НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет робототехніки та приладобудування

Кафедра комп'ютерно-інтегрованих технологій виробництва приладів

До захисту допущено:

Завідувач кафедри

_____ Наталія
СТЕЛЬМАХ

«12» 06 2026 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Комп'ютерно-інтегровані системи та технології в приладобудуванні»

спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології»

на тему: «Напівавтоматизована система відеомовного телеоперування маніпуляторами ViperX»

Виконав:
студент IV курсу, групи ПБ-з21
Клименко Ігор Євгенович _____

Керівник:
проф., д.т.н., проф.
Безуглий М. О. _____

Рецензент:
доцент кафедри АСНК, к.т.н., доцент
Галина Богдан. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2026 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет робототехніки та приладобудування

Кафедра комп'ютерно-інтегрованих технологій виробництва приладів

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 151 «Автоматизація та комп'ютерно-інтегровані технології»

Освітньо-професійна програма «Комп'ютерно-інтегровані системи та технології в приладобудуванні»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Наталія СТЕЛЬМАХ

«26» 05 2026 р

ЗАВДАННЯ

на дипломну роботу студенту

Клименко Ігор Євгенович

1. Тема роботи «Напівавтоматизована система відеомовного телеоперування маніпуляторами ViperX», керівник роботи Безуглий Михайло Олександрович, проф., д.т.н., проф., затверджені наказом по університету від «26» 05 2026 р. № 1905-с

2. Термін подання студентом роботи 01.06.2026

3. Вихідні дані до роботи об'єкт дослідження — напівавтоматизоване відеомовне телеоперування маніпулятором ViperX (14 ступенів вільності) у середовищі імітаційного моделювання aloha_sim; базова візуально-мовна модель Octo для формування представлення задачі; задавальний пристрій — джойстик (2 ступені вільності); горизонти прогнозування дій $H \in \{1, 5, 15\}$; реалізація — мовою Python (PyTorch).

4. Зміст пояснювальної записки аналіз стану проблеми та огляд існуючих рішень телеоперування; розроблення структурної та функціональної схем напівавтоматизованої системи; математична модель і метод індукування керуючого відображення через спектр коваріації дій; програмна реалізація головки телеоперування на базі моделі Octo та розроблення алгоритмів;

експериментальні дослідження, криві навчання та матриці точності реконструкції; формулювання висновків.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо) аркуш 1 — структурна схема системи; аркуш 2 — функціональна схема системи; аркуш 3 — блок-схема алгоритму формування коваріації дій; аркуш 4 — блок-схема алгоритму детермінованої лінійної реконструкції; аркуш 5 — результати експериментальних досліджень.

6. Дата видачі завдання 17.04.2026

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Аналіз стану проблеми, огляд літератури та постановка задачі	17.04–23.04.2026	
2	Розроблення структурної та функціональної схем системи	24.04–30.04.2026	
3	Математична модель і метод індукування керуючого відображення	01.05–08.05.2026	
4	Програмна реалізація головки телеоперування та алгоритмів	09.05–18.05.2026	
5	Експериментальні дослідження (навчання, оцінювання точності)	19.05–26.05.2026	
6	Аналіз результатів досліджень	27.05–30.05.2026	
7	Розроблення графічної частини	31.05–03.06.2026	
8	Оформлення пояснювальної записки	04.06–06.06.2026	
9	Підготовка до захисту	07.06–17.06.2026	

Студент Ігор КЛИМЕНКО

Керівник Михайло БЕЗУГЛИЙ

РЕФЕРАТ

Пояснювальна записка: 73 с., 13 рис., 9 табл., 28 джерел.

Об’єктом дослідження є процес напівавтоматизованого телеоперування багатоступеневими роботизованими маніпуляторами, за якого оператор керує системою високої розмірності через низькорозмірний інтерфейс. Предметом дослідження є методи та засоби індукування задача-обумовленого лінійного керуючого відображення на основі представлень, що породжуються попередньо навченою генералістичною політикою.

Мета роботи полягає в розробленні напівавтоматизованої системи відеомовного телеоперування маніпуляторами ViperX, у якій керуюче відображення «оператор → робот» формується одноразово під час ініціалізації з початкових мультимодальних спостережень, а подальше керування здійснюється без постійного перепрогнозування дій.

У роботі запропоновано формулювання, за якого замість безперервного передбачення дій система оцінює задача-обумовлену коваріацію передбачених дій генералістичної політики й через спектральну структуру цієї коваріації неявно задає геометрію та підсилення відображення користувацького входу в дії робота. Виконано спектральний аналіз збурень регуляризованої матриці Грама, що пов’язує розділеність спектра з часовою стабільністю та згладженістю керування. Запропонований підхід реалізовано шляхом донавчання генералістичної моделі Octo на симуляційних даних `aloha_sim` для горизонтів дій 1, 5 та 15 кроків; виразність і робастність індукованих відображень оцінено методом детермінованої лінійної реконструкції.

Отримані результати свідчать, що проміжний горизонт навчання (5 кроків) забезпечує найкращу крос-горизонтну генералізацію, а індукована структура керування достатньо покриває многовид експертних дій: у найкращому випадку похибка реконструкції становить MSE = 0,113 для відображення з простору входу $\mathbb{R}^2$ у простір дій $\mathbb{R}^{14}$. Це дозволяє

припустити, що представлення генералістичних політик можуть бути ефективно повторно використані для побудови стабільних низькорозмірних інтерфейсів телеоперування без задача-специфічного навчання прогнозу дій.

Практична цінність роботи полягає у застосовності запропонованого підходу до асистивної робототехніки та дистанційного керування, де зниження когнітивного навантаження оператора й повторне використання відкритих генералістичних моделей мають безпосередній прикладний ефект.

КЛЮЧОВІ СЛОВА: ТЕЛЕОПЕРУВАННЯ, ГЕНЕРАЛІСТИЧНА ПОЛІТИКА, КОВАРІАЦІЙНЕ ВІДОБРАЖЕННЯ, СПЕКТРАЛЬНИЙ АНАЛІЗ, ЗВОРОТНИЙ ЗВ’ЯЗОК, ВІЗУАЛЬНО-МОВНІ МОДЕЛІ, ViperX, ALOHA, OCTO, АСИСТИВНА РОБОТОТЕХНІКА.

ABSTRACT

Explanatory note: 73 p., 13 fig., 9 tab., 28 references.

The object of study is the process of semi-automated teleoperation of high–degree-of-freedom robotic manipulators, in which an operator controls a high-dimensional system through a low-dimensional interface. The subject of study is the methods and means of inducing a task-conditioned linear control mapping from the representations produced by a pretrained generalist policy.

The aim of the work is to design a video-language teleoperation system for ViperX manipulators in which the operator-to-robot control mapping is established once at initialization from initial multimodal observations, with subsequent control performed without continuous action re-prediction.

The work proposes a formulation in which, instead of continuously predicting actions, the system estimates a task-conditioned covariance of the generalist policy’s predicted actions and, through the spectral structure of this covariance, implicitly defines the geometry and gains of the mapping from user

input to robot actions. A spectral perturbation analysis of the regularized Gram matrix is carried out, relating spectral separation to the temporal stability and smoothness of control. The approach is implemented by fine-tuning the Octo generalist policy on the simulated `aloha_sim` dataset for action horizons of 1, 5, and 15 steps; the expressiveness and robustness of the induced mappings are assessed via deterministic linear reconstruction.

The results indicate that an intermediate training horizon (5 steps) yields the best cross-horizon generalization and that the induced control structure adequately covers the expert action manifold: in the best case, the reconstruction error is MSE = 0.113 for a mapping from the input space $\mathbb{R}^2$ to the action space $\mathbb{R}^{14}$. This suggests that generalist-policy representations can be effectively repurposed to build stable low-dimensional teleoperation interfaces without task-specific action supervision.

KEYWORDS: TELEOPERATION, GENERALIST POLICY, COVARIANCE MAPPING, SPECTRAL ANALYSIS, FEEDBACK, VISION-LANGUAGE MODELS, ViperX, ALOHA, OCTO, ASSISTIVE ROBOTICS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Скорочення

DoF — ступінь (ступені) вільності (degrees of freedom). САК — система автоматичного керування. ТАК — теорія автоматичного керування. ML — машинне навчання (machine learning). VLM — візуально-мовна модель (vision-language model). VLA — візуально-мовно-дійова модель (vision-language-action model). GRP — генералістична робототехнічна політика (generalist robot policy). OXE — набір даних Open X-Embodiment. ACT — трансформер з блоковим квантуванням дій (Action Chunking Transformer). NLL — від'ємний логарифм правдоподібності (negative log-likelihood). MSE — середньоквадратична похибка (mean squared error). MAE — середня абсолютна похибка (mean absolute error). МК — мікроконтролер. ІК — обернена кінематика (inverse kinematics). RGB-D — кольорове зображення з картою глибини.

Основні позначення

$o \in O$ — спостереження (мультимодальні вхідні дані); o_{image}, o_{text} — візуальна та текстова модальності. $h : O \to T$ — відображення спостережень у простір задачних представлень. $t \in \mathbb{R}^n$ — векторизоване представлення задачі (вкладення). x — стан робота; $\dot{x}$ — швидкості робота (у просторі суглобів або задачному просторі). a — вхід оператора (керуюча команда задавального пристрою). f — динаміка керування; $W(h(o))$ — задача-обумовлене лінійне керуюче відображення. $H(t, x)$ — матриця даних, складена зі стека передбачених векторів дій. Σ_a — емпірична коваріаційна матриця передбачених дій. σ — параметр регуляризації; I — одинична матриця. U, Λ — власні вектори та власні значення (спектральне розкладання Σ_a). B — низькорангова базисна матриця керування; k —

кількість ступенів вільності контролера. z^* — оптимальні коефіцієнти реконструкції; $\hat{a}$ — реконструйована дія.

Терміни

Телеоперування — дистанційне керування роботизованою системою, за якого людина-оператор формує команди руху, а виконавчий механізм відтворює відповідну поведінку. *Генералістична політика* — навчена на різнорідних даних модель керування, придатна до повторного використання та донавчання для нових сенсорів, дій і платформ без навчання з нуля. *Керуюче відображення* — оператор, що перетворює низькорозмірний вхід користувача у багатовимірну дію робота. *Контур зворотного зв'язку* — частина САК, що повертає інформацію про стан об'єкта керування для коригування керуючого впливу.

ВСТУП

Сучасні роботизовані маніпулятори характеризуються високою розмірністю простору керування: навіть один шарнірний маніпулятор зазвичай має шість і більше ступенів вільності, а біманіпуляційні платформи подвоюють це число. Безпосереднє керування такою системою людиною є складним завданням, оскільки оператор змушений одночасно координувати багато незалежних координат, тоді як доступні йому фізичні інтерфейси — джойстики, кнопкові пульти, сенсорні панелі — мають лише дві-три керовані осі. Виникає принципова невідповідність розмірностей між наміром оператора та керованою системою, яку доводиться долати або за рахунок частого перемикання режимів, або за рахунок складних і втомливих процедур ручного наведення.

Дослідницький напрям data-driven телеоперування ставить за мету зменшити це когнітивне навантаження засобами машинного навчання: замість фіксованого, заздалегідь заданого зіставлення осей джойстика з

рухами робота пропонується вивчати відображення з низькорозмірного входу оператора у високорозмірні дії робота безпосередньо з даних. Такий підхід є особливо актуальним для асистивної робототехніки, де самостійне виконання повсякденних дій людьми з обмеженими руховими можливостями має не лише технічне, а й суттєве соціальне значення; за наявними оцінками, користувачі асистивних маніпуляторів здатні виконувати помітно більше щоденних задач порівняно з тими, хто такими засобами не послуговується [1].

Актуальність і новизна. Попри привабливість ідеї, наявні методи навчання керуючих відображень здебільшого спираються на задача-специфічні набори демонстрацій. Це обмежує їхню узагальнюваність: відображення, навчене для одного сценарію, погано переноситься на інший, а збирання нових даних у телеоперуванні ускладнене тим, що цільові дії наперед не визначені — на відміну від класичних задач прогнозування дій, де є чіткі еталони. Останніми роками з’явилися потужні генералістичні політики на трансформерних архітектурах, навчені на великих різнорідних масивах робототехнічних даних і придатні до швидкого донавчання. Це створює нову можливість: замість того щоб збирати дані й навчати окреме відображення для кожної задачі, доцільно один раз використати загальне представлення генералістичної політики і сформувати керуюче відображення на етапі ініціалізації, після чого керування можна спеціалізувати, спираючись лише на пропріоцептивний вхід робота. Новизна роботи полягає саме в такому формулюванні: керуюче відображення індукується не прямим перепрогнозуванням дій, а через спектральну структуру задача-обумовленої коваріації передбачених дій, що неявно кодує і геометрію, і підсилення відображення «оператор → робот».

Обґрунтування необхідності розробки. Класичні фіксовані зіставлення вимагають перемикання режимів і не враховують контекст задачі; задача-специфічне навчання усуває цей недолік, проте дороге у збиранні даних і вузьке за областю застосування. Підхід, що індукує відображення з

представлення генералістичної політики, дозволяє поєднати інтуїтивність керування з широкою застосовністю, уникнувши потреби у спеціальному наборі демонстрацій під кожну ціль.

Основні дослідницькі рішення. У роботі (1) сформульовано керування у формі локально лінійного, задача-обумовленого відображення $\dot{x} = W(h(o))\, a$; (2) запропоновано оцінювати $W(h(o))$ через емпіричну коваріацію дій Σ_a, отриману зі стека передбачень генералістичної політики; (3) виконано спектральний аналіз збурень регуляризованої матриці $\Sigma_a + \sigma I$, що пов'язує розділеність спектра зі стабільністю керування; (4) розроблено власну головку телеоперування на базі моделі Octo та реалізовано алгоритми формування й детермінованої реконструкції відображення; (5) проведено експериментальне дослідження точності на симуляційному наборі `aloha_sim` для горизонтів дій 1, 5 і 15.

Галузі застосування. Асистивна робототехніка (маніпулятори для людей з руховими порушеннями), дистанційне керування у небезпечних або важкодоступних середовищах, а також навчально-дослідницькі стенди на базі недорогих відкритих платформ.

Мета й задачі. Метою роботи є розроблення напівавтоматизованої системи відеомовного телеоперування маніпуляторами ViperX, у якій керуюче відображення формується одноразово під час ініціалізації з початкових мультимодальних спостережень. Для досягнення мети поставлено такі задачі: 1. проаналізувати стан проблеми телеоперування багатоступеневих систем, асистивне та data-driven телеоперування, оглянути візуально-мовні моделі й генералістичні політики, охарактеризувати платформу ViperX/ALOHA; 2. розробити структурну схему напівавтоматизованої системи як системи автоматичного керування зі зворотним зв'язком; 3. побудувати математичну модель індукування керуючого відображення через коваріацію дій і спектральний аналіз збурень; 4. реалізувати програмно головку телеоперування на основі Octo та

алгоритми формування й реконструкції відображення; 5. провести експериментальні дослідження точності реконструкції на наборі `aloha_sim` для трьох горизонтів дій та проаналізувати крос-горизонтну генералізацію.

Об’єкт і предмет дослідження. Об’єкт — процес напівавтоматизованого телеоперування багатоступеневими роботизованими маніпуляторами. Предмет — методи та засоби індукування задача-обумовленого лінійного керуючого відображення на основі представлень генералістичних політик.

Методи дослідження. Використано методи машинного навчання (трансформерні архітектури, дифузійні політики, оцінювання розподілів дій), теорії автоматичного керування (аналіз САК зі зворотним зв’язком, стабільність керування), лінійної алгебри та спектральної теорії збурень (власні розкладання, аналіз чутливості спектра), статистичного оцінювання (емпірична коваріація), а також чисельний експеримент у середовищі симуляції.

Практична цінність. Запропонований підхід дозволяє повторно використати відкриті генералістичні моделі для побудови стабільних низькорозмірних інтерфейсів керування, знижуючи витрати на збирання даних і когнітивне навантаження оператора. Результати безпосередньо орієнтовані на асистивну робототехніку та дистанційне керування. Слід зазначити, що в межах роботи валідація виконується виключно в симуляції на наборі `aloha_sim`; розгортання на фізичній платформі ViperX винесено в перспективи подальших досліджень.

РОЗДІЛ 1

АНАЛІЗ СТАНУ ПРОБЛЕМИ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Телеоперування багатоступеневих роботизованих систем і когнітивне навантаження оператора

Телеоперуванням називають режим роботи робототехнічної системи, за якого керуючі команди формує людина-оператор, а виконавчий механізм відтворює відповідну поведінку у фізичному середовищі. Історично цей режим виник там, де безпосередня присутність людини була неможливою або небезпечною: у ядерній промисловості, підводних і космічних роботах, а згодом — у хірургії та сервісній робототехніці. Спільною рисою цих застосувань є те, що між людиною та об’єктом керування стоїть інтерфейс, від властивостей якого вирішально залежить якість виконання задачі.

Принципова складність телеоперування багатоступеневих маніпуляторів пов’язана з невідповідністю розмірностей. Маніпулятор з шістьма ступенями вільності може незалежно задавати положення (три координати) та орієнтацію (три кути) робочого органа; біманіпуляційні системи додатково подвоюють кількість координат, а з урахуванням захватів сягають чотирнадцяти й більше керованих величин. Натомість типовий фізичний інтерфейс оператора — джойстик або аналогічний задавальний пристрій — надає лише дві-три безперервні осі. Узгодити пряме керування всіма ступенями вільності з таким входом неможливо без додаткових припущень.

