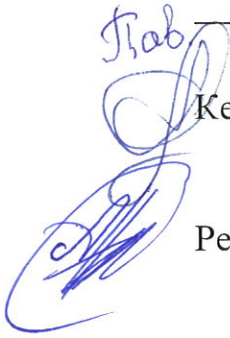


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
Факультет комп'ютерно-інтегрованих технологій, мехатроніки і робототехніки
Кафедра робототехніки, електроенергетики та автоматизації
ім. проф. Б.Б. Самотокіна

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи магістра
на тему: «Інтеграція контролерів управління системи PS та скриптів Python у
системи автоматизованого керування роботизованими платформами»

Виконав студент 2 курсу, групи АТ-29м
спеціальності 174 “ Автоматизація,
комп'ютерно-інтегровані технології
та робототехніка”

 Павлюк Владислав Віталійович

(прізвище та ініціали студента)

Керівник: к.т.н., доц. Добржанський О.О.

(науковий ступінь, вчене звання, посада,
прізвище та ініціали)

Рецензент: д.філос., Кравчук А.Р

(науковий ступінь, вчене звання, посада,
прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ “ЖИТОМИРСЬКА ПОЛІТЕХНІКА”
Факультет комп’ютерно-інтегрованих технологій, мехатроніки і робототехніки
Кафедра робототехніки, електроенергетики та автоматизації
ім. проф. Б.Б. Самотокіна

Спеціальність 174 “Автоматизація, комп’ютерно-інтегровані технології та робототехніка”

ЗАТВЕРДЖУЮ

Завідувач кафедри робототехніки,
електроенергетики та автоматизації
ім. проф. Б.Б. Самотокіна
к.т.н., доц. Громовий О.А.
« 17 » жовтня 2025р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра

Студент Павлюк Владислав Віталійович

1. Тема роботи: Інтеграція контролерів управління системи PS та скриптів Python у системи автоматизованого керування роботизованими платформами
2. Термін здачі закінченої роботи на кафедру «29» листопада 2025 р.
3. Вихідні дані до роботи: матеріали звіту про переддипломну практику, модель роботизованої платформи з електродвигунами та сервоприводами, контролер Sony PlayStation DualShock / DualSense, підключення контролера через USB або Bluetooth, програмне середовище Python з бібліотеками зчитування та обробки керуючих сигналів, реалізація ручного, напівавтоматичного й автоматизованого керування рухом платформи, забезпечення аварійної зупинки та безпечного припинення руху в разі втрати зв’язку
4. Зміст пояснювальної записки (перелік питань, які підлягають розробці): технічне завдання, огляд та аналіз існуючих систем керування роботизованими платформами, дослідження можливостей контролерів PlayStation і бібліотек Python, розробка структурної та функціональної схем системи, розробка алгоритмів керування, програмна реалізація, тестування та аналіз ефективності системи
5. Перелік презентативного матеріалу (з точним визначенням обов’язкових слайдів/креслень): структурна схема системи керування, функціональна схема,

схема взаємодії контролера PlayStation та Python-скриптів, блок-схема алгоритму керування й аварійної зупинки, результати тестування системи

6. Консультанти з кваліфікаційної роботи магістра із зазначенням розділів роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1, 2, 3	Добржанський О.О.		Пав.
4,5	Добржанський О.О.		Пав.

7. Дата видачі завдання 22.09.2025

Календарний план

№ з/п	Назва етапів виконання кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Підготувати звіт про переддипломну практику	1 тиждень	виконано
2.	Збір та аналіз джерел за темою роботи, огляд існуючих систем	2 тиждів	виконано
3.	Дослідження об'єкта та визначення вимог до системи	2 тиждів	виконано
4.	Розробка структурної та функціональної схеми	2 тиждів	виконано
5.	Вибір і обґрунтування технічних та програмних засобів	3 тиждів	виконано
6.	Розробка блок-схеми алгоритму	3 тиждів	виконано
7.	Аналіз отриманих результатів	4 тиждів	виконано
8.	Оформлення пояснювальної записки та презентаційного матеріалу	4 тиждів	виконано

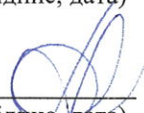
Студент


(підпис, дата)

В.В. Павлюк

(прізвище та ініціали)

