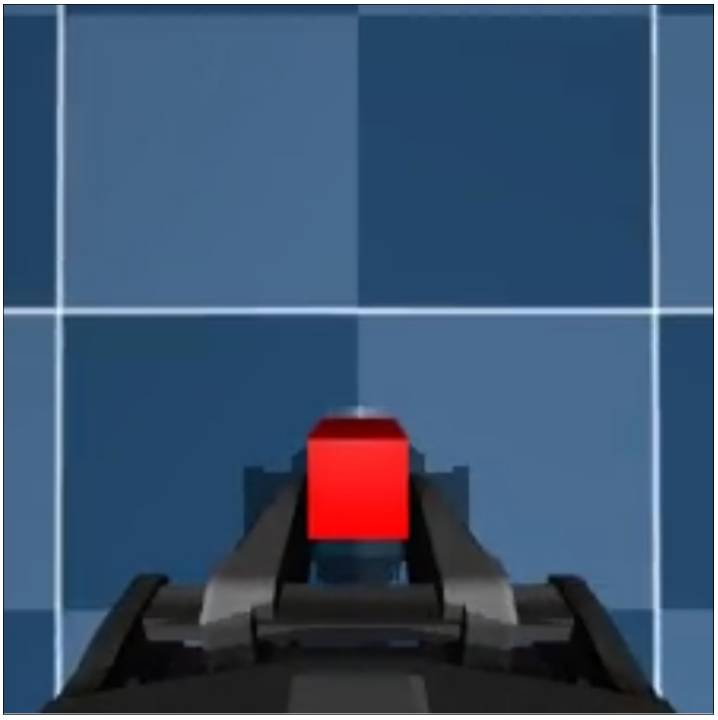

Рисунок 4.3 – Демонстрація успішного захоплення кубика в умовах знайомої конфігурації

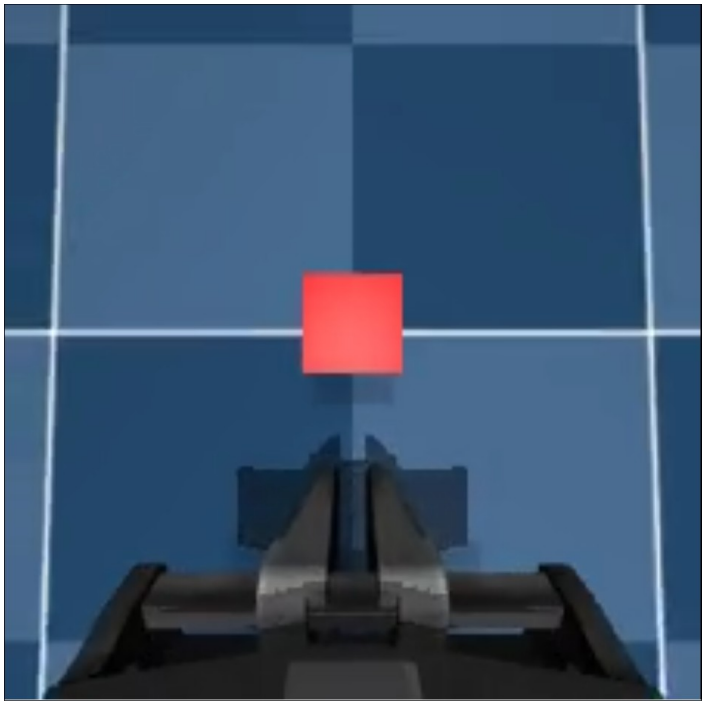

Рисунок 4.4 – Демонстрація неузгодженості поведінки при зміненій початковій конфігурації об’єкта

4.2 Порівняння з традиційними підходами

Традиційні підходи до керування роботизованими маніпуляторами ґрунтуються, як правило, на ручному програмуванні послідовностей рухів або використанні детермінованих контролерів, які передбачають жорстке моделювання навколишнього середовища, чітке визначення кінематичних параметрів системи, та обов’язкову наявність точної сенсорної інформації про розташування об'єктів. Подібні методи є надійними у строго визначених та контрольованих середовищах, однак виявляють низьку гнучкість і вимагають значного ручного налаштування при зміні завдання чи конфігурації сцени.