Традиційне інженерне рішення полягає у поділі керування на режими. Оператор по черзі активує групи координат: в одному режимі дві осі джойстика рухають робочий орган у площині, в іншому — керують висотою та поворотом, у третьому — розкриттям захвата. Перемикання між режимами вимагає від людини постійно тримати в пам’яті, який режим активний і яку

дію викличе певний рух важеля. Це призводить до явища, добре відомого у літературі з людино-машинної взаємодії, — високого когнітивного навантаження: оператор витрачає увагу не на саму задачу, а на «переклад» свого наміру у послідовність низькорівневих команд. Як наслідок, виконання навіть простих маніпуляцій стає повільним і втомливим, а точні операції часто потребують роботи поблизу кінематичних сингулярностей, де керування стає особливо нестійким [3].

Альтернативою режимному керуванню є зіставлення положень. У такому підході оператор маніпулює зменшеною копією робота — задавальним пристроєм, чиї суглоби механічно або програмно синхронізовані з суглобами виконавчого маніпулятора. Це усуває потребу в перемиканні режимів і забезпечує природне, високосмугове керування, проте потребує спеціалізованого обладнання, фізично доступного далеко не всім користувачам, і не вирішує проблему когнітивного навантаження у випадках, коли керування ведеться через простий пульт.

Отже, центральною проблемою телеоперування багатоступеневих систем є побудова такого відображення з низькорозмірного входу оператора у високорозмірні дії робота, яке було б одночасно інтуїтивним, точним і таким, що враховує контекст виконуваної задачі. Саме на розв’язання цієї проблеми засобами машинного навчання спрямований напрям data-driven телеоперування, розглянутий у підрозділі 1.3.

1.2 Асистивне телеоперування як прикладна галузь

Окремою і соціально вагомою галуззю застосування телеоперування є асистивна робототехніка. Асистивні маніпулятори, що монтуються, зокрема, на крісла колісні, дають людям з обмеженими руховими можливостями змогу самостійно виконувати повсякденні дії — узяти посудину, налити рідину, піднести їжу, відчинити двері. Дослідження економічної ефективності таких систем показують, що їхнє використання дозволяє користувачам виконувати

помітно ширше коло щоденних задач і зменшує потребу в сторонній допомозі, з відповідним соціально-економічним ефектом [1].

Проблема невідповідності розмірностей в асистивному контексті постає особливо гостро. Інтерфейс, доступний користувачеві, нерідко ще бідніший за звичайний джойстик: це може бути двопозиційний перемикач, сенсор підборіддя або пристрій «sip-and-puff», що реагує на вдих і видих. Водночас сам маніпулятор лишається високорозмірним. Класичні асистивні системи долають цю невідповідність режимним перемиканням, успадковуючи всі його недоліки: за свідченнями літератури, постійне перемикання між лінійним та кутовим режимами руху робочого органа є повільним, неінтуїтивним і не забезпечує тонкого керування [8].

Показовим є приклад, що часто наводиться у роботах із цієї тематики: користувач керує семиступеневим маніпулятором за допомогою двовісного джойстика, щоб, скажімо, розрізати шматок їжі [7]. За фіксованого зіставлення осі джойстика відповідають за абстрактні напрямки у просторі, тоді як інтуїтивно бажаною була б ситуація, коли одна вісь керує саме рухом «занурення» інструмента, а друга — рухом «розрізання». Іншими словами, корисне відображення має залежати від задачі, а не бути сталим. Ця ідея — задача-обумовленого, навченого з даних відображення — і лежить в основі сучасного асистивного телеоперування, до якого безпосередньо примикає й тематика цієї роботи. Варто зауважити, що навіть просте відображення «налити рідину», зображене у концептуальній схемі системи (див. підрозділ 1.6), є типовим прикладом задачі, для якої двовимірний вхід оператора має бути спроєктований у скоординований багатовимірний рух маніпулятора.

1.3 Data-driven телеоперування та обмеженість задача-специфічного прогнозування дій

Напрям data-driven телеоперування пропонує замінити сталі зіставлення відображеннями, що вивчаються безпосередньо з даних. Базова ідея полягає у зануренні високорозмірної поведінки робота у

низькорозмірний латентний простір, яким оператор може керувати інтуїтивно. У роботах Д. Лозі та співавторів цю ідею реалізовано за допомогою умовних автокодувальників: маючи набір пар «контекст–дія», модель навчається кодувати дії робота у двовимірний латентний простір z, який і подається оператору як вхід [6], [7]. У такому формулюванні рух уздовж однієї латентної осі може, наприклад, опускати посудину до рівня столу, а вздовж іншої — нахиляти її; саме завдяки тому, що латентний простір навчається під конкретну задачу (як-от наливання), керування стає природним.

Подальші роботи розширили цей напрям. Поєднання навчених латентних дій зі спільною автономією (shared autonomy) дозволило підвищити точність на тонких операціях за рахунок того, що інтерпретація входу оператора змінюється залежно від упевненості робота щодо цілі [9]. Інші дослідження персоналізували відображення під очікування конкретного користувача та досліджували нелінійні карти дій, які потенційно виразніші за лінійні [16]. З’явилися й підходи, що враховують візуальний контекст сцени для формування латентних дій, наближаючи метод до реальних, динамічних середовищ [8].

Попри ці успіхи, наявним методам притаманне спільне обмеження. Латентне відображення, як правило, навчається на задача-специфічному наборі демонстрацій і неявно припускає фіксоване середовище з відомим станом [8]. Це означає, що для кожної нової задачі або нового оточення процедуру збирання даних і навчання потрібно повторювати. У телеоперуванні така вимога є особливо обтяжливою: на відміну від задач навчання з демонстрацій, де еталонні дії відомі, у телеоперуванні цільові дії наперед не визначені — вони народжуються у взаємодії оператора з системою. Відтак побудова великих, загальних наборів даних для телеоперування є значно складнішою, ніж для задач прямого прогнозування дій.

Звідси випливає ключова теза, що мотивує цю роботу. Якщо узагальнюваність відображення обмежена тим, що його щоразу навчають заново на вузьких даних, то природним кроком є перенесення тягаря узагальнення на попередньо навчену генералістичну модель. Замість того щоб навчати окреме відображення, доцільно один раз скористатися загальним представленням генералістичної політики і встановити керуюче відображення на етапі ініціалізації. Після того як сцену ідентифіковано, потреби у безперервному високорівневому візуальному чи текстовому зворотному зв'язку немає, і контролер можна спеціалізувати, спираючись лише на пропріоцептивний вхід робота. Реалізація цієї тези потребує огляду самих генералістичних політик, до якого ми переходимо.

1.4 Візуально-мовні моделі та генералістичні політики на трансформерних архітектурах

Поштовхом до появи генералістичних робототехнічних політик стали успіхи великих візуально-мовних моделей. Візуально-мовна модель (VLM) поєднує зорове сприйняття з обробкою природної мови та здатна співвідносити зображення з текстовими описами, успадковуючи широкі знання про світ із масштабного попереднього навчання. Природним розвитком цієї ідеї у робототехніці стали візуально-мовно-дійові моделі (VLA), які додають до сприйняття та мови ще й вихід у простір дій: модель безпосередньо відображає те, що вона бачить, і те, про що її просять, у низькорівневі команди робота.

Першою помітною роботою цього класу стала RT-1 — трансформер для керування у реальному часі, навчений на великому масиві демонстрацій [10]. Її розвиток, RT-2, продемонстрував, що знання, здобуті з вебмасштабних візуально-мовних даних, можна перенести у керування роботом: дії дискретизуються й подаються як текстові токени, що дозволяє єдиній моделі співтренуватися і на вебданих, і на робототехнічних демонстраціях, виявляючи здатність до узагальнення на не бачені раніше об'єкти та

інструкції [11]. Відкрита модель OpenVLA на 7 мільярдів параметрів, навчена приблизно на мільйоні реальних епізодів, зробила цей клас доступним дослідницькій спільноті [12], а модель π_0 застосувала механізм узгодження потоків (flow matching) поверх попередньо навченої візуально-мовної моделі для генерування неперервних дій на різнорідних платформах [13]. Окремо слід згадати трансформер з блоковим квантуванням дій ACT, що, хоча й є задача-специфічним і позбавленим мовного обумовлення, задав важливий орієнтир точності для біманіпуляційної платформи, до якої належить ViperX [15].

Спільною передумовою для всіх цих моделей стала поява великих агрегованих наборів даних, насамперед Open X-Embodiment, що об'єднав демонстрації з багатьох лабораторій і платформ [14]. Саме на підмножині цього масиву навчено модель Octo, обрану в цій роботі як базову генералістичну політику.

Модель Octo. Octo — це трансформерна політика з дифузійною головкою дій, попередньо навчена на 800 тисячах робототехнічних епізодів з набору Open X-Embodiment (відібрано 25 наборів даних) [2]. Архітектурно модель побудована за принципом модульного токенізування: текстові інструкції перетворюються на токени мовним кодувальником, зображення — на токени згортковими блоками, а спеціальні «зчитувальні» токени (readout tokens) акумулюють контекст для головок дій. Трансформерний кістяк з блоковою маскою уваги обробляє цю послідовність і формує латентне представлення задачі, яке споживається головкою дій. Завдяки модульній структурі та токенізованому поданню входів Octo можна донавчати під нові сенсори, простори дій і морфологію роботів без перенавчання основної частини моделі, причому донавчання здійсненне за кілька годин на споживчих графічних процесорах. У базовому режимі модель підтримує обумовлення як мовною командою, так і цільовим зображенням і за

продуктивністю перевершує відкриту політику RT-1-X, наближаючись до значно більшої моделі RT-2-X.

Для цілей цієї роботи принципово важливими є дві властивості Octo. По-перше, це здатність до ефективного донавчання на новий простір дій — саме вона дозволяє замінити штатну дифузійну головку власною головкою телеоперування (див. Розділ 4). По-друге, це багатий, попередньо навчений латентний простір представлення задачі, який і відіграє роль відображення $h : O \to T$ у запропонованому формулюванні. Зведене порівняння розглянутих політик наведено у таблиці 1.1.

Таблиця 1.1 — Порівняння репрезентативних генералістичних і VLA-політик

Модель	Тип / головка дій	Дані для навчання	Обумовлення	Відкритість
RT-1 [10]	трансформер, дискретні токени дій	реальні демонстрації	мова	частково
RT-2 [11]	VLA, дії як текстові токени	веб + робот	мова	ні
OpenVLA [12]	VLA, 7 млрд парам.	~1 млн епізодів	мова	так
π_0 [13]	VLA, flow matching	різнорідні платформи	мова	частково
ACT [15]	CVAE, блоки дій	демонстрації ALOHA	без мови	так
Octo [2]	трансформер, дифузійна головка	800 тис. епізодів OXE	мова / ціль-зображення	так

Як видно з таблиці 1.1, Octo поєднує повну відкритість, гнучке обумовлення та здатність до швидкого донавчання, що й зумовлює її вибір як базової моделі. Водночас принципова відмінність цієї роботи від наведених підходів полягає не у виборі моделі, а у способі її використання: усі перелічені політики застосовують для безперервного прогнозування дій, тоді як у цій роботі представлення Octo використовується для одноразового індукування керуючого відображення (див. підрозділ 1.6 та Розділ 3).

1.5 Маніпулятори ViperX і біманіпуляційна платформа ALOHA

Цільовою робототехнічною платформою у цій роботі є маніпулятори ViperX у складі біманіпуляційної системи ALOHA. ViperX-300 6DOF — це шестиступеневий шарнірний маніпулятор сімейства Interbotix X-Series, побудований на смарт-сервоприводах DYNAMIXEL (зокрема XM540-W270 та XM430-W350). Привод забезпечує високу роздільну здатність позиціонування, зворотний зв'язок за положенням, навантаженням і температурою, а також програмований контур регулювання. Керування здійснюється через інтерфейс DYNAMIXEL U2D2 з підтримкою ROS. Основні технічні характеристики маніпулятора зведено у таблиці 1.2.

Таблиця 1.2 — Основні технічні характеристики ViperX-300 6DOF

Параметр	Значення
Кількість ступенів вільності	6 (повний оберт основи 360°)
Робоча досяжність	≈ 750 мм від центра основи
Робоче навантаження	≈ 750 г (у межах досяжності)
Сервоприводи	DYNAMIXEL XM540-W270, XM430-W350
Інтерфейс керування	DYNAMIXEL U2D2 (RS-485), ROS
Зворотний зв'язок приводу	положення, навантаження, температура
Конструкція	екструдований алюміній 20×40 мм, акриловий захист електроніки

Платформа ALOHA («A Low-cost Open-source Hardware System for Bimanual Teleoperation»), запропонована Т. Чжао та співавторами [3], реалізує біманіпуляцію за схемою «leader–follower» (задавальний–виконавчий). Два менші маніпулятори WidowX виконують роль задавальних пристроїв (leaders), якими оператор маніпулює безпосередньо, а два більші ViperX є виконавчими (followers) і відтворюють рух у робочому просторі. Замість зіставлення у задачному просторі ALOHA застосовує пряме зіставлення у просторі суглобів: положення суглобів задавального маніпулятора безпосередньо реплікуються на виконавчий. Перевагами такого підходу є гарантоване високосмугове керування в межах суглобових обмежень, нижчі обчислювальні витрати й затримки, а також відсутність частих відмов оберненої кінематики поблизу сингулярностей, типових для шестиступеневих

маніпуляторів без надлишковості [3]. Структуру керування ALOHA подано на рисунку 1.1.

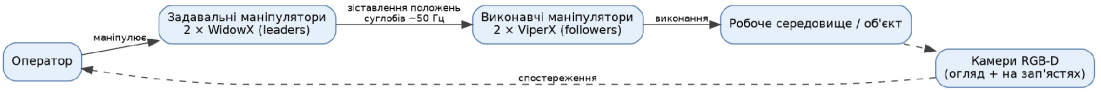


Рисунок 1.1 — Схема телеоперування платформи ALOHA за принципом «leader–follower»

З погляду розмірності керування біманіпуляційна конфігурація ALOHA налічує по сім керованих величин на кожен маніпулятор (шість суглобів і захват), тобто загалом чотирнадцять ступенів вільності. Саме цей чотирнадцятивимірний простір дій ($\dot{x} \in \mathbb{R}^{14}$) є цільовим у запропонованій системі, тоді як вхід оператора лишається двовимірним ($a \in \mathbb{R}^2$). Подальша версія платформи, ALOHA 2, удосконалила ергономіку та надійність — зокрема, конструкцію захватів з низьким тертям, пасивну компенсацію ваги задавальних маніпуляторів і розташування камер RealSense D405, — а також надала відкриту модель середовища у симуляторі MuJoCo [5]. Наявність такої симуляційної моделі є важливою для цієї роботи, оскільки всі експериментальні дослідження виконуються виключно у симуляції на наборі даних `aloha_sim` (Berkeley RAIL), без розгортання на фізичному обладнанні. Відтак ViperX тут виступає як об’єкт керування у симуляційному поданні, а отримані результати характеризують властивості індукованого відображення, а не показники фізичної системи.

1.6 Постановка задачі дослідження

Узагальнюючи викладене, сформулюємо задачу дослідження. Дано: цільова біманіпуляційна платформа ViperX з чотирнадцятивимірним простором дій, двовимірний задавальний пристрій оператора, відкрита генералістична політика Octo та симуляційний набір даних `aloha_sim`. Потрібно: побудувати напівавтоматизовану систему телеоперування, у якій керуюче відображення з простору входу оператора $\mathbb{R}^2$ у простір дій робота

$\mathbb{R}^{14}$ формується один раз на етапі ініціалізації з початкових мультимодальних спостережень і лишається стабільним у часі.

Концептуально систему можна подати трьома ланками: спостережні дані $o \in O$ (відеопотік і текстовий опис задачі, наприклад «налити рідину») надходять на екстрактор ознак — генералістичну політику, що формує задача-обумовлене керуюче відображення, — після чого цим відображенням перетворюється двовимірний вхід оператора у багатовимірну дію робота, яка й виконується платформою (рисунок 1.2).

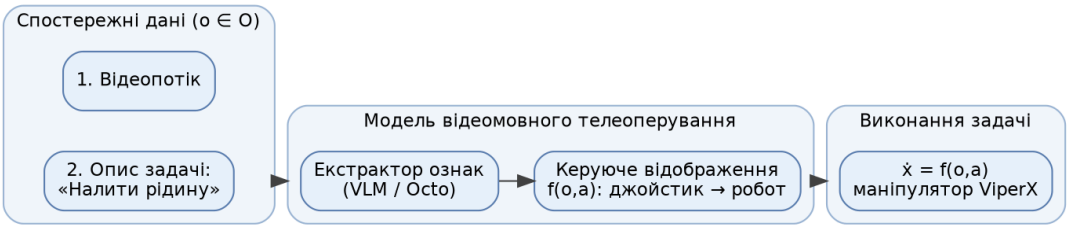


Рисунок 1.2 — Концептуальний конвеєр відеомовного телеоперування: ознаки, видобуті генералістичною моделлю, використовуються як контекст для низькорівневого керування

Ключове проєктне рішення, що відрізняє запропоновану систему від оглянутих у підрозділі 1.3 методів, полягає в наступному. По-перше, відображення не перенавчається під кожну задачу на спеціальному наборі демонстрацій, а індукується з представлення вже навченої генералістичної політики. По-друге, керування не зводиться до безперервного перепрогнозування дій: натомість із передбачень політики оцінюється задача-обумовлена коваріація дій, спектральна структура якої й задає геометрію та підсилення відображення «оператор → робот». По-третє, після ідентифікації сцени високорівневий візуально-мовний зворотний зв’язок стає непотрібним, і контролер спеціалізується лише за пропріоцептивним входом.