Керівник роботи


(підпис, дата)

О.О. Добржанський

(прізвище та ініціали)

ЗМІСТ

ЗМІСТ	1
ВСТУП	8
Актуальність теми дослідження	8
Мета та завдання роботи	8
Об'єкт і предмет дослідження	8
Методи дослідження.....	8
Практичне значення роботи.....	8
Структура роботи.....	8
1.1. Поняття та класифікація роботизованих платформ	10
1.2. Системи автоматизованого керування роботами	11
1.3. Людино-машинні інтерфейси у робототехніці	12
1.4. Контролери керування як засіб взаємодії з роботами.....	13
1.5. Аналіз існуючих методів дистанційного керування роботизованими системами	15
Висновки до Розділу 1	17
РОЗДІЛ 2. КОНТРОЛЕРИ УПРАВЛІННЯ СИСТЕМИ PLAYSTATION ЯК ІНТЕРФЕЙС КЕРУВАННЯ	18
2.1. Загальна характеристика контролерів PlayStation (DualShock, DualSense)	18
2.2. Апаратна архітектура та сенсорні можливості контролерів PS.....	23
2.3. Протоколи зв'язку (USB, Bluetooth)	27
2.4. Зчитування сигналів з контролерів PS.....	31

2.5. Переваги та обмеження використання контролерів PS у робототехніці	35
Висновки до Розділу 2	39
РОЗДІЛ 3. МОВА ПРОГРАМУВАННЯ PYTHON У СИСТЕМАХ АВТОМАТИЗОВАНОГО КЕРУВАННЯ	
	40
3.1. Особливості використання Python у робототехніці	41
3.2. Бібліотеки Python для роботи з контролерам	42
3.3. Інтеграція Python-скриптів з апаратними контролерами	Ошибка!
Закладка не определена.	
3.4. Обробка сигналів управління в режимі реального часу	42
3.5. Безпека та стабільність програмних рішень	43
Висновки до Розділу 3	44
РОЗДІЛ 4. ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗОВАНОГО КЕРУВАННЯ РОБОТИЗОВАНОЮ ПЛАТФОРМОЮ	
	45
4.1. Вимоги до системи керування.....	45
4.2 Архітектура програмно-апаратного комплексу.....	45
4.3 Схема взаємодії контролера PS та Python-скриптів.....	46
4.4 Алгоритми керування рухом роботизованої платформи.....	46
4.5 Реалізація системи керування на Python.....	47
Висновки до Розділу 4	48
РОЗДІЛ 5. ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ СИСТЕМИ	
	49
5.1. Методика тестування системи керування.....	49
5.2. Експериментальні дослідження роботи платформи.....	50
5.3. Аналіз результатів тестування.....	50

5.4. Порівняння з альтернативними системами керування.....	50
5.5. Оцінка надійності та точності управління.....	51
Висновки до Розділу 5	51
ВИСНОВКИ.....	52
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	53
ДОДАТОК А	54
ДОДАТОК Б.....	61
ДОДАТОК В.....	70
ДОДАТОК Г.....	81

АНОТАЦІЯ

У даній роботі розглянуто питання інтеграції контролерів управління системи PlayStation та скриптів мови програмування Python у систему автоматизованого керування роботизованими платформами. Основну увагу приділено аналізу теоретичних засад автоматизованого керування, апаратних і функціональних можливостей контролерів PlayStation, а також програмних засобів реалізації алгоритмів керування.

У роботі обґрунтовано доцільність використання ігрових контролерів PlayStation як людино-машинного інтерфейсу для керування роботизованими платформами. Розглянуто архітектуру програмно-апаратного комплексу системи керування, принципи зчитування та обробки сигналів контролера, а також реалізацію алгоритмів керування рухом із використанням Python-скриптів. Описано застосування спеціалізованих бібліотек Python для роботи з апаратними пристроями, математичної обробки сигналів і забезпечення роботи системи в режимі реального часу.

Проведено проєктування, тестування та аналіз ефективності системи автоматизованого керування роботизованою платформою. Результати експериментальних досліджень підтвердили стабільність роботи системи, точність виконання команд оператора та переваги використання контролерів PlayStation порівняно з традиційними засобами керування. Особливу увагу приділено питанням надійності та безпеки функціонування системи.