На відміну від цього, підхід, реалізований у межах даного дослідження, базується на використанні моделі π_0, яка здатна сприймати та інтерпретувати інструкції, сформульовані природною мовою, аналізувати візуальну інформацію з камер та враховувати пропріоцептивний стан робота. Така система не потребує жорстко зафіксованої логіки дій або заздалегідь розрахованих траєкторій – натомість вона адаптивно генерує блоки моторних команд у відповідь на контекст завдання та поточний стан середовища. Це дозволяє забезпечити вищу гнучкість при взаємодії з користувачем та потенційно зменшити вартість розгортання системи в нових умовах, особливо в задачах, де регулярне перепрограмування є недоцільним або неможливим.

Однак результати експериментального тестування показують, що система функціонує, але така гнучкість досягається лише за наявності достатньої кількості варіативних навчальних прикладів. Навчена лише на одному шаблоні дії модель π_0 показала високу здатність до повторення цього конкретного руху, але виявила неспроможність адаптуватися до зміни просторової конфігурації задачі, зокрема при зміщенні об’єкта взаємодії. Це суттєво відрізняє модель від традиційних підходів, у яких положення

об'єкта явно враховується в розрахунках та автоматично включається до процесу побудови траєкторії. Крім того, традиційні методи керування є значно менш вимогливими до обчислювальних ресурсів і, як правило, не потребують глибокого навчання або великих обсягів пам’яті, оскільки базуються на жорстко прописаних інструкціях для виконання. На відміну від них, використання нейронної мережі для керування роботом вимагає значних обчислювальних ресурсів: понад 50 ГБ оперативної пам’яті під час навчання та близько 15 ГБ для відтворення результатів і оцінювання моделі.

Загалом, незважаючи на те що мультимодальні моделі глибокого навчання відкривають нові можливості для інтуїтивного, гнучкого та високорівневого керування роботами, їх практична ефективність напряму залежить від від повноти та обсягу навчального представлення середовища. У випадках реалізації вузьких, чітко визначених дій у передбачуваних середовищах із жорсткими вимогами до точності та повторюваності маніпуляцій, класичні методи керування часто залишаються більш ефективним і надійним вибором. Водночас при масштабуванні навчальних даних та застосуванні багатоінструкційного підходу моделі на зразок π_0 мають потенціал суттєво перевершити традиційні алгоритми в умовах динамічних, частково непередбачуваних або неструктурованих середовищ, де важливу роль відіграє здатність до семантичної інтерпретації завдання та адаптації до змін без ручного втручання.

4.3 Безпека, обмеження та ризики застосування системи

Запропонована система адаптивного керування роботизованим маніпулятором, що ґрунтується на мультимодальній трансформерній нейронній мережі, продемонструвала високу ефективність у віртуальному середовищі та здатність до відтворення заданої поведінки на основі мовних інструкцій. Водночас, при переході до використання подібних систем у

реальних умовах виникає низка об’єктивних обмежень та ризиків, які потребують окремого розгляду. Насамперед, ідеться про проблему переносу знань із симуляції у фізичний світ (так званий *Sim-to-Real gap*), яка полягає у тому, що поведінка агента, сформована в умовах математично точного моделювання, не завжди залишається релевантною в умовах реального середовища, де присутні шум, затримки, неточності сенсорики, фрикційні варіації та механічні люфти.