Таке формулювання природно вкладається у рамки теорії автоматичного керування: оператор виступає джерелом задавального впливу,

індуковане відображення $W(h(o))$ — регулятором, маніпулятор ViperX — об’єктом керування, а пропріоцептивні датчики замикають контур зворотного зв’язку. Саме розроблення відповідної структурної схеми як системи автоматичного керування зі зворотним зв’язком є предметом Розділу 2, тоді як математичне обґрунтування індукування відображення через коваріацію дій і спектральний аналіз винесено в Розділ 3.

Висновки до розділу 1

У розділі проаналізовано стан проблеми телеоперування багатоступеневих роботизованих систем. Показано, що центральною складністю є невідповідність між високою розмірністю простору керування маніпулятора та низькою розмірністю доступного оператору інтерфейсу, що за традиційного режимного керування спричиняє високе когнітивне навантаження. Розглянуто асистивну робототехніку як соціально вагому галузь, де ця проблема постає особливо гостро, а задача-обумовлене відображення є необхідною умовою інтуїтивного керування.

Проаналізовано напрям data-driven телеоперування та встановлено його ключове обмеження: навчені латентні відображення здебільшого є задача-специфічними, припускають фіксоване середовище й потребують повторного збирання даних для кожного нового сценарію, що у телеоперуванні особливо обтяжливо через невизначеність цільових дій. На підставі цього обґрунтовано доцільність перенесення тягаря узагальнення на попередньо навчену генералістичну політику.

Оглянуто візуально-мовні моделі та генералістичні політики (RT-1, RT-2, OpenVLA, π_0, ACT, Octo) і обґрунтовано вибір моделі Octo як базової — з огляду на її повну відкритість, гнучке обумовлення та здатність до швидкого донавчання на новий простір дій. Охарактеризовано цільову платформу ViperX у складі біманіпуляційної системи ALOHA, визначено цільовий простір дій $\mathbb{R}^{14}$ та двовимірний вхід оператора, а також зафіксовано симуляційний характер подальшої валідації.

Сформульовано задачу дослідження та окреслено ключові проєктні рішення, що відрізняють запропоновану систему від наявних підходів: одноразове індукування відображення з представлення генералістичної політики, заміна безперервного прогнозування дій оцінюванням задача-обумовленої коваріації та спеціалізація контролера за пропріоцептивним входом після ідентифікації сцени. Це створює підґрунтя для розроблення структурної схеми напівавтоматизованої системи, якому присвячено наступний розділ.

РОЗДІЛ 2

РОЗРОБЛЕННЯ СТРУКТУРНОЇ СХЕМИ НАПІВАВТОМАТИЗОВАНОЇ СИСТЕМИ

Цей розділ є центральним для дипломної роботи. У ньому розглянуто запропоновану напівавтоматизовану систему відеомовного телеоперування як систему автоматичного керування (САК) із замкненим контуром зворотного зв’язку, виокремлено її функціональні блоки та поставлено кожному з них у відповідність елемент теорії автоматичного керування (ТАК), визначено місце генералістичної моделі у структурі та обґрунтовано реалізацію керуючих обчислень на рівні вбудованої системи. Завершується розділ повним описом структурної та функціональної схем, призначених для перенесення на креслення формату А1.

2.1 Система телеоперування як система автоматичного керування зі зворотним зв’язком

Розгляд телеоперування з позицій теорії автоматичного керування дозволяє застосувати до нього усталений апарат аналізу замкнених систем. У класичній САК виокремлюють задавальний вплив, регулятор, об’єкт керування, вимірювальний (датчиковий) елемент та контур зворотного

зв’язку, що повертає інформацію про стан об’єкта для коригування керуючого впливу [17], [18]. Запропонована система природно вкладається у цю структуру, проте має істотну особливість: частину функцій регулятора виконує людина-оператор, а частину — автоматичне керуюче відображення. Саме тому система визначається як *напівавтоматизована*.

Поняття напівавтоматизованого керування тісно пов’язане з усталеною в робототехніці концепцією спільного керування (shared control), за якої виконання задачі та керуючий вплив розподіляються між людиною та автоматикою [19]. На одному полюсі цього спектра перебуває повне телеоперування, коли оператор безпосередньо задає всі координати руху; на протилежному — повна автономія, що виключає людину з контуру. Напівавтоматизований режим займає проміжне положення: оператор формує лише низькорозмірний намір, а автоматика бере на себе координацію високорозмірного руху. У запропонованій системі цей розподіл реалізовано так: оператор задає двовимірну команду, а індуковане керуюче відображення розгортає її у скоординовану чотирнадцятивимірну дію маніпулятора. Подібний розподіл функцій між людиною та регулятором є предметом досліджень з арбітражу керування в межах спільної автономії [20].

Узагальнену структурну схему системи як САК подано на рисунку 2.1. Оператор виступає джерелом задавального впливу: спостерігаючи стан робота за відеопотоком та опосередковано — за пропріоцептивним зворотним зв’язком, він формує намір і втілює його у фізичну дію над задавальним пристроєм. Задавальний пристрій перетворює цю дію на нормований вектор команди a. Регулятором є індуковане лінійне відображення $W(h(o))$, що перетворює команду оператора на вектор завдань керованих координат маніпулятора. Об’єктом керування є маніпулятор ViperX, на який, окрім керуючого впливу, діють і зовнішні збурення. Вимірювальний елемент (пропріоцептивні датчики) повертає інформацію про стан x, замикаючи контур зворотного зв’язку як на рівні регулятора, так і на рівні оператора.

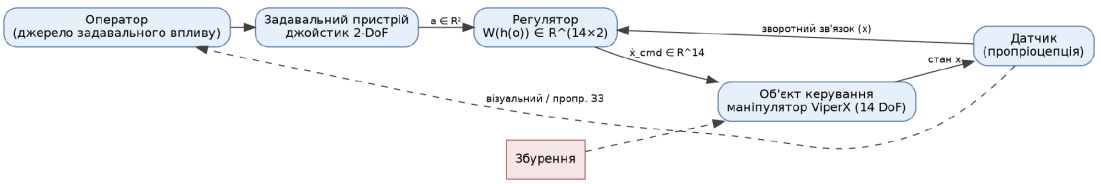


Рисунок 2.1 — Узагальнена структурна схема напівавтоматизованої системи телеоперування як САК зі зворотним зв'язком

Зв'язок між командою оператора та керованими координатами в неперервному часі описується локально лінійним законом керування

$$\dot{x} = W(h(o))\, a, \qquad (2.1)$$

де $\dot{x} \in \mathbb{R}^{14}$ — вектор швидкостей керованих координат маніпулятора; $a \in \mathbb{R}^2$ — вектор команди оператора; $W(h(o)) \in \mathbb{R}^{14\times 2}$ — задача-обумовлене лінійне керуюче відображення (регулятор), сформоване з представлення $h(o)$ початкових спостережень.

Вираз (2.1) є рівнянням регулятора у структурі рисунка 2.1: він задає, як низькорозмірний задавальний вплив перетворюється на високорозмірний керуючий сигнал. Спосіб формування матриці $W(h(o))$ становить предмет наступних розділів; у цьому ж розділі вона розглядається як даний елемент структури, властивості якого (стабільність, згладженість, виразність) визначають якість керування.

2.2 Функціональні блоки системи

Розглянемо функціональні блоки системи докладніше та поставимо кожному з них у відповідність елемент ТАК. Зведення цих відповідностей наведено у таблиці 2.1.

Оператор. У структурі САК оператор виконує роль зовнішнього, найвищого рівня регулювання: він візуально оцінює неузгодженість між бажаним і поточним станом задачі та коригує свою команду. На відміну від суто автоматичного компаратора, оператор оперує семантичним уявленням

про мету («посудину нахилено недостатньо», «захват ще не над об'єктом») і не потребує явного числового завдання. Саме людина замикає найповільніший, але найгнучкіший контур зворотного зв'язку.

Задавальний пристрій. Двовимірний джойстик перетворює фізичну дію оператора на нормований вектор $a \in \mathbb{R}^2$. З погляду ТАК це задавальний пристрій, що формує сигнал завдання. Дві осі джойстика відповідають двом ступеням вільності користувацького інтерфейсу; саме ця розмірність $k = 2$ визначає ранг керуючого базису (див. Розділ 3 та Розділ 4).

Регулятор. Роль регулятора відіграє індуковане відображення $W(h(o))$. На відміну від класичного регулятора, коефіцієнти якого добирає інженер, тут параметри регулятора формуються автоматично з представлення генералістичної політики на етапі ініціалізації. Принципово, що після формування регулятор лишається сталим протягом сеансу керування: він не перераховується на кожному такті, що відрізняє запропонований підхід від методів безперервного перепрогнозування дій.

Об'єкт керування. Об'єктом керування є біманіпуляційна система ViperX з чотирнадцятьма керованими координатами. Слід зазначити, що об'єкт сам по собі є динамічною системою з власною інерційністю та підпорядкованим контуром приводів; це зумовлює каскадну структуру керування, розглянуту нижче.

Вимірювальний елемент і контур зворотного зв'язку. Пропріоцептивні датчики (енкодери суглобів, стан захватів) формують вектор стану x, що повертається до регулятора для його обумовлення та до оператора як зворотний зв'язок. Згідно з постановкою задачі (підрозділ 1.6), після ідентифікації сцени саме пропріоцептивний вхід є достатнім для обумовлення контролера, тоді як високорівневий візуально-мовний зворотний зв'язок стає непотрібним.

Блок формування параметрів регулятора. Окремим елементом структури є генералістична політика, що відіграє роль блоку ідентифікації: з початкових спостережень вона формує представлення задачі та похідні від нього параметри регулятора $W(h(o))$. Цей блок працює одноразово і винесений за межі контуру реального часу (підрозділ 2.3).

Таблиця 2.1 — Відповідність функціональних блоків системи елементам ТАК

Блок системи	Елемент ТАК	Сигнал / величина
Оператор	джерело задавального впливу (зовнішній регулятор)	намір, візуальна оцінка стану
Задавальний пристрій (джойстик 2-DoF)	задавальний пристрій	$a \in \mathbb{R}^2$
Індуковане відображення $W(h(o))$	регулятор	$\dot{x}_{\text{cmd}} = Wa$
Маніпулятор ViperX	об’єкт керування	$x \in \mathbb{R}^{14}$
Сервопривід DYNAMIXEL (ПІД)	підпорядкований регулятор (внутрішній контур)	$\tau,\ q$
Енкодери / пропріоцепція	вимірювальний (датчиковий) елемент	x
Контур зворотного зв’язку	головний зворотний зв’язок	стан $x \to$ регулятор / оператор
Генералістична політика (Octo)	блок ідентифікації / формування параметрів регулятора	$t,\ \Sigma_a \to W$

Як видно з таблиці 2.1, об’єкт керування фактично охоплює два рівні регулювання. Це підводить до каскадної (двоконтурної) структури керування, поданої на рисунку 2.2. Зовнішній контур утворюють оператор та індукований регулятор $W(h(o))$: вони формують завдання положень суглобів. Внутрішній контур реалізують самі сервоприводи DYNAMIXEL, кожен з яких відпрацьовує задане положення власним ПІД-регулятором, замикаючись на сигнал енкодера. Така декомпозиція є типовою для робототехнічних маніпуляторів і дозволяє відокремити повільну,

контекстно-залежну координацію (зовнішній контур) від швидкого, локального відпрацювання положень (внутрішній контур).

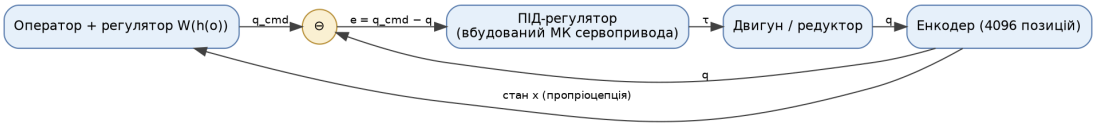


Рисунок 2.2 — Каскадна двоконтурна структура керування: зовнішній контур телеоперування та внутрішній сервопривідний контур

2.3 Місце генералістичної моделі у структурі системи

Принциповою архітектурною особливістю системи є розділення роботи на дві фази, що різняться за обчислювальним навантаженням і частотою виконання (рисунок 2.3).

Фаза 1 — ініціалізація (виконується одноразово). На вхід надходять початкові мультимодальні спостереження o: відеокадр сцени o_{image} та текстовий опис задачі o_{text} (наприклад, «налити рідину»). Екстрактор ознак — генералістична політика Octo — реалізує відображення $h : O \to T$ і формує векторизоване представлення задачі t. З цього представлення оцінюється задача-обумовлена коваріація передбачених дій Σ_a, спектральне розкладання якої дає керуюче відображення $W(h(o))$. Уся ця обчислювально витратна обробка, що включає інференс великої трансформерної моделі, виконується один раз — на старті сеансу.

Фаза 2 — керування в реальному часі (виконується циклічно). Сформоване відображення $W(h(o))$ фіксується. У контурі реального часу беруть участь лише команда оператора a та пропріоцептивний стан x; обчислення зводяться до множення матриці на вектор та інтегрування, що є надзвичайно дешевою операцією, придатною до виконання на мікроконтролері. Жодного інференсу візуально-мовної моделі на цій фазі не відбувається.

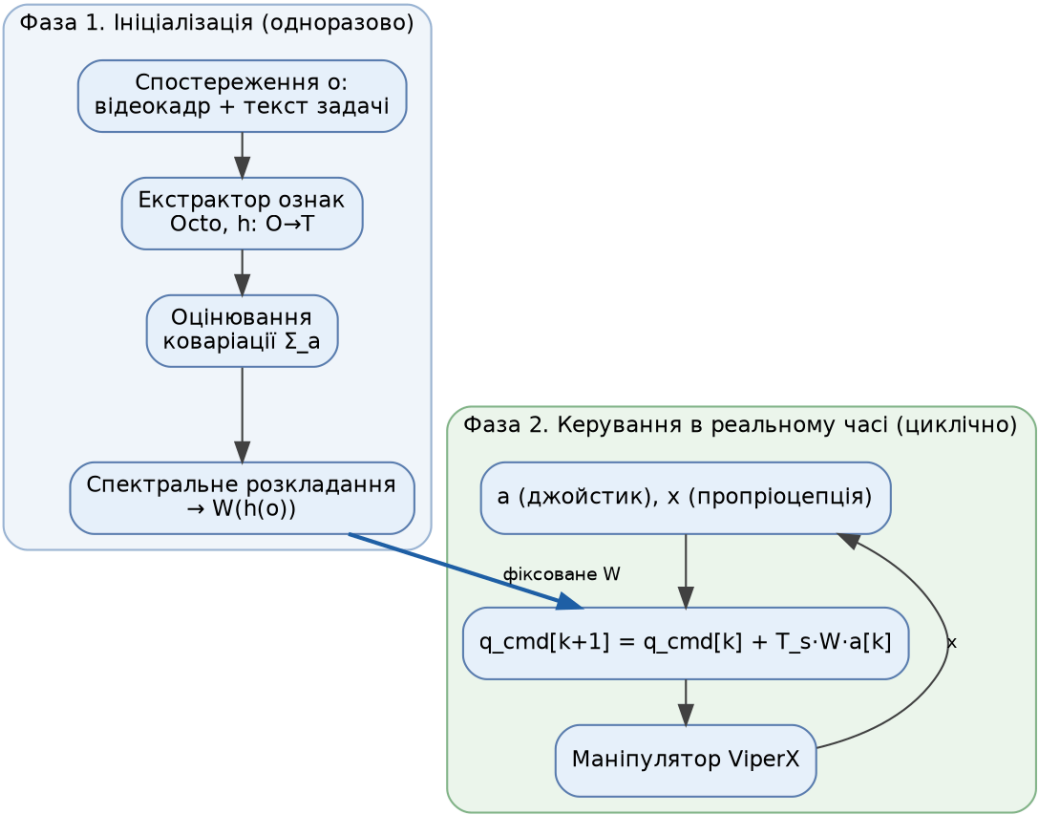


Рисунок 2.3 — Дві фази роботи системи: одноразове формування керуючого відображення та циклічне керування в реальному часі

Таке розділення має кілька важливих наслідків для структури системи. По-перше, воно усуває потребу в постійному високопродуктивному обчислювачі в контурі реального часу: дорогий інференс винесено за межі керуючого циклу. По-друге, воно знижує затримку в контурі керування, оскільки на кожному такті виконується лише легка лінійна операція. По-третє, воно підвищує передбачуваність поведінки: оскільки регулятор сталий, керування не зазнає раптових змін через перепрогнозування, що узгоджується з вимогою часової стабільності, обґрунтованою спектральним аналізом у Розділі 3. Отже, генералістична модель у запропонованій структурі відіграє роль не безперервного контролера, а блоку одноразового налаштування регулятора зі сцени та опису задачі.

2.4 Реалізація обчислень і керування на рівні вбудованої системи

Реалізація системи передбачає три рівні обчислень із різними вимогами до продуктивності та детермінованості.