Отримані результати свідчать про ефективність запропонованого підходу та можливість його практичного застосування у навчальних, дослідницьких і прикладних робототехнічних системах, а також про перспективність подальшого розвитку систем автоматизованого керування з використанням сучасних програмно-апаратних засобів.

Ключові слова: автоматизоване керування, роботизована платформа, контролери PlayStation, Python, людино-машинний інтерфейс, алгоритми керування.

THE SUMMARY

This paper addresses the integration of PlayStation control system controllers and Python programming language scripts into an automated control system for robotic platforms. The main focus is placed on the analysis of the theoretical foundations of automated control, the hardware and functional capabilities of PlayStation controllers, and software tools for implementing control algorithms.

The study substantiates the feasibility of using PlayStation game controllers as a human–machine interface for controlling robotic platforms. The architecture of the hardware–software control system is examined, along with the principles of reading and processing controller input signals and implementing motion control algorithms using Python scripts. The application of specialized Python libraries for hardware interaction, signal processing, and real-time system operation is described.

The design, testing, and performance analysis of the automated control system for a robotic platform were carried out. The experimental results confirmed the stability of the system, the accuracy of command execution, and the advantages of using PlayStation controllers compared to traditional control devices. Special attention is paid to issues of system reliability and operational safety.

The obtained results demonstrate the effectiveness of the proposed approach and its practical applicability in educational, research, and applied robotic systems, as well as the prospects for further development of automated control systems using modern hardware and software solutions.

Keywords: automated control, robotic platform, PlayStation controllers, Python, human–machine interface, control algorithms

Вступ

Актуальність роботи

Актуальність даної роботи зумовлена зростаючими вимогами до гнучкості, адаптивності та економічної доцільності систем керування роботизованими платформами. Використання масових ігрових контролерів як людино-машинного інтерфейсу дозволяє знизити вартість системи, скоротити час розробки та підвищити ергономічність керування. Застосування мови програмування Python забезпечує універсальність програмної реалізації та спрощує інтеграцію з сучасними апаратними платформами, що робить даний підхід перспективним для навчальних, дослідницьких і прикладних робототехнічних систем.

Мета роботи

Метою даної магістерської роботи є аналіз, розробка та дослідження методів інтеграції контролерів управління системи PlayStation зі скриптами мови програмування Python для створення ефективної системи автоматизованого керування роботизованими платформами.

Об'єкт дослідження

Об'єктом дослідження є процеси автоматизованого керування роботизованими платформами з використанням людино-машинних інтерфейсів.

Предмет дослідження

Предметом дослідження є методи, алгоритми та апаратно-програмні засоби інтеграції контролерів управління системи Sony PlayStation зі скриптами мови програмування Python у системах автоматизованого керування роботизованими платформами.

Основні задачі дослідження

Для досягнення поставленої мети в роботі необхідно розв'язати такі задачі:

- проаналізувати сучасні підходи до побудови систем автоматизованого керування роботизованими платформами;
- дослідити апаратні та функціональні можливості контролерів управління системи Sony PlayStation;
- проаналізувати протоколи зв'язку (USB, Bluetooth) та особливості передачі даних від контролерів до обчислювального модуля;
- розробити програмні модулі на мові Python для зчитування, обробки та фільтрації керуючих сигналів;
- реалізувати алгоритми ручного, напіваавтоматичного та автоматизованого керування роботизованою платформою;
- розробити структурні, функціональні та інформаційні схеми системи автоматизованого керування;
- провести експериментальну оцінку ефективності запропонованої системи за показниками точності, затримки та надійності керування.

Сучасний етап розвитку робототехніки характеризується активним упровадженням автоматизованих систем керування у різних сферах діяльності, зокрема в промисловості, наукових дослідженнях, освіті та сервісній робототехніці. Однією з ключових задач при створенні роботизованих платформ є розробка зручних, надійних та економічно доцільних засобів керування, які забезпечують ефективну взаємодію оператора з технічною системою.

Традиційні пульти керування та спеціалізовані промислові контролери часто мають високу вартість і потребують значних витрат часу на розробку та налаштування. У зв'язку з цим актуальним є пошук альтернативних рішень, які поєднують функціональність, ергономічність і доступність. Одним із таких

рішень є використання ігрових контролерів системи PlayStation (DualShock, DualSense), що володіють розвиненими сенсорними можливостями, інтуїтивним інтерфейсом та підтримкою стандартних протоколів зв'язку.