Зокрема, навіть незначні відмінності в позиціюванні об’єктів або варіації у освітленні можуть призводити до суттєвих відхилень у сприйнятті зображень, що є критичним у контексті мультимодальної моделі, яка спирається на зорову інформацію як одну з основних модальностей. Крім того, сама модель π_θ, будучи детермінованою політикою, навчену з імітації одного прикладу, має обмежену здатність до реакції на несподівані події або нові конфігурації середовища. У фізичних умовах це може становити загрозу як для цілісності апаратної платформи, так і для навколишнього середовища, особливо в разі помилкової інтерпретації інструкції або неправильного сприйняття сцени.

Ще одним аспектом є відсутність вбудованих механізмів перевірки безпечності дій, яка зазвичай реалізується в класичних системах керування через обмеження на сили, швидкість або допустимі зони переміщення. У даній роботі такі перевірки відсутні, оскільки симуляційне середовище не моделювало зіткнення з оператором або крихкими об'єктами, а сама модель не має модулів оцінки ризику. У реальному застосуванні це вимагає або використання додаткових систем контролю, або глибокого перенавчання моделі із включенням обмежень безпеки.

Подолання згаданих обмежень можливе шляхом застосування методів доменної адаптації, включенням віртуальних шумів у процес навчання (domain randomization), а також доповненням моделі системами виявлення помилок, як-от предиктивними модулями

невпевненості (uncertainty estimation). Додатково, перехід до реального середовища має здійснюватися поетапно, з використанням тестового стенду, де можна контролювати фізичну взаємодію робота з об’єктами в умовах ізоляції, забезпечуючи безпечне перенесення навичок з віртуального простору.

Загалом, запропонована система має високий потенціал до практичного використання, однак для її надійної та безпечної роботи в реальному середовищі необхідна подальша адаптація, розширення моделі, а також інтеграція додаткових контролерів безпеки та модулів оцінки контексту.

4.4 Перспективи подальшого розвитку системи

Результати дослідження засвідчили базову працездатність мультимодального підходу до керування роботизованим маніпулятором у симуляційному середовищі, однак для переходу від демонстраційного прототипу до повноцінної адаптивної системи необхідно реалізувати низку суттєвих покращень та функціональних розширень. Насамперед, йдеться про розширення навчального набору – як у кількісному, так і в якісному вимірі. Однієї демонстрації недостатньо для формування узагальненого уявлення про просторові взаємозв’язки між зображенням, об'єктами та рухами, тому подальше формування датасету має включати варіації одного і того самого завдання з різними початковими умовами, кутами огляду, положеннями об’єктів та фоновими характеристиками сцени. Це дозволить моделі розвивати здатність до генералізації, що є критично важливим для практичних застосувань у реальному світі.

Окремим напрямом розвитку є поступовий перехід до тренування та оцінки моделі в умовах реального робототехнічного обладнання. З огляду на те, що LeRobot підтримує узгоджений інтерфейс для роботи як у симуляції,

так і на фізичних роботах, можливо реалізувати перенесення моделі з мінімальними змінами в архітектурі. Для цього необхідно адаптувати датчики, уніфікувати передачу спостережень та забезпечити контроль дій із урахуванням механічних обмежень і затримок у реальному часі. Дообучення моделі безпосередньо на реальному маніпуляторі може проводитися в рамках стратегій fine-tuning або continual learning із низькою швидкістю оновлення параметрів, що дозволить зменшити ризики катастрофічного забування початково навчених поведінок.

Крім того, у межах розширення функціональності системи доцільно розглядати підтримку нових класів завдань, таких як сортування об'єктів, взаємодія з динамічними цілями або послідовні маніпуляції з декількома предметами. Для цього може знадобитися доповнення моделі механізмами короткочасної пам’яті, просторової уваги або зовнішнього планування. Також перспективним є впровадження мультимовної підтримки, що дозволить розширити спектр інструкцій, доступних користувачеві, і вийти за межі заздалегідь прописаних запитів. На архітектурному рівні це може бути реалізовано шляхом підключення більш потужної мовної моделі або навчання системи на великій кількості пар (запит – дія) у симульованих сценаріях.