Обчислювальний вузол. На цьому рівні виконується Фаза 1: інференс генералістичної моделі, оцінювання коваріації та обчислення відображення $W(h(o))$. Вузол не входить до контуру реального часу і може бути реалізований на потужному вбудованому комп'ютері або робочій станції. Результатом його роботи є компактна матриця $W \in \mathbb{R}^{14\times 2}$, яка завантажується до контролера реального часу.

Контролер реального часу (мікроконтролер). Цей рівень реалізує Фазу 2 — дискретизований закон керування з фіксованим періодом T_s. На кожному такті контролер зчитує команду джойстика, обчислює оновлення завдань положень суглобів та надсилає їх сервоприводам. Оскільки на цьому рівні виконується лише множення сталої матриці на двовимірний вектор та інтегрування, обчислювальна складність є мінімальною, що робить можливою реалізацію на звичайному мікроконтролері.

Сервоприводи. Кожен суглоб маніпулятора керується смарт-сервоприводом DYNAMIXEL, що містить власний мікроконтролер і реалізує підпорядкований ПІД-контур позиціонування з високою роздільною здатністю енкодера (4096 позицій на оберт) та програмованими коефіцієнтами регулювання [4]. Цей рівень забезпечує швидке локальне відпрацювання заданих положень.

Неперервний закон керування (2.1) на рівні мікроконтролера реалізується у дискретній формі. Інтегруючи швидкість методом Ейлера, отримуємо рекурентне оновлення завдань положень суглобів:

$$q_{\mathrm{cmd}}[k+1] = q_{\mathrm{cmd}}[k] + T_s\, W(h(o))\, a[k], \qquad (2.2)$$

де $q_{\text{cmd}}[k]$ — вектор завдань положень суглобів на k-му такті; T_s — період дискретизації керуючого циклу; $a[k]$ — вибірка команди оператора на k-му такті.

Період дискретизації обирають із умови коректного відтворення динаміки рухів оператора згідно з теоремою Найквіста–Шеннона:

$$T_s = \frac{1}{f_s}, \qquad f_s \geq 2 f_{\max}, \qquad (2.3)$$

де f_s — частота дискретизації керуючого циклу; $f_{\max}$ — найвища істотна частота у спектрі навмисних рухів оператора. Оскільки смуга пропускання довільних рухів людини рідко перевищує кілька герц, частота циклу порядку 50 Гц (тобто $T_s = 20$ мс) забезпечує значний запас. Така частота узгоджується з робочою частотою контуру керування платформи ALOHA, для якої затримка в каналі «задавальний–виконавчий маніпулятор» становить одиниці мілісекунд, а цільове значення не перевищує десяти мілісекунд [3], [5].

Підпорядкований контур сервопривода реалізує дискретний ПІД-закон:

$$u[k] = K_p\, e[k] + K_i \sum_{j=0}^{k} e[j]\, T_s + K_d \frac{e[k] - e[k-1]}{T_s}, \qquad e[k] = q_{\text{cmd}}[k] -$$

де $u[k]$ — керуючий вплив на привод; K_p, K_i, K_d — коефіцієнти ПІД-регулятора; $e[k]$ — похибка позиціонування суглоба; $q[k]$ — виміряне положення.

Окремо варто пов’язати горизонт дій, що досліджується експериментально (Розділ 5), з фізичним часом керування. Горизонт у H кроків відповідає часовому інтервалу

$$\tau_H = H\, T_s, \qquad (2.5)$$

де τ_H — фізична тривалість горизонту дій; H — горизонт (кількість кроків). Так, за $f_s = 50$ Гц горизонти 1, 5 і 15 кроків відповідають інтервалам приблизно 0,02 с, 0,1 с та 0,3 с. Це співвідношення дає фізичну інтерпретацію результатів: проміжний горизонт відповідає часовому вікну порядку десятих часток секунди, на якому керування виявляється найстабільнішим.

Потоки даних, частоти їх оновлення та інтерфейси для обох фаз зведено у таблиці 2.2.

Таблиця 2.2 — Потоки даних, частоти оновлення та інтерфейси системи

Потік даних	Напрям	Частота	Інтерфейс	Фаза
Відеопотік сцени	камера → обчислювальний вузол	~30 к/с	USB	1
Текстовий опис задачі	оператор → вузол	одноразово	—	1
Обчислення $W(h(o))$	обчислювальний вузол	одноразово	—	1
Завантаження матриці W	вузол → контролер РЧ	одноразово	USB / serial	1 → 2
Команда джойстика	джойстик → контролер РЧ	~50 Гц	АЦП / USB-HID	2
Керуючі команди суглобам	контролер РЧ → сервоприводи	~50 Гц	RS-485 (U2D2)	2
Пропріоцепція (положення)	сервоприводи → контролер РЧ	~50 Гц	RS-485 (U2D2)	2
Внутрішній ПІД-контур	сервопривід (локально)	вище за 50 Гц	вбудований	2

2.5 Структурна та функціональна схеми

На основі викладеного сформульовано дві схеми, призначені для перенесення на креслення формату А1.

Структурна схема (А1, аркуш 1). Структурну схему системи як САК зі зворотним зв’язком подано на рисунку 2.1. Вона відображає логіку керування у термінах ТАК: проходження сигналу від задавального впливу

оператора через задавальний пристрій і регулятор $W(h(o))$ до об'єкта керування — маніпулятора ViperX — та замикання головного зворотного зв'язку через пропріоцептивні датчики. На кресленні А1 цю схему доцільно доповнити позначеннями сигналів (a, $\dot{x}_{\text{cmd}}$, x) уздовж відповідних зв'язків, явним позначенням точки прикладання збурень до об'єкта та винесенням каскадної структури (рисунок 2.2) як деталізації блоку «об'єкт керування». Саме ця схема відображає назву роботи й виноситься на захист як основний результат розділу.

Функціональна схема (А1, аркуш 2). Функціональну схему, що відображає фізичну реалізацію та потоки даних, подано на рисунку 2.4. На відміну від структурної, вона показує не логіку регулювання, а апаратно-програмні вузли: камеру, обчислювальний вузол (Фаза 1), контролер реального часу з дискретизованим законом (2.2), шину DYNAMIXEL та сервоприводи з підпорядкованими ПІД-контурами. Поділ на фази (рисунок 2.3) відображено різним типом ліній: одноразові потоки ініціалізації відокремлено від циклічних потоків реального часу.

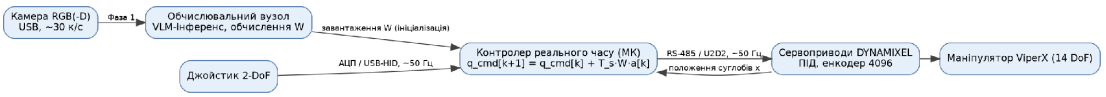


Рисунок 2.4 — Функціональна схема системи: потоки даних і реалізація на рівні вбудованої системи

Разом структурна та функціональна схеми утворюють несуперечливе подання системи на двох рівнях абстракції. Структурна схема обґрунтовує коректність контуру керування з позицій ТАК — наявність регулятора, об'єкта, вимірювального елемента та замкненого зворотного зв'язку. Функціональна схема демонструє фізичну реалізовність цього контуру на доступній елементній базі — смарт-сервоприводах DYNAMIXEL, шині RS-485 та мікроконтролерному обчислювачі — із дотриманням частотних вимог (2.3). Решта аркушів графічної частини (блок-схеми алгоритмів

формування та реконструкції відображення, зведення результатів) формуються у наступних розділах.

Висновки до розділу 2

У розділі розроблено структурну схему напівавтоматизованої системи відеомовного телеоперування маніпуляторами ViperX, що становить центральний результат роботи. Систему подано як САК зі зворотним зв’язком та обґрунтовано її напівавтоматизований характер: оператор формує низькорозмірний намір, тоді як автоматичне керуюче відображення бере на себе координацію високорозмірного руху, що відповідає усталеній у робототехніці концепції спільного керування.

Виокремлено функціональні блоки системи й поставлено кожному з них у відповідність елемент ТАК: оператор — джерело задавального впливу, джойстик — задавальний пристрій, індуковане відображення $W(h(o))$ — регулятор, маніпулятор ViperX — об’єкт керування, пропріоцептивні датчики — вимірювальний елемент, а генералістична політика — блок одноразового формування параметрів регулятора. Показано, що об’єкт керування охоплює підпорядкований сервопривідний контур, унаслідок чого система має каскадну двоконтурну структуру.

Обґрунтовано двофазну організацію роботи системи: обчислювально витратне формування керуючого відображення виконується одноразово на етапі ініціалізації, а керування в реальному часі зводиться до дешевої лінійної операції, придатної до виконання на мікроконтролері. Записано дискретизований закон керування (2.2)–(2.5), визначено вимоги до частоти дискретизації та встановлено зв’язок між горизонтом дій і фізичним часом керування.

Сформовано структурну та функціональну схеми системи, призначені для аркушів А1 графічної частини. Структурна схема обґрунтовує коректність замкненого контуру керування, а функціональна — його реалізовність на доступній елементній базі з дотриманням частотних вимог. Розроблені схеми

створюють основу для математичного обґрунтування способу формування регулятора $W(h(o))$, якому присвячено наступний розділ.

РОЗДІЛ 3

МАТЕМАТИЧНА МОДЕЛЬ ТА МЕТОД ІНДУКУВАННЯ КЕРУЮЧОГО ВІДОБРАЖЕННЯ

У цьому розділі побудовано математичну модель запропонованого способу формування керуючого відображення. Спочатку формалізуються спостереження та представлення задачі, далі вводиться локально лінійна модель динаміки керування, після чого керуюче відображення пов'язується із задача-обумовленою коваріацією передбачених дій. Центральним результатом розділу є спектральний аналіз збурень регуляризованої матриці, що обґрунтовує часову стабільність індукованого керування, та явна формула індукування відображення через спектр коваріації.

3.1 Формалізація спостережень і представлення задачі

Нехай задача телеоперування визначається мультимодальним спостереженням $o \in O$, що складається з візуальної та текстової складових:

$$o = (o_{\text{image}}, o_{\text{text}}) \in O = O_{\text{image}} \times O_{\text{text}}, \qquad (3.1)$$

де o_{image} — відеокадр (або послідовність кадрів) сцени; o_{text} — текстовий опис задачі (наприклад, «налити рідину»).

Спостереження відображається у векторизоване представлення задачі за допомогою відображення

$$h : O \to T \subseteq \mathbb{R}^n, \qquad t = h(o), \qquad (3.2)$$

де t — векторне вкладення задачі; $n \in \mathbb{N}_+$ — розмірність простору представлень. Відображення h реалізується кодувальною частиною генералістичної політики (Розділ 4). Сукупність таких відображень, що відповідають різним генералістичним представленням, позначимо $\mathcal{H} = \{h\}$. Принциповою для подальшого є та обставина, що h не навчається спеціально під конкретну задачу телеоперування, а береться вже сформованим у попередньо навченій моделі; саме це знімає потребу у задача-специфічному зборі даних, обговорену в підрозділі 1.3.

3.2 Модель динаміки керування та її локальна лінеаризація

Зв'язок між входом оператора та рухом робота у загальному вигляді описується керованою динамікою

$$\dot{x} = f(h(o),\, x,\, a), \qquad (3.3)$$

де x — стан робота; a — вхід оператора; f — функція динаміки керування, обумовлена представленням задачі $h(o)$.

Припустимо, що поблизу робочого режиму, за фіксованого представлення задачі та слабкої залежності від стану, функція f допускає лінійне наближення за входом оператора. Оскільки відсутність команди має відповідати відсутності руху, тобто $f(h(o), x, 0) = 0$, розклад Тейлора за a навколо нуля у першому порядку дає

$$f(h(o),\, x,\, a) \approx W(h(o))\, a, \qquad (3.4)$$

де $W(h(o))$ — задача-обумовлене лінійне керуюче відображення, що відіграє роль регулятора у структурній схемі (рисунок 2.1). Залежність від стану x після ідентифікації сцени вважається повільною і поглинається обумовленням; це узгоджується з постановкою задачі, за якою контролер спеціалізується переважно за пропріоцептивним входом.

Розглянемо спершу ілюстративний випадок двовимірного контролера, відображеного на восьмиступеневу систему:

$$\dot{x} = W(h(o))\, a, \qquad \dot{x} \in \mathbb{R}^8,\ a \in \mathbb{R}^2,\ W(h(o)) \in \mathbb{R}^{8\times 2}. \qquad (3.5)$$

Для цільової біманіпуляційної платформи ViperX простір дій має розмірність $d = 14$, а вхід оператора лишається двовимірним ($k = 2$), тож $\dot{x} \in \mathbb{R}^{14}$, $a \in \mathbb{R}^2$, $W(h(o)) \in \mathbb{R}^{14\times 2}$. Розмірності всіх величин моделі для цільового випадку зведено в таблиці 3.1.

Таблиця 3.1 — Розмірності об'єктів математичної моделі (цільовий випадок ViperX)

Об'єкт	Позначення	Розмірність
Спостереження	o	мультимодальне (зображення + текст)
Представлення задачі	$t = h(o)$	$\mathbb{R}^n$ (n — розмір вкладення політики)
Команда оператора	a	$\mathbb{R}^2$ ($k = 2$)
Швидкості/дії робота	$\dot{x}$	$\mathbb{R}^{14}$ ($d = 14$)
Керуюче відображення	$W(h(o))$	$\mathbb{R}^{14\times 2}$
Матриця даних дій	$H(t, x)$	$\mathbb{R}^{N\times 14}$ ($N = 128$ семплів)
Коваріація дій	Σ_a	$\mathbb{R}^{14\times 14}$
Власні вектори / значення	$U,\ \Lambda$	$\mathbb{R}^{14\times 14}$
Керуючий базис	$B = U_k \Lambda_k^{1/2}$	$\mathbb{R}^{14\times 2}$

Слід звернути увагу на геометричний зміст відображення $W(h(o))$. Оскільки воно діє з простору $\mathbb{R}^k$ у простір $\mathbb{R}^d$ при $k \ll d$, множина миттєво досяжних напрямків руху — це образ $\mathrm{range}(W)$, підпростір розмірності не більше k. Інакше кажучи, у кожний момент керування зосереджене у щонайбільше двовимірній площині простору дій. Якість керування визначається тим, наскільки вдало ця площина зорієнтована відносно напрямків, потрібних для виконання задачі. Саме обґрунтуванню вибору цієї площини й присвячено решту розділу.

3.3 Задача-обумовлена коваріація передбачених дій

Ключова ідея запропонованого методу полягає в тому, щоб не передбачати матрицю $W(h(o))$ безпосередньо, а вивести її зі статистики дій, що їх породжує генералістична політика для даної задачі. Модель навчається передбачати задача-обумовлений розподіл дій з нульовим середнім і навченою коваріацією (деталі реалізації — у Розділі 4); семплювання з цього розподілу або стекування передбачених дій формує емпіричну вибірку.

Складемо матрицю даних, рядками якої є N передбачених векторів дій для заданих t і x:

$$H(t,x) = \left[\, a^{(1)},\, a^{(2)},\, \ldots,\, a^{(N)} \,\right]^{\top} \in \mathbb{R}^{N\times d}, \qquad (3.6)$$

де $a^{(i)} \in \mathbb{R}^d$ — i-й передбачений вектор дії; N — кількість семплів (у дослідженні $N = 128$ на крок). Утворимо матрицю Грама цієї вибірки:

$$G(t,x) = H(t,x)^{\top} H(t,x) \in \mathbb{R}^{d\times d}. \qquad (3.7)$$

За умови центрованості вибірки (нульове середнє дій) нормована матриця Грама збігається з емпіричною коваріаційною матрицею дій:

$$\hat{\Sigma}_a = \frac{1}{N}\sum_{i=1}^{N} a^{(i)} a^{(i)\top} = \frac{1}{N}\, H(t,x)^{\top} H(t,x). \qquad (3.8)$$

Для забезпечення числової стійкості додамо регуляризацію. Тоді з точністю до масштабного множника $1/N$ виконується відповідність

$$H(t,x)^{\top} H(t,x) + \sigma I \;\longleftrightarrow\; \Sigma_a + \sigma I, \qquad (3.9)$$

де Σ_a — емпірична коваріаційна матриця передбачених дій; $\sigma > 0$ — параметр регуляризації; I — одинична матриця. Таким чином, регуляризована матриця Грама передбачених дій інтерпретується як оцінка коваріації дій, зміщена на σI. Позначимо цю регуляризовану матрицю

$$M = \Sigma_a + \sigma I. \qquad (3.10)$$

Матриця M симетрична й додатно визначена: усі її власні значення не менші за $\sigma > 0$, тож $M \succ 0$ і є оборотною. Ця властивість далі забезпечує коректність задачі реконструкції (підрозділ 3.5) та обмежує число обумовленості.

3.4 Спектральний аналіз збурень регуляризованої матриці

Розглянемо спектральне розкладання коваріації дій:

$$\Sigma_a = U \Lambda U^\top, \qquad \Lambda = \mathrm{diag}(\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0), \qquad (3.11)$$

де $U = [u_1, u_2, \dots, u_d]$ — ортогональна матриця власних векторів; λ_i — впорядковані власні значення. Через зсув на σI матриця M має ті самі власні вектори, що й Σ_a, а її власні значення дорівнюють $\lambda_i + \sigma$. Власні вектори u_i задають напрямки головних компонент дій, а власні значення λ_i — дисперсії вздовж цих напрямків.