Водночас мова програмування Python набула широкого застосування у сфері автоматизації та робототехніки завдяки простоті синтаксису, великій кількості бібліотек і можливості швидкої розробки програмного забезпечення. Поєднання контролерів управління системи PS зі скриптами Python дозволяє реалізувати гнучкі алгоритми обробки сигналів, організувати різні режими керування та забезпечити інтеграцію з мікроконтролерами й одноплатними комп'ютерами роботизованих платформ.

Таким чином, дослідження питань інтеграції контролерів управління системи PlayStation та скриптів Python у систему автоматизованого керування роботизованими платформами є актуальним і практично значущим. Метою даної роботи є аналіз можливостей використання контролерів PS у робототехнічних системах, розробка принципів їх програмної інтеграції з використанням Python та оцінка ефективності запропонованого підходу для керування роботизованими платформами.

РОЗДІЛ 1

ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗОВАНОГО КЕРУВАННЯ РОБОТИЗОВАНИМИ ПЛАТФОРМАМИ

1.1. Загальні положення автоматизованого керування роботизованими системами

Роботизована платформа є складною технічною системою, призначеною для виконання фізичних дій у реальному середовищі з використанням програмного керування, сенсорних засобів та виконавчих механізмів. У сучасній науково-технічній літературі роботизовану платформу розглядають як інтегрований апаратно-програмний комплекс, здатний функціонувати в автоматичному, напіваавтоматичному або автономному режимі.

З позицій системного підходу роботизована платформа включає такі основні підсистеми: механічну (шасі, приводи, маніпулятори), електронну (контролери, сенсори, засоби зв'язку) та програмну (алгоритми керування, обробка даних, інтерфейс взаємодії). Взаємодія між цими підсистемами забезпечує виконання заданих функцій з необхідною точністю та надійністю.

Класифікація роботизованих платформ є багатовимірною та здійснюється за низкою критеріїв. За типом пересування виділяють колісні, гусеничні, крокуючі та гібридні платформи. Колісні платформи характеризуються простотою конструкції та високою енергоефективністю, тоді як гусеничні забезпечують кращу прохідність у складних умовах. Крокуючі роботи мають підвищену маневреність, але потребують складніших алгоритмів керування.

За рівнем автономності роботизовані платформи поділяються на дистанційно керовані, напіваавтономні та повністю автономні. Дистанційно керовані системи потребують постійної участі оператора, тоді як автономні платформи здатні самостійно приймати рішення на основі сенсорної інформації.

Напівавтономні системи поєднують переваги обох підходів і є найбільш поширеними у прикладній робототехніці.



Рис.1.

Автоматизоване керування є одним із ключових напрямів розвитку сучасної робототехніки, що забезпечує ефективну, точну та безпечну взаємодію роботизованих платформ з навколишнім середовищем. У широкому розумінні автоматизоване керування являє собою сукупність методів, алгоритмів та технічних засобів, спрямованих на забезпечення заданої поведінки технічної системи без безпосередньої участі оператора або з його мінімальним втручанням.

Роботизовані платформи відносяться до класу складних динамічних систем, які функціонують у реальному часі та характеризуються багатоканальністю, нелінійністю та наявністю зворотних зв'язків. Процес керування такими системами передбачає формування керуючих впливів на основі інформації,

отриманої від сенсорів, а також зовнішніх команд оператора або програмного модуля.

Основною метою автоматизованого керування роботизованими платформами є забезпечення стабільного руху, точного позиціонування, адаптації до змін зовнішнього середовища та виконання заданих технологічних або функціональних операцій. Для досягнення цієї мети використовуються класичні та інтелектуальні методи керування, зокрема пропорційно-інтегрально-диференціальні (PID) регулятори, адаптивні алгоритми, методи оптимального та нечіткого керування.

1.2. Роботизовані платформи як об'єкти керування

Роботизована платформа являє собою мобільну або стаціонарну технічну систему, оснащену виконавчими механізмами, сенсорними модулями та обчислювальними засобами для реалізації керування. З точки зору теорії керування роботизовані платформи розглядаються як об'єкти керування з багатьма входами та виходами, де кожен керуючий сигнал впливає на певний параметр руху або стану системи.