Загалом, подальший розвиток системи може йти в напрямку зростання масштабованості, гнучкості та інтерпретованості. Розширення набору завдань, підвищення рівня генералізації, перенесення в реальний світ та інтеграція з голосовими або багатомовними інтерфейсами – усе це відкриває широкі можливості для застосування мультимодальних політик не лише у дослідницьких цілях, а й у промислових, сервісних чи побутових сценаріях використання.

ВИСНОВКИ

У межах цього дослідження було реалізовано повноцінну систему адаптивного керування роботизованим маніпулятором на основі мультимодального глибокого навчання, яка інтегрує візуальну інформацію, пропріоцептивні дані та природномовні інструкції. Для досягнення поставленої мети було створено кастомізоване симуляційне середовище на основі фізичного рушія MuJoCo, фреймворку Gymnasium та адаптованого під дані задачі робота VX300S. Це оточення було глибоко інтегроване у програмну архітектуру проєкту LeRobot, що потребувало ручного налаштування конфігурацій, узгодження назв мультимодальних ознак, їх розмірностей та модифікації вихідного коду самого фреймворку. Було впроваджено підтримку візуалізації з камери, обчислення винагороди з урахуванням евристичних ознак, а також забезпечено сумісність із форматами, які очікує мультимодальна нейронна модель.

Обраний нейромережевий фреймворк π_0 (pi-zero), представлений у 2024 році дослідницькою командою Physical Intelligence, був обраний для реалізації високорівневого керування роботизованим маніпулятором у мультимодальному середовищі. Ключовими аргументами на користь цієї моделі стали її модульна архітектура, вбудована підтримка природномовних інструкцій, а також можливість поєднання візуального входу із вектором стану робота. Pi0 є трансформерною політикою, яка базується на передтренованій моделі PaliGemma, що включає візуальний енкодер SigLip та мовну модель Gemma, і дозволяє формувати узгоджені блоки дій у відповідь на мультимодальний контекст. У межах проєкту ця модель була адаптована до кастомного середовища (MyRobotEnv), з урахуванням специфіки структури спостережень, назв ознак та формату даних, що відповідають очікуванням політики.

Навчання моделі проводилося в умовах обмежених апаратних ресурсів – виключно на центральному процесорі, без використання GPU. У зв’язку з недостатнім обсягом оперативної пам’яті – 32 ГБ, для забезпечення стабільного функціонування було додатково активовано swap-розділи, які надали ще 30 ГБ, що дозволило довести доступну пам’ять до 60 ГБ. Навчальний процес здійснювався з розміром батчу, рівним одному, що обмежувало ефективність обробки, однак дозволяло уникнути перевантаження пам’яті. Завдяки ретельній адаптації конфігурацій середовища, правильному узгодженню вхідних ознак та ручному внесенню змін у скрипти (train.py), було реалізовано повноцінний цикл тренування з інтегрованим логуванням у TensorBoard. Крім числових метрик, було налагоджено періодичне збереження контрольних точок політики та генерацію валідаційних відео, які забезпечили наочне спостереження за динамікою навчання. Незважаючи на технічні обмеження, реалізоване навчання підтвердило життєздатність обраної архітектури та її здатність до відтворення демонстраційної поведінки.

Результати експериментального оцінювання показали, що модель π_0, навіть за умов мінімального обсягу навчальних даних, демонструє здатність якісно відтворювати дію, яка була представлена в єдиному прикладі – зокрема, успішно здійснювати захоплення кубика та його підняття. Разом із тим, при зміні конфігурації середовища, наприклад при просторовому зміщенні об'єкта, модель не продемонструвала здатності до адаптації: вона продовжувала намагатися виконати дію в зафіксованій, знайомій позиції, що вказує на відсутність узагальнення поведінки поза межами навчального прикладу. Отже, зроблений висновок полягає в тому, що модель, навіть за умов обмеженого навчального набору, здатна якісно відтворювати одиничну дію, однак виявляє повну неспроможність до просторової адаптації або переносу на нові конфігурації середовища. Для формування такої здатності необхідним є створення більш різноманітного датасету, який