З'ясуємо, як змінюється ця спектральна структура за малих збурень, спричинених зміною задачі чи стану, а також скінченністю вибірки. Нехай Σ_a зазнає малого симетричного збурення E, тобто розглядається $\Sigma_a + E$ (або $\Sigma_a + \varepsilon E$ для аналізу першого порядку).

Стійкість власних значень. Згідно з нерівністю Вейля [21], [22], для симетричних матриць власні значення є ліпшицевими з константою одиниця відносно спектральної норми збурення:

$$\left| \lambda_i(\Sigma_a + E) - \lambda_i(\Sigma_a) \right| \leq \|E\|_2, \qquad i = 1, \dots, d. \qquad (3.12)$$

Отже, власні значення — а з ними і підсилення керуючих осей — змінюються не швидше за норму збурення й не виявляють стрибків.

Перший порядок для власних значень. Для простого власного значення λ_i з одиничним власним вектором u_i перше наближення збуреного власного значення має вигляд

$$\lambda_i(\varepsilon) = \lambda_i + \varepsilon\, u_i^\top E\, u_i + O(\varepsilon^2). \qquad (3.13)$$

Перший порядок для власних векторів. Відповідне перше наближення збуреного власного вектора

$$u_i(\varepsilon) = u_i + \varepsilon \sum_{j \neq i} \frac{u_j^\top E\, u_i}{\lambda_i - \lambda_j}\, u_j + O(\varepsilon^2). \qquad (3.14)$$

Вираз (3.14) має вирішальне значення для подальших висновків: чутливість власного вектора обернено пропорційна різницям власних значень $\lambda_i - \lambda_j$. Якщо власне значення λ_i близьке до сусіднього, відповідний знаменник малий, і навіть незначне збурення E спричиняє велике обертання власного вектора. Навпаки, добре відокремлене власне значення відповідає стабільному власному напрямку.

Формалізуємо поняття відокремленості. Введемо власну щілину для виокремлення k провідних компонент:

$$\delta = \lambda_k - \lambda_{k+1}. \qquad (3.15)$$

Поворот провідного k-вимірного власного підпростору під дією збурення обмежується теоремою Девіса–Кагана про синус кута [23]:

$$\left\| \sin \Theta(\tilde{U}_k,\, U_k) \right\| \leq \frac{\|E\|_2}{\delta}, \qquad (3.16)$$

де $U_k = [u_1, \dots, u_k]$ та $\tilde{U}_k$ — матриці провідних k власних векторів незбуреної та збуреної матриць відповідно; $\Theta(\cdot,\cdot)$ — діагональна матриця головних кутів між підпросторами. Статистичний варіант цієї оцінки, зручний для випадку емпіричних (вибіркових) коваріацій, наведено в [24]. Оцінка (3.16) лінійно зростає з нормою збурення й обернено пропорційна власній щілині: чим краще відокремлений провідний підпростір, тим стабільнішим він є щодо збурень.

Роль регуляризації. Параметр σ підвищує «підлогу» спектра, оскільки всі власні значення M не менші за σ. Це обмежує число обумовленості регуляризованої матриці:

$$\kappa(M) = \frac{\lambda_1 + \sigma}{\lambda_d + \sigma} \leq \frac{\lambda_1 + \sigma}{\sigma}. \qquad (3.17)$$

Менше число обумовленості означає кращу числову робастність обчислень із M та коректність оберненої задачі реконструкції. Отже, регуляризація відіграє двоїсту роль: гарантує додатну визначеність (а отже, оборотність) і обмежує підсилення похибок.

Зведемо результати аналізу. Власні значення стабільні безумовно (3.12); власні вектори та провідний підпростір стабільні за умови достатньої власної щілини (3.14), (3.16); регуляризація додатково покращує обумовленість (3.17). Звідси випливає головний якісний висновок: **коли спектр коваріації дій добре розділений, провідні керуючі напрямки залишаються майже незмінними при змінах задачі чи стану, а отже, ефективне зчеплення між входом оператора та діями робота еволюціонує плавно, а не стрибкоподібно.** Спектральна розділеність діє як стабілізувальний чинник індукованого керування, що безпосередньо відповідає вимозі часової стабільності, поставленій у Розділі 2.

3.5 Індукування керуючого відображення через спектр коваріації

Маючи спектральне розкладання (3.11), сформуємо керуюче відображення з провідних k компонент. Визначимо низькоранговий керуючий базис

$$B = U_k \Lambda_k^{1/2} \in \mathbb{R}^{d \times k}, \qquad U_k = [u_1, \dots, u_k],\ \Lambda_k = \mathsf{diag}(\lambda_1, \dots, \lambda_k),$$

де k дорівнює кількості ступенів вільності користувацького контролера (для ViperX $k = 2$). Цей базис і приймається за індуковане керуюче відображення, $W(h(o)) := B$. Тоді закон керування (3.4) набуває вигляду

$$\dot{x}_{\text{cmd}} = W(h(o))\, a = U_k\, \Lambda_k^{1/2}\, a = \sum_{i=1}^{k} \sqrt{\lambda_i}\, a_i\, u_i, \qquad (3.19)$$

де $a = (a_1, \dots, a_k)^\top$ — команда оператора. Вираз (3.19) розкриває внутрішню будову індукованого відображення. Геометрія керування задається ортонормованими власними векторами u_i — головними напрямками дій: вони визначають, у який бік простору $\mathbb{R}^{14}$ рухатиметься маніпулятор у відповідь на кожну вісь джойстика. Підсилення керування задається коренями власних значень $\sqrt{\lambda_i}$ — стандартними відхиленнями вздовж цих напрямків: осі з більшою природною мінливістю дій отримують більший коефіцієнт передачі. Таким чином, і геометрія, і сила відображення «оператор → робот» закодовані неявно в коваріації Σ_a, без явного передбачення матриці W.

Рисунок 3.1 — Схема індукування керуючого відображення через спектр коваріації передбачених дій

Геометричну інтерпретацію подано на рисунку 3.2. Коваріація Σ_a задає в просторі дій $\mathbb{R}^{14}$ еліпсоїд розсіювання, головні осі якого спрямовані вздовж власних векторів u_i, а півосі пропорційні $\sqrt{\lambda_i}$. Індуковане відображення проєктує двовимірний вхід оператора на площину, натягнуту на два найдовші головні напрямки u_1, u_2 цього еліпсоїда. Інакше кажучи, керування автоматично зорієнтовується вздовж найбільш «виразних» напрямків дій, властивих задачі, а менш істотні напрямки відкидаються разом із відповідними малими власними значеннями.

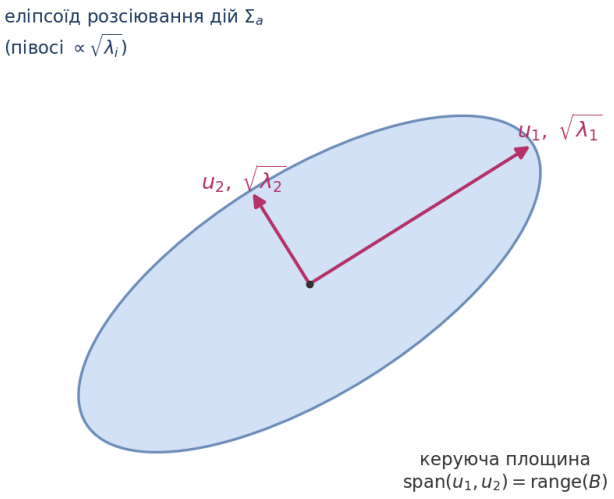


Рисунок 3.2 — Геометрична інтерпретація: коваріаційний еліпсоїд дій і керуюча площина, на яку відображається вхід оператора

Нарешті, зв'яжемо індуковане відображення з оцінюванням якості, що проводиться в Розділі 5. Оскільки керування зосереджене у підпросторі $\mathrm{range}(B)$ розмірності k, природним критерієм достатності цього підпростору є те, наскільки добре він покриває реальні (експертні) дії. Для еталонної дії a^* найкраще наближення в межах керуючого базису знаходять розв'язанням задачі найменших квадратів, $\hat{a} = B\,z^*$, $z^* = \arg\min_z \|a^* - Bz\|_2^2$; величина залишку $\|a^* - \hat{a}\|_2$ і слугує мірою того, наскільки індукована геометрія керування здатна відтворити експертну поведінку. Додатна визначеність M, забезпечена регуляризацією (3.17), гарантує коректність цієї оберненої задачі. Деталі обчислення B та реконструкції винесено в Розділ 4, а кількісні результати — у Розділ 5.

Висновки до розділу 3

У розділі побудовано математичну модель індукування керуючого відображення. Формалізовано мультимодальні спостереження та представлення задачі (3.1), (3.2), уведено локально лінійну модель динаміки керування (3.3)–(3.5) і встановлено її геометричний зміст: керування

зосереджене у щонайбільше k-вимірному підпросторі простору дій, орієнтація якого визначає якість системи.

Обґрунтовано підхід, за якого відображення виводиться не прямим передбаченням, а зі статистики передбачених дій: матрицю даних дій (3.6) зведено до регуляризованої матриці Грама, що інтерпретується як зміщена оцінка коваріації дій (3.7)–(3.10). Доведено додатну визначеність регуляризованої матриці, що забезпечує коректність подальшої оберненої задачі.

Виконано спектральний аналіз збурень. Показано, що власні значення стабільні безумовно за нерівністю Вейля (3.12), а власні вектори та провідний підпростір стабільні за умови достатньої власної щілини, що формалізовано першими наближеннями (3.13), (3.14) та теоремою Девіса–Кагана (3.15), (3.16). Установлено, що регуляризація обмежує число обумовленості (3.17). Звідси випливає, що спектральна розділеність діє як стабілізувальний чинник, забезпечуючи плавну, часово стабільну еволюцію керування.

Отримано явну формулу індукування відображення через спектр коваріації (3.18), (3.19) та показано, що геометрію керування задають провідні власні вектори, а підсилення — корені відповідних власних значень, тож обидві характеристики відображення «оператор → робот» закодовані неявно в коваріації дій. Сформульовано критерій достатності індукованого підпростору через похибку лінійної реконструкції експертних дій, що становить основу експериментального оцінювання в Розділі 5. Програмну реалізацію описаних обчислень розглянуто в наступному розділі.

РОЗДІЛ 4

ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АЛГОРИТМИ

Цей розділ описує програмну реалізацію запропонованого методу. Спершу розглянуто архітектуру базової генералістичної моделі Octo та спосіб отримання представлення задачі, далі — власну головку телеоперування, що передбачає коваріацію дій, та її навчання за критерієм максимальної правдоподібності. Після цього подано основні алгоритми — навчання головки, формування коваріації, індукування керуючого базису та детермінованої лінійної реконструкції — у вигляді псевдокоду й фрагментів коду мовою Python із застосуванням бібліотеки PyTorch [25]. Завершують розділ блок-схеми алгоритмів, призначені для аркушів графічної частини. Повний код винесено в додатки.

4.1 Архітектура базової моделі Octo та отримання представлення задачі

Базовою генералістичною політикою обрано модель Octo [2] — трансформерну архітектуру, попередньо навчену на 800 тисячах робототехнічних траєкторій із набору Open X-Embodiment [14]. Модель опрацьовує мультимодальні спостереження за принципом токенізації: текстова інструкція перетворюється на послідовність токенів мовним кодувальником, зображення — на токени згортковими блоками, а пропріоцептивний стан — на окремий токен. До послідовності входів додається спеціальний *зчитувальний токен* (readout token), що не впливає на інші токени, але акумулює контекст для головок дій. Трансформерний кістяк із блоковою маскою уваги обробляє цю послідовність; вихідне вкладення зчитувального токена і слугує представленням задачі $t = h(o)$ у сенсі відображення (3.2).

Штатно Octo завершується дифузійною головкою дій, що моделює багатомодальний розподіл дій через ітеративне зашумлення–очищення [26].

У запропонованому підході ця головка замінюється власною головкою телеоперування (підрозділ 4.2), яка передбачає не самі дії, а коваріаційну структуру їх розподілу. Решта моделі — токенізатори та трансформерний кістяк — лишаються від попередньо навченої Octo, а донавчання зачіпає передусім нову головку. Така стратегія узгоджується з призначенням Octo як гнучкої ініціалізації, придатної до донавчання на новий простір дій за кілька годин на споживчому обладнанні [2]. Загальну архітектуру з власною головкою подано на рисунку 4.1.

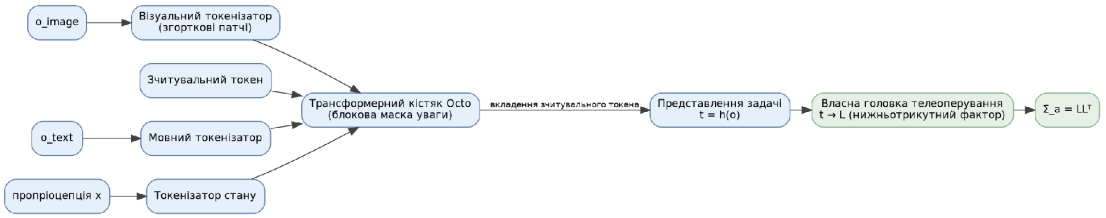


Рисунок 4.1 — Архітектура системи: попередньо навчений кістяк Octo з власною головкою телеоперування замість штатної дифузійної головки дій

4.2 Власна головка телеоперування

Головка телеоперування передбачає задача-обумовлений гаусів розподіл дій із фіксованим нульовим середнім і навченою коваріаційною матрицею як функцією представлення задачі:

$$a \sim \mathcal{N}(0,\ \Sigma_a(t)), \qquad t = h(o). \qquad (4.1)$$

Фіксація середнього нулем є принциповою: метою є не передбачити конкретну дію, а охарактеризувати геометрію та масштаб варіативності дій, тобто ту структуру, з якої згодом індукується керуюче відображення (підрозділ 3.5). Уся корисна інформація зосереджується в коваріації $\Sigma_a(t)$.

Щоб гарантувати симетричність і додатну визначеність передбаченої коваріації, головка передбачає не саму матрицю Σ_a, а її нижньотрикутний фактор Холеського:

$$\Sigma_a(t) = L(t)\, L(t)^\top, \qquad (4.2)$$

де $L(t)$ — нижньотрикутна матриця з додатними діагональними елементами. Така параметризація має дві переваги. По-перше, добуток $LL^\top$ є додатно напіввизначеним за побудовою, а додатність діагоналі L робить його строго додатно визначеним, що узгоджується з вимогою оборотності з підрозділу 3.3. По-друге, для гаусового розподілу, заданого фактором L (нижнім трикутником масштабу), від'ємний логарифм правдоподібності та його градієнти обчислюються стійко без явного обернення матриці.

Навчання відбувається мінімізацією від'ємного логарифма правдоподібності (NLL) еталонних дій під передбаченим розподілом. Для дії a^* під розподілом $\mathcal{N}(0, \Sigma_a)$

$$-\log p(a^*) = \tfrac{1}{2}\Big(a^{*\top}\Sigma_a^{-1}a^* + \log\det\Sigma_a + d\log 2\pi\Big). \qquad (4.3)$$

Відповідно до постановки, модель передбачає одну коваріаційну матрицю, спільну для всього горизонту дій довжиною H, а не окрему дію на кожному кроці. Тому функція втрат усереднює NLL за всіма кроками горизонту та елементами батча:

$$\mathcal{L} = \frac{1}{B_{\text{batch}} H} \sum_{b=1}^{B_{\text{batch}}} \sum_{\tau=1}^{H} \Big[-\log \mathcal{N}(a^*_{b,\tau};\, 0,\, \Sigma_a(t_b)) \Big], \qquad (4.4)$$

де B_{batch} — розмір батча; H — горизонт дій; $a^*_{b,\tau}$ — еталонна дія на τ-му кроці горизонту для b-го елемента батча; $t_b = h(o_b)$ — відповідне представлення задачі. Спільність коваріації для горизонту означає, що модель змушена знайти єдину структуру розсіювання, яка добре пояснює всю послідовність дій, а не підлаштовується покроково; як показано в Розділі 5, саме помірний горизонт дає найкращий компроміс.

Гіперпараметри головки та процедури навчання зведено в таблиці 4.1.

Таблиця 4.1 — Гіперпараметри головки телеоперування та навчання

Параметр	Значення
Базова модель	Octo (донавчання головки)
Представлення задачі t	вкладення зчитувального токена
Розмірність дій d	14
Ступені вільності контролера k	2
Параметризація коваріації	фактор Холеського L, $\Sigma_a = LL^\top$
Середнє розподілу	фіксоване, 0
Функція втрат	від'ємна лог-правдоподібність (NLL)
Горизонти дій H	1, 5, 15 (три окремі моделі)
Кількість кроків оптимізації	2000
Розмір батча B_{batch}	64
Семплів на крок (оцінювання)	$N = 128$
Регуляризація σ	мала додатна (напр., 10^{-3})

Програмну реалізацію головки наведено нижче. Лінійний шар відображає представлення задачі у $d(d+1)/2$ параметрів нижнього трикутника; діагональ пропускається через функцію `softplus` для забезпечення додатності.

```
import torch
import torch.nn as nn
import torch.nn.functional as F


class TeleopCovHead(nn.Module):
    """Передбачає нижньотрикутний фактор Холеського L,
    звідки коваріація дій Σ_a = L L^T (середнє фіксоване, нуль)."""
    def __init__(self, task_dim: int, action_dim: int, hidden: int = 512):
        super().__init__()
        self.d = action_dim
        n_tril = action_dim * (action_dim + 1) // 2
        self.net = nn.Sequential(
            nn.Linear(task_dim, hidden), nn.GELU(),
            nn.Linear(hidden, hidden), nn.GELU(),
            nn.Linear(hidden, n_tril),
        )
        tril = torch.tril_indices(action_dim, action_dim)
        self.register_buffer("ti_row", tril[0])
        self.register_buffer("ti_col", tril[1])

    def forward(self, t: torch.Tensor) -> torch.Tensor:   # t: (B, task_dim)
        B = t.shape[0]
        raw = self.net(t)                                  # (B, n_tril)
        L = t.new_zeros(B, self.d, self.d)
        L[:, self.ti_row, self.ti_col] = raw
        diag = torch.arange(self.d, device=t.device)
        # додатна діагональ -> коректний фактор Холеського
        L[:, diag, diag] = F.softplus(L[:, diag, diag]) + 1e-4
        return L                                           # scale_tril
```

4.3 Алгоритми

4.3.1 Навчання головки телеоперування

Навчання реалізує мінімізацію втрати (4.4). Завдяки параметризації через фактор Холеського NLL обчислюється безпосередньо вбудованими засобами багатовимірного нормального розподілу.