Особливістю роботизованих платформ є їхня здатність до автономного або напіваавтономного функціонування. Це досягається завдяки використанню сенсорів положення, швидкості, прискорення, а також датчиків навколишнього середовища, таких як ультразвукові, інфрачервоні або візуальні сенсори. Дані, отримані з цих пристроїв, використовуються для формування зворотного зв'язку, який є основою стабільного керування.

Класифікація роботизованих платформ може здійснюватися за різними ознаками: типом пересування (колісні, гусеничні, крокуючі), рівнем

автономності, функціональним призначенням та способом керування. Кожен з цих типів потребує застосування відповідних алгоритмів керування, що враховують динамічні характеристики платформи.

Система автоматизованого керування роботами являє собою сукупність технічних і програмних засобів, що забезпечують формування керуючих впливів на роботизовану платформу відповідно до заданих алгоритмів. Основним завданням такої системи є досягнення бажаної поведінки об'єкта керування з урахуванням зовнішніх та внутрішніх збурень.

Класична структура системи автоматизованого керування включає об'єкт керування, керуючий пристрій, датчики та канал зворотного зв'язку. У робототехнічних системах ця структура ускладнюється за рахунок багатоканальності та необхідності роботи в режимі реального часу. Система повинна одночасно обробляти сигнали з численних сенсорів і формувати керуючі команди для кількох виконавчих механізмів.

У сучасних роботизованих платформах широко застосовуються цифрові системи керування, реалізовані на мікроконтролерах або одноплатних комп'ютерах. Програмне забезпечення таких систем реалізує алгоритми стабілізації, планування руху та адаптації до змін умов експлуатації. Важливою характеристикою системи автоматизованого керування є її стійкість, швидкодія та точність.

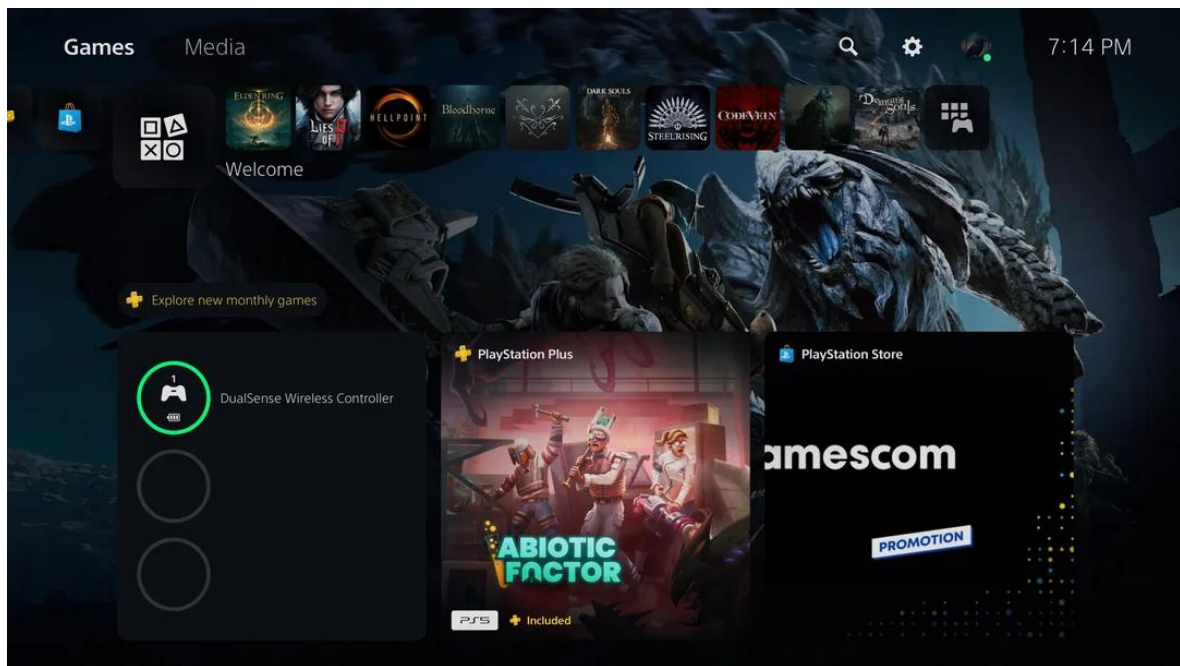


Рис.2.