включатиме декілька (наприклад, 5–10) варіацій тієї самої дії з різним розташуванням об’єкта. Такий підхід дозволить моделі виявити узагальнені закономірності між візуальною інформацією та моторними командами, сформувати залежності на вищому семантичному рівні та в перспективі забезпечити адаптивність до динамічних умов середовища. Водночас, слід враховувати, що збільшення варіативності даних вимагатиме суттєвого розширення часу тренування, а також ресурсів для збору демонстрацій і забезпечення їхньої репрезентативності.

Загалом, результати даного дослідження підтверджують доцільність застосування мультимодальних моделей глибокого навчання, таких як π_0, для задач високорівневого інтелектуального керування роботизованими системами. Створене середовище, навчальна інфраструктура та зібрані дані заклали основу для подальшої розбудови системи – зокрема шляхом розширення сценаріїв навчання та перенесення на фізичні робототехнічні платформи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. LeRobot: Making AI for Robotics more accessible with end-to-end learning. GitHub. URL: https://github.com/huggingface/lerobot (date of access: 23.05.2025).

2. *Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware* / T. Zhao et al. Robotics: science and systems 2023. 2023. URL: https://doi.org/10.15607/rss.2023.xix.016 (date of access: 23.05.2025).

3. *MuJoCo playground*. arXiv.org. URL: https://arxiv.org/abs/2502.08844 (date of access: 23.05.2025).

4. Singh R. P., Gergondet P., Kanehiro F. *Mc-Mujoco: simulating articulated robots with FSM controllers in mujoco*. 2023 IEEE/SICE international symposium on system integration (SII), Atlanta, GA, USA, 17–20 January 2023. 2023. URL: https://doi.org/10.1109/sii55687.2023.10039218 (date of access: 23.05.2025).

5. Todorov E., Erez T., Tassa Y. *MuJoCo: a physics engine for model-based control*. 2012 IEEE/RSJ international conference on intelligent robots and systems (IROS 2012), Vilamoura-Algarve, Portugal, 7–12 October 2012. 2012. URL: https://doi.org/10.1109/iros.2012.6386109 (date of access: 23.05.2025).

6. *Gymnasium: a standard interface for reinforcement learning environments*. arXiv.org. URL: https://arxiv.org/abs/2407.17032 (date of access: 23.05.2025).

7. *Our first generalist policy. Physical Intelligence (π)*. URL: https://www.physicalintelligence.company/blog/pi0 (date of access: 23.05.2025).

8. *π_0: a vision-language-action flow model for general robot control*. arXiv.org. URL: https://arxiv.org/abs/2410.24164v1 (дата звернення: 23.05.2025).

9. *PaliGemma: A versatile 3B VLM for transfer*. arXiv.org. URL: https://arxiv.org/abs/2407.07726 (date of access: 23.05.2025).

10. Our latest advances in robot dexterity. Google DeepMind. URL: https://deepmind.google/discover/blog/advances-in-robot-dexterity/?utm_source=chatgpt.com (date of access: 23.05.2025).

11. *RT-2: vision-language-action models transfer web knowledge to robotic control*. arXiv.org. URL: https://arxiv.org/abs/2307.15818 (date of access: 23.05.2025).

12. *ROS-PyBullet interface: a framework for reliable contact simulation and human-robot interaction*. arXiv.org. URL: https://arxiv.org/abs/2210.06887 (date of access: 23.05.2025).

13. Genesis: A generative world for general-purpose robotics & embodied AI learning. GitHub. URL: https://github.com/Genesis-Embodied-AI/Genesis (date of access: 23.05.2025).