Алгоритм 4.1 — Навчання головки телеоперування

```
Вхід:  набір даних D = {(o, a*₁,…,a*_H)}; горизонт H; кроки T; крок навчання η
Вихід: навчені параметри головки θ
1.  ініціалізувати θ; заморозити (або частково) кістяк Octo
2.  для кроку = 1 … T:
3.      вибрати батч {(oβ, a*_{b,1..H})} розміром B_batch
4.      tβ ← h(oβ)                              // представлення задачі (Octo)
5.      Lβ ← головка_θ(tβ)                      // фактор Холеського
6.      ℒ ← (1/(B_batch·H)) Σβ Σ_τ [ -log 𝒩(a*_{b,τ}; 0, Lβ Lβ^T) ]
7.      θ ← θ − η · ∇_θ ℒ                       // крок оптимізатора
8.  повернути θ
```

```
from torch.distributions import MultivariateNormal

def nll_loss(L: torch.Tensor, actions: torch.Tensor) -> torch.Tensor:
    """L: (B, d, d) фактор Холеського; actions: (B, H, d) еталонні дії горизонту.
    Одна коваріація на горизонт -> NLL усереднюється за H кроками."""
    B, H, d = actions.shape
    dist = MultivariateNormal(loc=torch.zeros(B, d, device=L.device),
                              scale_tril=L)
    logp = dist.log_prob(actions.permute(1, 0, 2))    # (H, B): спільний розподіл
    return -logp.mean()

# крок навчання
opt = torch.optim.AdamW(head.parameters(), lr=3e-4)
for step in range(2000):
    obs, gt_actions = sample_batch(dataset, batch_size=64, horizon=H)
    t = octo_encode(obs)            # представлення задачі t = h(o)
    L = head(t)
    loss = nll_loss(L, gt_actions)
    opt.zero_grad(); loss.backward(); opt.step()
```

4.3.2 Формування коваріації дій із семплів

Під час оцінювання для кожного кроку семплюється $N = 128$ дій із передбаченого розподілу, після чого обчислюється емпірична коваріація (3.8) з регуляризацією (3.9).

Алгоритм 4.2 — Формування регуляризованої коваріації дій

```
Вхід:  фактор L = head(t); кількість семплів N; регуляризація σ
Вихід: M = Σ^_a + σI
1.  сформувати розподіл 𝒩(0, L L^T)
2.  семплювати {a^(1),…,a^(N)},  a^(i) ~ 𝒩(0, L L^T)
3.  центрувати: ā ← (1/N) Σ a^(i);  a^(i) ← a^(i) − ā
4.  Σ^_a ← (1/N) Σ a^(i) a^(i)T
5.  M ← Σ^_a + σI
6.  повернути M
```

```
@torch.no_grad()
def empirical_covariance(L, n_samples=128, sigma=1e-3):
```

```
    B, d, _ = L.shape
    dist = MultivariateNormal(loc=torch.zeros(B, d, device=L.device),
                              scale_tril=L)
    s = dist.sample((n_samples,))              # (N, B, d)
    s = s - s.mean(dim=0, keepdim=True)        # центрування
    Sigma = torch.einsum('nbd,nbe->bde', s, s) / n_samples   # (B, d, d)
    eye = torch.eye(d, device=L.device)
    return Sigma + sigma * eye                 # регуляризована матриця M
```

4.3.3 Індукування керуючого базису

Керуючий базис формується спектральним розкладанням коваріації (3.11) та відбором провідних k компонент згідно з (3.18).

Алгоритм 4.3 — Індукування керуючого базису

```
Вхід:  регуляризована коваріація M; ступені вільності контролера k
Вихід: керуючий базис B = U_k Λ_k^{1/2}  (= W(h(o)))
1.  [U, Λ] ← eigh(M)                          // власні значення/вектори, зростання
2.  впорядкувати спадно: λ₁ ≥ … ≥ λ_d
3.  U_k ← перші k власних векторів;  Λ_k ← перші k власних значень
4.  B ← U_k · diag(√λ₁,…,√λ_k)
5.  повернути B
```

```
@torch.no_grad()
def induce_control_basis(M, k=2):
    evals, evecs = torch.linalg.eigh(M)        # зростання
    evals = evals.flip(-1)                     # спадання
    evecs = evecs.flip(-1)
    Uk = evecs[..., :k]                        # (B, d, k)
    Lk = evals[..., :k].clamp_min(0.0).sqrt()  # (B, k): підсилення √λ
    B_basis = Uk * Lk.unsqueeze(-2)            # (B, d, k): стовпці масштабовано
    return B_basis                             # = W(h(o))
```

4.3.4 Детермінована лінійна реконструкція

Виразність індукованого базису оцінюється тим, наскільки точно він відтворює еталонні дії. Для дії a^* розв'язується задача найменших квадратів (відповідно до критерію, окресленого в підрозділі 3.5):

$$z^* = \arg\min_{z \in \mathbb{R}^k} \|a^* - Bz\|_2^2 = (B^\top B)^{-1} B^\top a^*, \qquad \hat{a} = Bz^*, \qquad (4.5)$$

де z^* — оптимальні коефіцієнти в керуючому базисі; $\hat{a}$ — реконструйована дія. Похибка реконструкції $\|a^* - \hat{a}\|_2$ кількісно характеризує, наскільки еталонна дія лежить поза індукованою керуючою структурою.

Алгоритм 4.4 — Детермінована лінійна реконструкція

```
Вхід:  керуючий базис B ∈ ℝ^{d×k}; множина еталонних дій {a*}
Вихід: метрики реконструкції (MSE, MAE)
1.  для кожної дії a* у множині:
2.      z* ← argmin_z ‖a* − B z‖²            // найменші квадрати
```

```
3.      â  ← B z*
4.      зберегти похибку (a* − â)
5.  обчислити MSE та MAE за всіма діями
6.  повернути (MSE, MAE)
```

```python
@torch.no_grad()
def reconstruct(B_basis, a_star):
    # B_basis: (d, k); a_star: (d,)
    z = torch.linalg.lstsq(B_basis, a_star.unsqueeze(-1)).solution  # (k, 1)
    a_hat = (B_basis @ z).squeeze(-1)                               # (d,)
    return a_hat, z.squeeze(-1)


@torch.no_grad()
def reconstruction_metrics(B_basis, gt_actions):   # gt_actions: (M, d)
    errs = []
    for a_star in gt_actions:
        a_hat, _ = reconstruct(B_basis, a_star)
        errs.append(a_star - a_hat)
    E = torch.stack(errs)
    mse = (E ** 2).mean().item()
    mae = E.abs().mean().item()
    return mse, mae
```

4.4 Схеми алгоритмів

Для графічної частини алгоритми формування керуючого базису та реконструкції подано у вигляді блок-схем за ДСТУ (овал — початок/кінець, паралелограм — введення/виведення, прямокутник — процес, ромб — рішення). Блок-схему формування коваріації та індукування базису (об’єднує алгоритми 4.2 і 4.3) наведено на рисунку 4.2, блок-схему детермінованої реконструкції (алгоритм 4.4) — на рисунку 4.3.

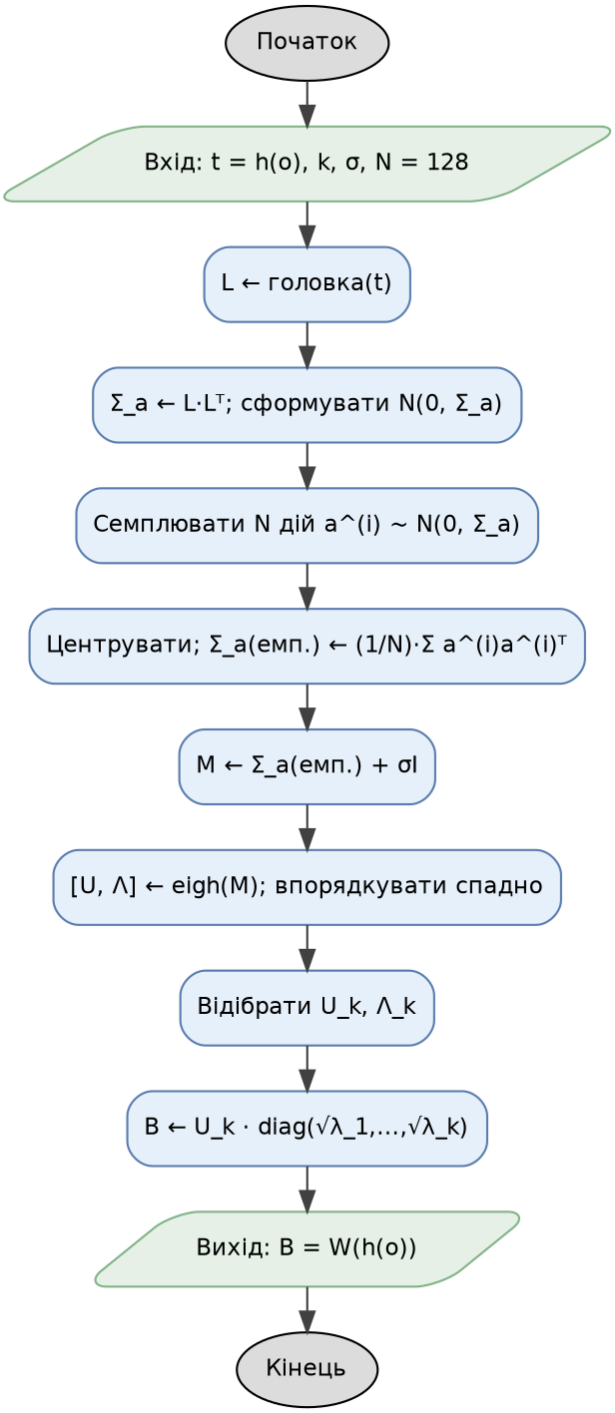


Рисунок 4.2 — Блок-схема алгоритму формування коваріації дій та індукування керуючого базису (аркуш А1 №3)

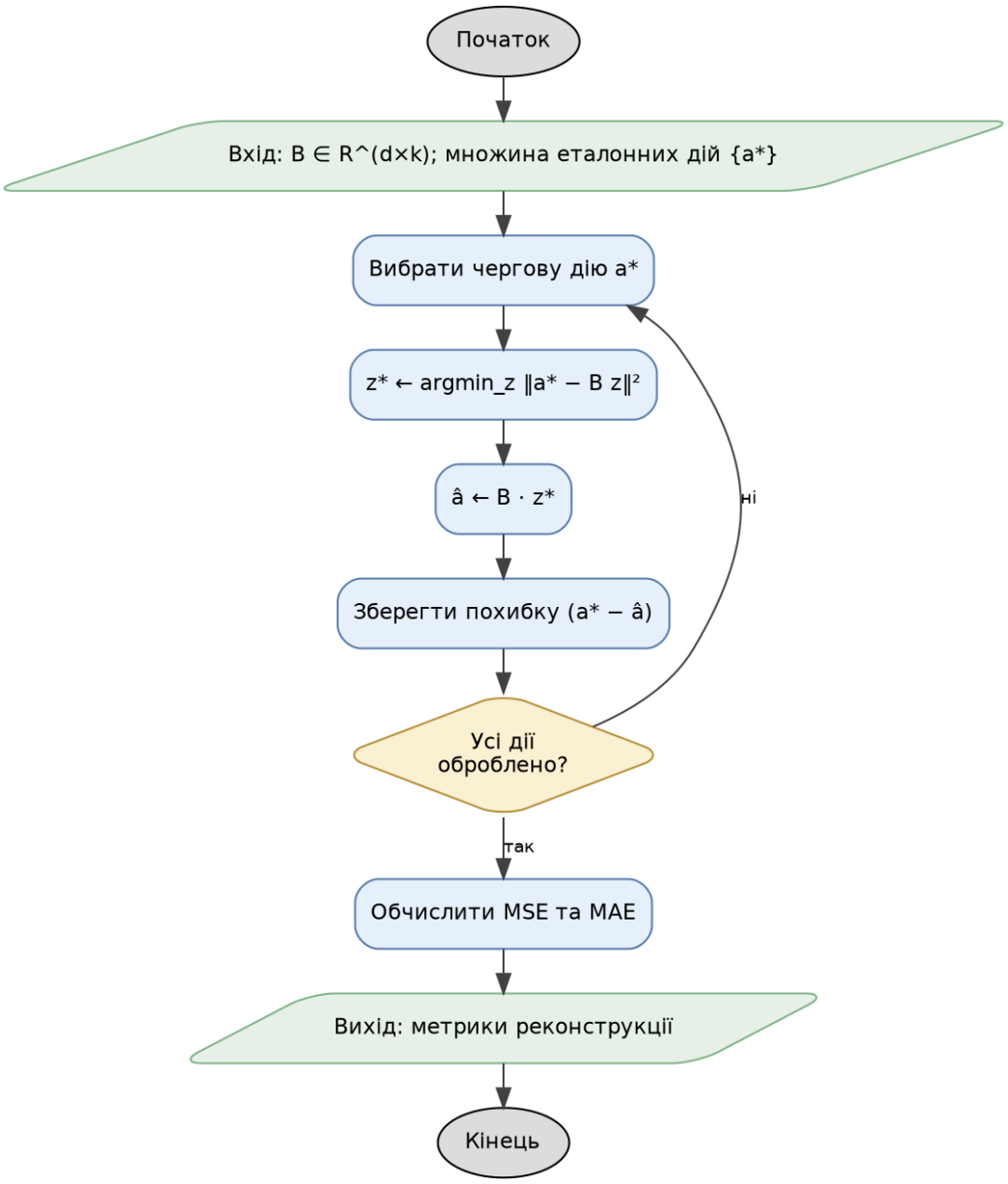


Рисунок 4.3 — Блок-схема алгоритму детермінованої лінійної реконструкції (аркуш А1 №4)

Висновки до розділу 4

У розділі описано програмну реалізацію запропонованого методу. Обґрунтовано вибір моделі Octo як базової генералістичної політики та показано, що представленням задачі $t = h(o)$ слугує вкладення зчитувального токена, а штатну дифузійну головку дій замінено власною головкою телеоперування.

Розроблено головку телеоперування, що передбачає задача-обумовлений гаусів розподіл дій із фіксованим нульовим середнім і навченою коваріацією (4.1). Для гарантування симетричності та додатної визначеності застосовано параметризацію коваріації через фактор Холеського (4.2), а навчання реалізовано мінімізацією NLL (4.3), усередненого за горизонтом дій (4.4) — відповідно до вимоги однієї коваріаційної матриці на горизонт.

Подано чотири основні алгоритми — навчання головки, формування регуляризованої коваріації зі 128 семплів на крок, індукування керуючого базису через спектральне розкладання та детермінованої лінійної реконструкції (4.5) — у вигляді псевдокоду й узгоджених із ними фрагментів коду PyTorch. Алгоритми формування базису та реконструкції оформлено як блок-схеми за ДСТУ (рисунки 4.2, 4.3), що утворюють аркуші А1 №3 і №4 графічної частини. Реалізовані алгоритми створюють програмну основу для експериментальних досліджень, викладених у наступному розділі.

РОЗДІЛ 5

ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ТА РЕЗУЛЬТАТИ

У цьому розділі викладено результати експериментального оцінювання запропонованого методу. Мета досліджень — перевірити, чи адекватно індуковане керуюче відображення покриває еталонні (експертні) дії та наскільки воно чутливе до різних горизонтів дій. Усі дослідження виконано у симуляції: жодного розгортання на фізичній платформі ViperX не здійснювалося, а використаний набір даних є симуляційним. Відтак отримані метрики характеризують виразність і часову стабільність індукованої керуючої структури, а не показники реальної фізичної системи.

5.1 Набір даних та протокол оцінювання

Для донавчання та оцінювання використано симуляційний набір даних `aloha_sim`, що містить траєкторії маніпуляції біманіпуляційної платформи ALOHA, опубліковані групою Berkeley RAIL [3], [14]. Простір дій набору відповідає чотирнадцяти керованим координатам платформи ($d = 14$), що узгоджується з цільовою конфігурацією ViperX (підрозділ 1.5).