1.3. Структура системи автоматизованого керування

Система автоматизованого керування роботизованою платформою зазвичай складається з декількох функціональних рівнів: рівня сприйняття, рівня прийняття рішень та рівня виконавчих механізмів. Така багаторівнева архітектура дозволяє ефективно розподілити обчислювальні ресурси та забезпечити гнучкість системи.

На рівні сприйняття відбувається збір та попередня обробка даних з сенсорів. Отримана інформація використовується для оцінки поточного стану платформи та умов зовнішнього середовища. Рівень прийняття рішень реалізує алгоритми керування, планування руху та реагування на зовнішні впливи. Рівень виконавчих механізмів забезпечує фізичну реалізацію керуючих команд шляхом впливу на електродвигуни, серводвигуни або інші приводи.

Особливу роль у структурі системи керування відіграє інтерфейс взаємодії з оператором, який може реалізовуватися як апаратними засобами (контролери керування), так і програмними модулями. Саме на цьому рівні доцільним є використання контролерів управління системи PlayStation у поєднанні з Python-скриптами.

Людино-машинний інтерфейс (НМІ) є ключовою складовою робототехнічних систем, оскільки він забезпечує взаємодію оператора з роботизованою платформою. Ефективність керування значною мірою залежить від зручності та інтуїтивності інтерфейсу.

У робототехніці використовуються різні типи НМІ: графічні інтерфейси користувача, сенсорні панелі, джойстики, спеціалізовані пульти та ігрові контролери. Основною тенденцією розвитку людино-машинних інтерфейсів є зменшення когнітивного навантаження на оператора та підвищення точності виконання команд.

Особливе місце серед НМІ займають пристрої, що забезпечують багатоканальне аналогове керування, оскільки вони дозволяють оператору більш природно взаємодіяти з роботом. Саме до таких пристроїв належать контролери ігрових систем, які все частіше використовуються у прикладній робототехніці.



Рис.3.

1.4. Людино-машинна взаємодія у системах керування роботами

Людино-машинна взаємодія (HMI) є важливим аспектом автоматизованих систем керування, оскільки вона визначає ефективність та зручність управління роботизованою платформою. Сучасні HMI-системи спрямовані на зменшення когнітивного навантаження на оператора та підвищення точності виконання команд.

Використання ігрових контролерів як засобів керування роботами є перспективним напрямом, оскільки такі пристрої мають ергономічний дизайн, велику кількість органів керування та підтримку аналогових сигналів. Контролери PlayStation, зокрема DualShock та DualSense, дозволяють реалізувати інтуїтивне керування рухом роботизованої платформи, маніпуляторами та допоміжними функціями.

Інтеграція таких контролерів у системи керування потребує використання програмних засобів для зчитування та інтерпретації сигналів, що обґрунтовує застосування високорівневих мов програмування, зокрема Python.

Контролери керування є апаратними засобами введення, призначеними для передачі команд оператора у системи керування роботизованою платформою. Вони можуть містити аналогові та цифрові органи керування, що дозволяє реалізувати складні сценарії взаємодії.

Ігрові контролери, зокрема контролери системи PlayStation, мають значні переваги порівняно з традиційними промисловими пультами. Вони характеризуються високою ергономічністю, наявністю аналогових стиків, датчиків руху та зручних кнопок. Завдяки стандартним інтерфейсам підключення такі контролери можуть бути інтегровані у системи керування роботами з використанням програмних засобів.

Застосування контролерів керування дозволяє реалізувати інтуїтивне дистанційне керування роботизованими платформами, що є особливо важливим у навчальних та дослідницьких проєктах.