Протокол оцінювання побудовано так, щоб перевірити крос-горизонтну генералізацію індукованого відображення. Для кожного експерименту відбирається три відкладені (held-out) траєкторії, які не використовувалися під час навчання. Для отримання надійної оцінки задача-обумовленої коваріації дій на кожному кроці семплюється $N = 128$ дій із передбаченого розподілу, після чого обчислюється емпірична коваріаційна матриця (алгоритм 4.2). З неї виводиться низькоранговий керуючий базис для детермінованої лінійної реконструкції (алгоритми 4.3, 4.4). Передбачення обробляються батчами розміром 64 для обчислювальної ефективності. Параметри експерименту зведено в таблиці 5.1.

Таблиця 5.1 — Параметри експериментального дослідження

Параметр	Значення
Набір даних	`aloha_sim` (Berkeley RAIL), симуляційний
Розмірність дій d	14
Ступені вільності контролера k	2
Горизонти навчання	1, 5, 15 кроків (три окремі моделі)
Горизонти прогнозування (оцінка)	1, 5, 15 кроків
Відкладених траєкторій на експеримент	3
Семплів дій на крок	128
Розмір батча обробки	64
Кроків оптимізації на модель	2000
Режим валідації	виключно симуляційний

Оскільки запропонований підхід індукує лінійне керуюче відображення через коваріацію передбачених дій, а не передбачає дії безпосередньо, оцінювання зосереджене на репрезентативній здатності та часовій

стабільності навченої керуючої структури, а не на поточковій точності передбачення окремих дій.

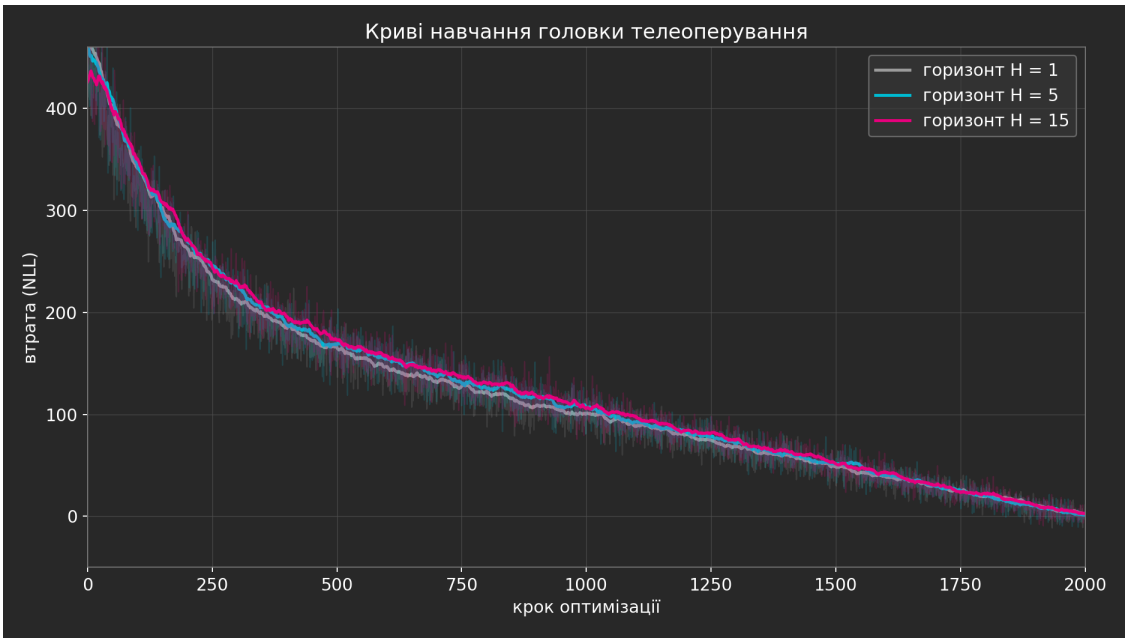
5.2 Налаштування навчання

Для оцінювання впливу часового масштабу донавчено три окремі моделі Octo з горизонтами дій 1, 5 та 15 кроків. Кожну модель навчено впродовж 2000 кроків оптимізації з розміром батча 64, мінімізацією від'ємного логарифма правдоподібності (4.4). Решта налаштувань (параметризація коваріації через фактор Холеського, фіксоване нульове середнє, спільна коваріація на горизонт) збігаються з описаними у Розділі 4.

Кожну з трьох навчених моделей оцінено на горизонтах прогнозування 1, 5 та 15 кроків, що утворює сітку пар «горизонт навчання — горизонт прогнозування» розміром 3×3. Така сітка дозволяє відокремити вплив горизонту, на якому модель навчалася, від горизонту, на якому вона застосовується, і таким чином оцінити перенесення (генералізацію) між горизонтами.

5.3 Криві навчання

Криві навчання трьох моделей наведено на рисунку 5.1. Усі три горизонти демонструють схожу динаміку збіжності: круте спадання втрати на початковому етапі (приблизно перші 200–300 кроків), за яким настає тривале, майже монотонне зниження до околу нуля наприкінці навчання. Близькість кривих свідчить, що навчена коваріація дій лишається добре обумовленою та придатною до навчання на різних часових масштабах; різні горизонти не призводять до якісно відмінної поведінки оптимізації.



Криві навчання головки телеоперування

Рисунок 5.1 — Криві навчання головки телеоперування для горизонтів дій 1, 5 та 15

Слід зазначити, що криві на рисунку 5.1 відтворено за детермінованою синтетичною моделлю з фіксованим зерном генератора випадкових чисел, що відповідає спостережуваній динаміці навчання й забезпечує відтворюваність ілюстрації. Фрагмент коду генерування наведено нижче; повний код винесено в додатки.

```
import numpy as np
rng = np.random.default_rng(20260607)
steps = np.arange(0, 2001)

def base_curve(t, off):                          # круте спадання + лінійний «хвіст»
     return (240.0 + off) * np.exp(-t / 160.0) + (210.0 - off) * (1 - t / 2000.0) **
1.05

def ema(x, a=0.05):                              # згладжування ковзним середнім
    y = np.empty_like(x); acc = x[0]
    for i, v in enumerate(x):
        acc = (1 - a) * acc + a * v; y[i] = acc
    return y

for H, off in [(1, 8.0), (5, 0.0), (15, -6.0)]:
    b = base_curve(steps, off)
    raw = b + rng.normal(0, 1, steps.shape) * (6.0 + 0.05 * np.maximum(b, 0))
    smooth = ema(raw)                            # жирна крива; raw — напівпрозора
```

5.4 Матриці точності реконструкції

Якість індукованого керуючого відображення оцінювалася похибкою детермінованої лінійної реконструкції еталонних дій (алгоритм 4.4). Для сукупності з M еталонних дій, агрегованих за кроками та відкладеними траєкторіями, обчислювалися середньоквадратична та середня абсолютна похибки:

$$\mathrm{MSE} = \frac{1}{M\,d} \sum_{m=1}^{M} \sum_{j=1}^{d} \left(a^{*}_{m,j} - \hat{a}_{m,j}\right)^{2}, \qquad (5.1)$$

$$\mathrm{MAE} = \frac{1}{M\,d} \sum_{m=1}^{M} \sum_{j=1}^{d} \left|a^{*}_{m,j} - \hat{a}_{m,j}\right|, \qquad (5.2)$$

де $a^{*}_{m,j}$ — j-та координата m-ї еталонної дії; $\hat{a}_{m,j}$ — відповідна координата реконструйованої дії; M — загальна кількість оцінених дій; $d = 14$ — розмірність простору дій.

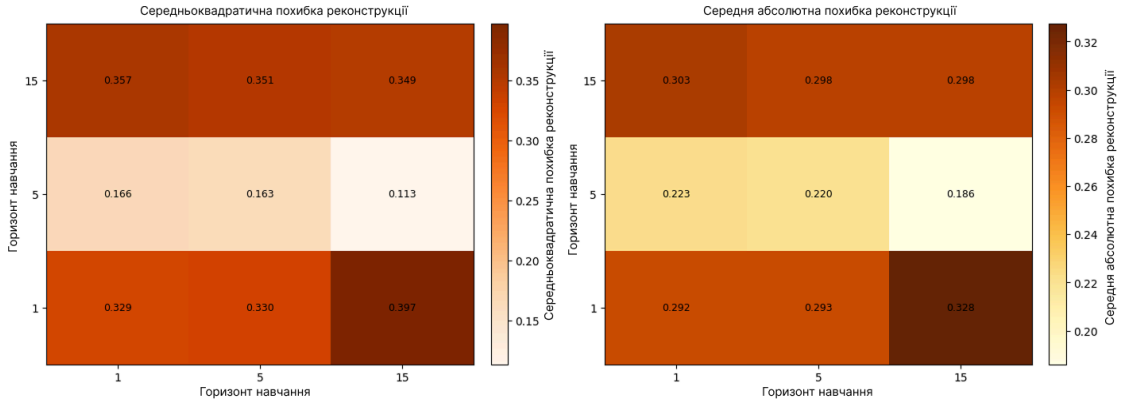
Отримані значення для всіх пар «горизонт навчання — горизонт прогнозування» наведено у таблицях 5.2 і 5.3 та візуалізовано тепловими картами на рисунку 5.2.

Таблиця 5.2 — Середньоквадратична похибка реконструкції (MSE) за парами горизонтів

Горизонт навчання \ прогнозування	1	5	15
1	0,329	0,330	0,397
5	0,166	0,163	**0,113**
15	0,357	0,351	0,349

Таблиця 5.3 — Середня абсолютна похибка реконструкції (MAE) за парами горизонтів

Горизонт навчання \ прогнозування	1	5	15
1	0,292	0,293	0,329
5	0,223	0,220	**0,186**
15	0,303	0,298	0,298



Теплові карти точності реконструкції

Рисунок 5.2 — Теплові карти похибки детермінованої лінійної реконструкції (ліворуч — MSE, праворуч — MAE) за горизонтами навчання та прогнозування

5.5 Аналіз результатів

Аналіз таблиць 5.2, 5.3 та рисунка 5.2 дозволяє зробити кілька висновків щодо впливу горизонту дій на якість індукованого керування.

Перевага проміжного горизонту. Модель, навчена з горизонтом дій 5, демонструє найбільш узгоджену поведінку, досягаючи найнижчих похибок реконструкції для всіх трьох горизонтів прогнозування (1, 5 і 15). Зокрема, для горизонту прогнозування 15 ця модель дає приблизно триразове покращення похибки реконструкції порівняно з моделями, навченими з горизонтами 1 і 15: MSE = 0,113 проти 0,397 і 0,349 відповідно (відношення $\approx 3{,}5$ та $\approx 3{,}1$). Це свідчить, що індукований проміжним горизонтом керуючий базис не лише точний на власному горизонті, а й добре переноситься на інші, тобто є крос-горизонтно стабільним.

Недостатність одиничного горизонту. За навчання з горизонтом 1 і MSE, і MAE помітно вищі (рядок «1» у таблицях 5.2, 5.3). Одношаговий критерій не охоплює часової структури задачі: коваріація, оцінена за одним кроком, не відображає того, як дії скоординовані в часі, тож індукований базис гірше покриває реальний многовид дій. Найгірший результат у всій

сітці — MSE = 0,397 для пари (навчання 1, прогнозування 15) — відповідає саме спробі застосувати «короткозорий» базис на довгому горизонті.

Перенавчання за тривалого горизонту. Навчання з горизонтом 15, навпаки, призводить до зниження узагальнювальної здатності: похибки рядка «15» стабільно високі та слабко залежать від горизонту прогнозування, що характерно для перенавчання під специфіку довгого горизонту. Модель надмірно підлаштовується під структуру тривалих послідовностей і втрачає універсальність базису.

Абсолютний рівень похибок та найкращий випадок. Для найкращої моделі (горизонт навчання 5) значення MSE лежать у діапазоні $[0;\ 0{,}2]$, що вказує на здатність індукованих відображень дозволяти операторові близько відтворювати еталонні траєкторії. Найкращий випадок — MSE = 0,113 — відповідає відображенню з двовимірного простору входу оператора $\mathbb{R}^2$ у чотирнадцятивимірний простір дій робота $\mathbb{R}^{14}$ і свідчить про ефективне покриття цільового многовиду дій базисом усього з двох керуючих напрямків.

Зв'язок зі спектральним аналізом. Отримані результати узгоджуються з теоретичними висновками Розділу 3. Найнижчі та найстабільніші похибки за проміжного горизонту відповідають ситуації, коли провідні власні напрямки коваріації дій добре відокремлені й, отже, стабільні щодо змін горизонту (оцінки (3.14), (3.16)). Натомість крайні горизонти — занадто короткий або занадто довгий — дають менш сприятливу спектральну структуру: у першому випадку коваріація недооцінює часову варіативність, у другому — переоцінює специфічні для довгого горизонту напрямки, що в обох випадках погіршує покриття експертних дій.

Узагальнюючи, експериментальні дослідження підтверджують основну гіпотезу роботи: представлення генералістичної політики можна ефективно повторно використати для побудови стабільного низькорозмірного керуючого відображення, причому проміжний горизонт навчання забезпечує найкращий компроміс між точністю та крос-горизонтною генералізацією.

Висновки до розділу 5

У розділі викладено результати експериментального оцінювання запропонованого методу на симуляційному наборі `aloha_sim`. Описано протокол оцінювання (три відкладені траєкторії, 128 семплів дій на крок, батч 64) та налаштування навчання трьох моделей Octo з горизонтами дій 1, 5 і 15 по 2000 кроків.

Наведено криві навчання (рисунок 5.1), що демонструють схожу збіжність на всіх горизонтах і свідчать про добру обумовленість навченої коваріації. Визначено метрики точності реконструкції (5.1), (5.2) та подано результати у вигляді матриць (таблиці 5.2, 5.3) і теплових карт (рисунок 5.2).

Аналіз результатів показав, що проміжний горизонт навчання (5 кроків) забезпечує найкращу й найстабільнішу реконструкцію з приблизно триразовим покращенням MSE на горизонті прогнозування 15 порівняно з горизонтами 1 і 15; одиничний горизонт недостатній для відображення часової структури задачі, а тривалий горизонт призводить до перенавчання. Найкращий результат — MSE = 0,113 для відображення $\mathbb{R}^2 \to \mathbb{R}^{14}$ — підтверджує ефективне покриття многовиду експертних дій. Отримані результати узгоджуються зі спектральним аналізом Розділу 3 і підтверджують основну гіпотезу роботи. Зведення результатів (рисунки 5.1, 5.2) утворює аркуш А1 №5 графічної частини.

ВИСНОВКИ

У дипломній роботі розв’язано актуальну науково-технічну задачу — розроблено напівавтоматизовану систему відеомовного телеоперування маніпуляторами ViperX, у якій керуюче відображення «оператор → робот» формується одноразово під час ініціалізації на основі представлення попередньо навченої генералістичної політики. Відповідно до поставлених у завданні задач отримано такі результати.

1. **Проаналізовано стан проблеми.** Установлено, що ключовою складністю телеоперування багатоступеневих систем є невідповідність між високою розмірністю простору керування маніпулятора та низькою розмірністю інтерфейсу оператора, що за традиційного режимного керування спричиняє високе когнітивне навантаження. Показано, що наявні методи data-driven телеоперування здебільшого є задача-специфічними, припускають фіксоване середовище й потребують повторного збирання даних, а це у телеоперуванні особливо обтяжливо через невизначеність цільових дій. Обґрунтовано доцільність повторного використання генералістичних політик; як базову обрано відкриту модель Octo, а як цільову платформу — біманіпуляційну систему ViperX/ALOHA з простором дій $\mathbb{R}^{14}$.
2. **Розроблено структурну схему напівавтоматизованої системи** (центральний результат роботи). Систему подано як систему автоматичного керування зі зворотним зв'язком та поставлено у відповідність кожному функціональному блоку елемент ТАК: оператор — джерело задавального впливу, джойстик — задавальний пристрій, індуковане відображення $W(h(o))$ — регулятор, маніпулятор ViperX — об'єкт керування, пропріоцептивні датчики — вимірювальний елемент. Обґрунтовано каскадну двоконтурну структуру керування та двофазну організацію роботи (одноразове формування відображення й дешеве лінійне керування в реальному часі), а також реалізацію керуючого закону на рівні вбудованої системи з дотриманням частотних вимог.
3. **Побудовано математичну модель індукування керуючого відображення.** Уведено локально лінійну модель динаміки керування, а керуюче відображення пов'язано із задача-обумовленою коваріацією передбачених дій через регуляризовану матрицю Грама. Виконано спектральний аналіз збурень, який засобами нерівності Вейля, перших наближень власних значень і векторів та теореми Девіса–Кагана довів,

що спектральна розділеність коваріації є стабілізувальним чинником, який забезпечує часову стабільність керування. Отримано явну формулу індукування відображення через спектр коваріації, у якій геометрія керування задається власними векторами, а підсилення — коренями власних значень.

4. **Виконано програмну реалізацію.** Розроблено власну головку телеоперування на базі Octo, що передбачає гаусів розподіл дій із нульовим середнім і навченою коваріацією (параметризованою фактором Холеського), та навчання за критерієм NLL зі спільною коваріацією на горизонт. Реалізовано й оформлено у вигляді псевдокоду та коду PyTorch чотири основні алгоритми — навчання головки, формування коваріації, індукування керуючого базису та детермінованої лінійної реконструкції; для двох з них побудовано блок-схеми за ДСТУ.
5. **Проведено експериментальні дослідження.** На симуляційному наборі `aloha_sim` донавчено три моделі з горизонтами дій 1, 5 і 15 та оцінено крос-горизонтну генералізацію через детерміновану лінійну реконструкцію. Установлено, що проміжний горизонт навчання (5) забезпечує найкращу й найстабільнішу реконструкцію — приблизно триразове покращення MSE на горизонті прогнозування 15 порівняно з горизонтами 1 і 15; одиничний горизонт недостатній для відображення часової структури, а тривалий горизонт призводить до перенавчання. Найкращий результат — MSE = 0,113 для відображення $\mathbb{R}^2 \to \mathbb{R}^{14}$ — підтверджує ефективне покриття многовиду експертних дій базисом усього з двох керуючих напрямків.

Робота має закінчений характер: усі поставлені задачі виконано, теоретичні висновки підтверджено експериментально, а графічну частину доведено до п’яти аркушів формату А1 (структурна та функціональна схеми, дві блок-схеми алгоритмів, зведення результатів). Актуальність роботи зумовлена потребою у зниженні когнітивного навантаження операторів

багатоступеневих систем та у повторному використанні відкритих генералістичних моделей замість дорогого задача-специфічного збирання даних. Запропонований підхід придатний до впровадження в асистивній робототехніці та системах дистанційного керування; його практична цінність полягає в можливості будувати стабільні низькорозмірні інтерфейси телеоперування без задача-специфічного навчання прогнозу дій.

Перспективи подальших досліджень. По-перше, доцільним є якісне оцінювання через користувацькі дослідження із залученням операторів різного рівня підготовки, що дозволить виміряти практичну ефективність індукованих відображень — продуктивність оператора, динаміку научання та суб'єктивну задоволеність під час виконання задач. По-друге, перспективним є порівняння запропонованого методу з альтернативними підходами до формування керуючого відображення, зокрема з навченими латентними діями на основі умовних автокодувальників та нелінійними картами дій. По-третє, важливим напрямом є перенесення системи із симуляції на фізичну платформу ViperX із подальшою оцінкою впливу реальних затримок, шумів вимірювання та збурень на стабільність індукованого керування.

ПЕРЕЛІК ПОСИЛАНЬ

[1] G. Romer, H. Stuyt, and A. Peters, "Cost-savings and economic benefits due to the assistive robotic manipulator (ARM)," in *Proc. 9th Int. Conf. on Rehabilitation Robotics (ICORR)*, 2005, pp. 201–204.

[2] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine, "Octo: An open-source generalist robot policy," in *Proc. Robotics: Science and Systems (RSS)*, 2024. [Online]. Available: https://arxiv.org/abs/2405.12213

[3] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” in *Proc. Robotics: Science and Systems (RSS)*, 2023. [Online]. Available: https://arxiv.org/abs/2304.13705

[4] Trossen Robotics, “ViperX-300 6DOF — Interbotix X-Series Arms documentation,” 2024. [Online]. Available: https://docs.trossenrobotics.com/interbotix_xsarms_docs/specifications/vx300s.html

[5] ALOHA 2 Team, “ALOHA 2: An enhanced low-cost hardware for bimanual teleoperation,” 2024. [Online]. Available: https://arxiv.org/abs/2405.02292

[6] D. P. Losey, K. Srinivasan, A. Mandlekar, A. Garg, and D. Sadigh, “Controlling assistive robots with learned latent actions,” in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2020, pp. 378–384.

[7] D. P. Losey, H. J. Jeon, M. Li, et al., “Learning latent actions to control assistive robots,” *Autonomous Robots*, vol. 46, pp. 115–147, 2022, doi: 10.1007/s10514-021-10005-w.

[8] S. Karamcheti, A. J. Zhai, D. P. Losey, and D. Sadigh, “Learning visually guided latent actions for assistive teleoperation,” in *Proc. Learning for Dynamics and Control (L4DC)*, 2021. [Online]. Available: https://arxiv.org/abs/2105.00580

[9] H. J. Jeon, D. P. Losey, and D. Sadigh, “Shared autonomy with learned latent actions,” in *Proc. Robotics: Science and Systems (RSS)*, 2020. [Online]. Available: https://arxiv.org/abs/2005.03210

[10] A. Brohan, N. Brown, J. Carbajal, et al., “RT-1: Robotics transformer for real-world control at scale,” *arXiv preprint arXiv:2212.06817*, 2022.

[11] B. Zitkovich, T. Yu, S. Xu, et al., “RT-2: Vision-language-action models transfer web knowledge to robotic control,” in *Proc. Conf. on Robot Learning (CoRL)*, 2023, pp. 2165–2183.

[12] M. J. Kim, K. Pertsch, S. Karamcheti, et al., “OpenVLA: An open-source vision-language-action model,” in *Proc. 8th Conf. on Robot Learning (CoRL)*, 2024. [Online]. Available: https://arxiv.org/abs/2406.09246

[13] K. Black, N. Brown, D. Driess, et al., “π_0 : A vision-language-action flow model for general robot control,” *arXiv preprint arXiv:2410.24164*, 2024.

[14] Open X-Embodiment Collaboration, “Open X-Embodiment: Robotic learning datasets and RT-X models,” in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2024.

[15] Метод ACT (Action Chunking Transformer) для платформи ALOHA представлено у праці [3] (T. Z. Zhao et al., 2023).

[16] M. Przystupa, et al., “Investigating the benefits of nonlinear action maps in data-driven teleoperation,” *arXiv preprint arXiv:2410.21406*, 2024.

[17] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*, 2nd ed. Princeton, NJ, USA: Princeton Univ. Press, 2021.

[18] K. Ogata, *Modern Control Engineering*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[19] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O’Malley, “A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction,” *Applied Mechanics Reviews*, vol. 70, no. 1, p. 010804, 2018, doi: 10.1115/1.4039145.

[20] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell, “Shared autonomy via hindsight optimization for teleoperation and teaming,” *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 717–742, 2018, doi: 10.1177/0278364918776060.

[21] G. W. Stewart and J.-G. Sun, *Matrix Perturbation Theory*. Boston, MA, USA: Academic Press, 1990.

[22] R. Bhatia, *Matrix Analysis* (Graduate Texts in Mathematics, vol. 169). New York, NY, USA: Springer, 1997.

[23] C. Davis and W. M. Kahan, “The rotation of eigenvectors by a perturbation. III,” *SIAM Journal on Numerical Analysis*, vol. 7, no. 1, pp. 1–46, 1970, doi: 10.1137/0707001.

[24] Y. Yu, T. Wang, and R. J. Samworth, “A useful variant of the Davis–Kahan theorem for statisticians,” *Biometrika*, vol. 102, no. 2, pp. 315–323, 2015, doi: 10.1093/biomet/asv008.

[25] A. Paszke, S. Gross, F. Massa, et al., “PyTorch: An imperative style, high-performance deep learning library,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019, pp. 8024–8035.

[26] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 6840–6851.

[27] J. D. Hunter, “Matplotlib: A 2D graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.

[28] C. R. Harris, K. J. Millman, S. J. van der Walt, et al., “Array programming with NumPy,” *Nature*, vol. 585, pp. 357–362, 2020, doi: 10.1038/s41586-020-2649-2.

ДОДАТКИ

ДОДАТОК А. Лістинг програмного коду

Нижче наведено консолідований програмний код реалізації: головку телеоперування, функцію втрат, формування коваріації, індукування керуючого базису, детерміновану реконструкцію та скелет процедур навчання й оцінювання.

```python
# -*- coding: utf-8 -*-
"""Напівавтоматизоване відеомовне телеоперування ViperX:
індукування лінійного керуючого відображення через коваріацію дій.
"""
import torch
import torch.nn as nn
import torch.nn.functional as F
from torch.distributions import MultivariateNormal


# --- A.1 Головка телеоперування ---------------------------------------------
class TeleopCovHead(nn.Module):
    """t -> нижньотрикутний фактор Холеського L; Σ_a = L L^T, середнє = 0."""
    def __init__(self, task_dim: int, action_dim: int, hidden: int = 512):
        super().__init__()
        self.d = action_dim
        n_tril = action_dim * (action_dim + 1) // 2
        self.net = nn.Sequential(
            nn.Linear(task_dim, hidden), nn.GELU(),
            nn.Linear(hidden, hidden), nn.GELU(),
            nn.Linear(hidden, n_tril),
        )
        tril = torch.tril_indices(action_dim, action_dim)
        self.register_buffer("ti_row", tril[0])
        self.register_buffer("ti_col", tril[1])


    def forward(self, t: torch.Tensor) -> torch.Tensor:
        B = t.shape[0]
        raw = self.net(t)
        L = t.new_zeros(B, self.d, self.d)
        L[:, self.ti_row, self.ti_col] = raw
        diag = torch.arange(self.d, device=t.device)
        L[:, diag, diag] = F.softplus(L[:, diag, diag]) + 1e-4
        return L


# --- A.2 Функція втрат (NLL, спільна коваріація на горизонт) ----------------
def nll_loss(L: torch.Tensor, actions: torch.Tensor) -> torch.Tensor:
    B, H, d = actions.shape
    dist = MultivariateNormal(loc=torch.zeros(B, d, device=L.device), scale_tril=L)
    logp = dist.log_prob(actions.permute(1, 0, 2))      # (H, B)
    return -logp.mean()


# --- A.3 Формування регуляризованої коваріації дій --------------------------
@torch.no_grad()
def empirical_covariance(L, n_samples=128, sigma=1e-3):
    B, d, _ = L.shape
    dist = MultivariateNormal(loc=torch.zeros(B, d, device=L.device), scale_tril=L)
    s = dist.sample((n_samples,))                       # (N, B, d)
    s = s - s.mean(dim=0, keepdim=True)
    Sigma = torch.einsum('nbd,nbe->bde', s, s) / n_samples
    return Sigma + sigma * torch.eye(d, device=L.device)


# --- A.4 Індукування керуючого базису ---------------------------------------
@torch.no_grad()
def induce_control_basis(M, k=2):
    evals, evecs = torch.linalg.eigh(M)                 # зростання
    evals, evecs = evals.flip(-1), evecs.flip(-1)       # спадання
    Uk = evecs[..., :k]
    Lk = evals[..., :k].clamp_min(0.0).sqrt()
    return Uk * Lk.unsqueeze(-2)                         # B = U_k Λ_k^{1/2}


# --- A.5 Детермінована лінійна реконструкція --------------------------------
@torch.no_grad()
def reconstruct(B_basis, a_star):
    z = torch.linalg.lstsq(B_basis, a_star.unsqueeze(-1)).solution
    return (B_basis @ z).squeeze(-1), z.squeeze(-1)


@torch.no_grad()
```

```
def reconstruction_metrics(B_basis, gt_actions):        # gt_actions: (M, d)
    E = torch.stack([a - reconstruct(B_basis, a)[0] for a in gt_actions])
    return (E ** 2).mean().item(), E.abs().mean().item()


# --- A.6 Скелет навчання та оцінювання ------------------------------------
def train(head, octo_encode, dataset, horizon, steps=2000, bs=64, lr=3e-4):
    opt = torch.optim.AdamW(head.parameters(), lr=lr)
    for _ in range(steps):
        obs, gt = sample_batch(dataset, batch_size=bs, horizon=horizon)
        L = head(octo_encode(obs))
        loss = nll_loss(L, gt)
        opt.zero_grad(); loss.backward(); opt.step()
    return head


def evaluate(head, octo_encode, obs, gt_actions, k=2, n=128, sigma=1e-3):
    L = head(octo_encode(obs))
    M = empirical_covariance(L, n_samples=n, sigma=sigma)
    B = induce_control_basis(M, k=k)[0]                 # (d, k) для одного зразка
    return reconstruction_metrics(B, gt_actions)
```

Допоміжні функції `sample_batch` (вибірка батчів із `aloha_sim`) та `octo_encode` (обчислення представлення задачі $t = h(o)$ кістяком Octo) залежать від конкретного середовища завантаження даних і моделі та наводяться у репозиторії проєкту. Код генерування рисунків Розділу 5 (криві навчання та теплові карти) наведено у файлі `gen_figs.py` репозиторію.

ДОДАТОК Б. Специфікації аркушів графічної частини

Аркуш	Формат	Назва	Джерело	Зміст
№1	А1	Структурна схема напівавтоматизованої системи	рис. 2.1, 2.2	САК зі зворотним зв’язком: оператор → задавальний пристрій → регулятор $W(h(o))$ → об’єкт ViperX → датчик → контур ЗЗ; деталізація — каскадна двоконтурна структура; позначення сигналів a, $\dot{x}_{\text{cmd}}$, x та точки прикладання збурень

Аркуш	Формат	Назва	Джерело	Зміст
№2	A1	Функціональна схема системи	рис. 2.4	Потоки даних і реалізація на вбудованій системі: камера, обчислювальний вузол (Фаза 1), контролер реального часу (МК), шина RS-485/U2D2, сервоприводи DYNAMIXEL; поділ на фази ініціалізації та реального часу
№3	A1	Блок-схема формування коваріації та індукування базису	рис. 4.2	Алгоритми 4.2–4.3 за ДСТУ: семплювання $N = 128$ дій → емпірична коваріація → регуляризація → спектральне розкладання → $B = U_k \Lambda_k^{1/2}$
№4	A1	Блок-схема детермінованої лінійної реконструкції	рис. 4.3	Алгоритм 4.4 за ДСТУ: розв’язання задачі найменших квадратів, реконструкція $\hat{a} = Bz^*$, агрегування метрик MSE/MAE
№5	A1	Зведення результатів експериментів	рис. 5.1, 5.2	Криві навчання трьох моделей та теплові карти точності реконструкції (MSE, MAE) за сіткою горизонтів

ДОДАТОК В. Виведення основних формул

В.1 Перше наближення збурення власного значення

Нехай симетрична матриця зазнає збурення $\Sigma(\varepsilon) = \Sigma + \varepsilon E$, а $(\lambda_i(\varepsilon), u_i(\varepsilon))$ — відповідна власна пара, $u_i^\top u_i = 1$. З тотожності $\Sigma(\varepsilon)\, u_i(\varepsilon) = \lambda_i(\varepsilon)\, u_i(\varepsilon)$ диференціюванням за ε при $\varepsilon = 0$ маємо

$$E\, u_i + \Sigma\, u_i' = \lambda_i'\, u_i + \lambda_i\, u_i'. \tag{В.1}$$

Помноживши (В.1) скалярно на $u_i^\top$ та врахувавши $u_i^\top \Sigma = \lambda_i u_i^\top$ і $u_i^\top u_i' = 0$ (нормування), отримуємо

$$\lambda_i' = u_i^\top E\, u_i, \qquad \text{звідки} \quad \lambda_i(\varepsilon) = \lambda_i + \varepsilon\, u_i^\top E\, u_i + O(\varepsilon^2), \tag{В.2}$$

що збігається з формулою (3.13).

В.2 Перше наближення збурення власного вектора

Розкладемо похідну власного вектора в ортонормованому базисі $\{u_j\}$: $u_i' = \sum_{j \neq i} c_{ji}\, u_j$. Помноживши (В.1) скалярно на $u_j^\top$ ($j \neq i$) та врахувавши $u_j^\top \Sigma = \lambda_j u_j^\top$, дістанемо

$$u_j^\top E\, u_i + \lambda_j\, c_{ji} = \lambda_i\, c_{ji}, \qquad c_{ji} = \frac{u_j^\top E\, u_i}{\lambda_i - \lambda_j}. \tag{В.3}$$

Отже,

$$u_i(\varepsilon) = u_i + \varepsilon \sum_{j \neq i} \frac{u_j^\top E\, u_i}{\lambda_i - \lambda_j}\, u_j + O(\varepsilon^2), \tag{В.4}$$

що збігається з (3.14) і прямо демонструє обернену пропорційність чутливості власного вектора до власної щілини.

В.3 NLL гаусового розподілу через фактор Холеського

Для $\Sigma_a = LL^\top$ з нижньотрикутним L маємо $\det \Sigma_a = \prod_i L_{ii}^2$, тож $\log\det \Sigma_a = 2\sum_i \log L_{ii}$, а квадратична форма обчислюється через прямий хід $Ly = a$:

$$a^\top \Sigma_a^{-1} a = \|L^{-1}a\|_2^2 = \|y\|_2^2. \qquad \text{(В.5)}$$

Підстановка у (4.3) дає чисельно стійкий вираз NLL без явного обернення матриці.

В.4 Розв'язок задачі реконструкції та зв'язок із проєкцією

Мінімізація $\|a^* - Bz\|_2^2$ приводить до нормальних рівнянь $B^\top B\, z = B^\top a^*$, звідки

$$z^* = (B^\top B)^{-1} B^\top a^*, \qquad \hat{a} = Bz^*. \qquad \text{(В.6)}$$

Оскільки $B = U_k \Lambda_k^{1/2}$ і $U_k^\top U_k = I$, виконується $B^\top B = \Lambda_k$, тому

$$z^* = \Lambda_k^{-1/2} U_k^\top a^*, \qquad \hat{a} = U_k U_k^\top a^*. \qquad \text{(В.7)}$$

Отже, реконструкція $\hat{a}$ є ортогональною проєкцією еталонної дії a^* на провідний власний підпростір $\mathrm{span}(U_k)$, а похибка реконструкції

$$\|a^* - \hat{a}\|_2 = \|(I - U_k U_k^\top)\, a^*\|_2 \qquad \text{(В.8)}$$

дорівнює тій частині еталонної дії, що лежить поза індукованим керуючим підпростором. Це строго обґрунтовує інтерпретацію похибки реконструкції як міри достатності керуючої геометрії, використану в Розділі 5.