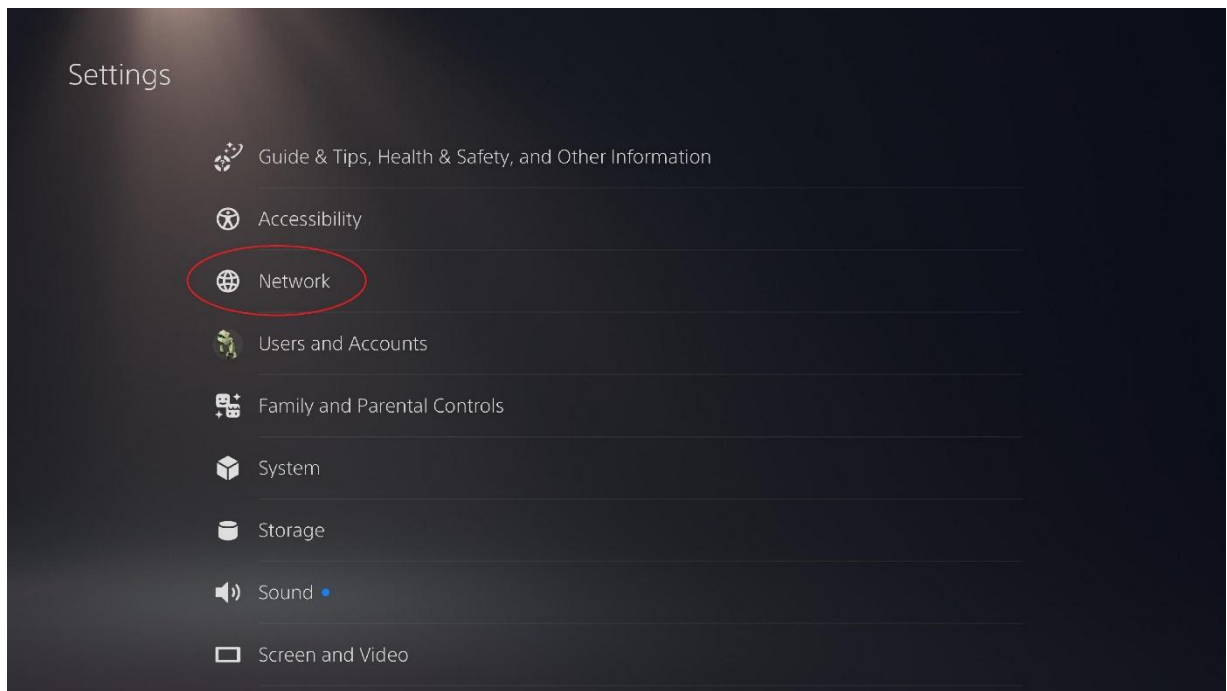


Рис.4.

1.5. Значення програмних засобів у автоматизованому керуванні

Програмне забезпечення є центральним елементом сучасних систем автоматизованого керування роботизованими платформами. Саме програмні алгоритми забезпечують обробку даних, прийняття рішень та адаптацію системи до змінних умов функціонування.

Мова програмування Python набула широкого застосування в робототехніці завдяки простоті синтаксису, наявності великої кількості бібліотек та підтримці роботи в режимі реального часу. Python дозволяє швидко реалізовувати прототипи систем керування, інтегрувати апаратні контролери та забезпечувати гнучке налаштування параметрів керування.

Таким чином, поєднання автоматизованих систем керування, роботизованих платформ, апаратних контролерів та програмних рішень створює основу для розробки ефективних та універсальних робототехнічних систем.

Методи дистанційного керування роботизованими системами поділяються на прямі та опосередковані. Пряме керування передбачає безпосередню передачу команд оператора до виконавчих механізмів, що забезпечує швидку реакцію системи. Однак такий підхід потребує постійної участі людини.

Опосередковані методи дистанційного керування базуються на використанні автономних алгоритмів, де оператор задає лише загальні параметри або цілі. Такі системи є більш стійкими до збоїв, але потребують складніших алгоритмів обробки даних.

Порівняльний аналіз показує, що поєднання дистанційного керування за допомогою ігрових контролерів та програмної обробки сигналів мовою Python дозволяє досягти оптимального балансу між простотою реалізації, точністю керування та гнучкістю системи.

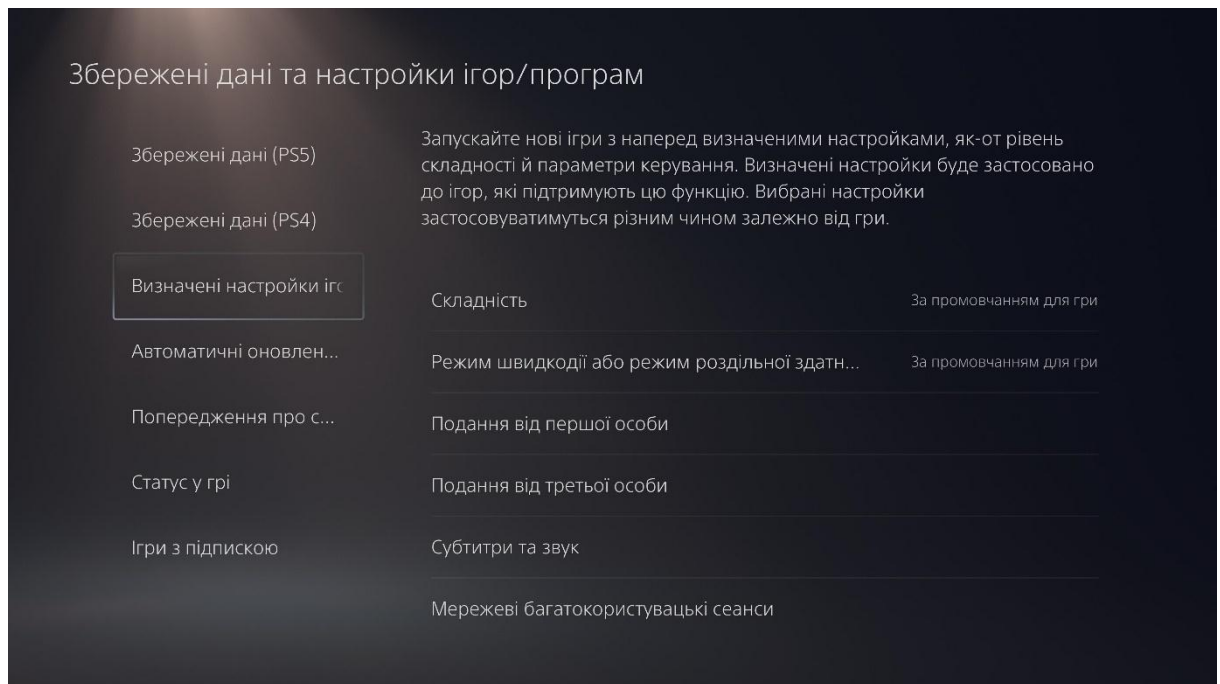


Рис.5.

Висновки до Розділу 1

У першому розділі було розглянуто теоретичні основи автоматизованого керування роботизованими платформами, що є фундаментом для подальшого аналізу та практичної реалізації систем керування. Встановлено, що роботизовані платформи являють собою складні апаратно-програмні комплекси, функціонування яких базується на тісній взаємодії механічних, електронних та програмних підсистем.

Проаналізовано поняття автоматизованого керування та визначено його ключову роль у забезпеченні стабільності, точності та адаптивності роботизованих систем. Показано, що роботизовані платформи як об'єкти керування характеризуються багатоканальністю, нелінійністю та необхідністю роботи в режимі реального часу, що зумовлює застосування як класичних, так і інтелектуальних алгоритмів керування.

Розглянуто структуру системи автоматизованого керування, яка має багаторівневу архітектуру та включає рівні сприйняття, прийняття рішень і виконавчих механізмів. Особливу увагу приділено ролі людино-машинного інтерфейсу, який забезпечує ефективну взаємодію оператора з роботизованою платформою та суттєво впливає на якість керування.

Обґрунтовано доцільність використання ігрових контролерів, зокрема контролерів системи PlayStation, як засобів введення керуючих команд у робототехнічних системах. Їх ергономічність, наявність багатоканальних аналогових органів керування та підтримка стандартних інтерфейсів роблять такі пристрої ефективним інструментом дистанційного керування роботизованими платформами.

Показано значення програмних засобів у автоматизованому керуванні, зокрема мови програмування Python, яка забезпечує гнучку обробку сигналів, реалізацію алгоритмів керування та інтеграцію апаратних контролерів. Поєднання апаратних засобів керування та програмної обробки даних дозволяє досягти оптимального балансу між простотою реалізації, функціональністю та адаптивністю системи.

Отримані у розділі теоретичні положення створюють науково-методичну основу для подальшого дослідження інтеграції контролерів керування системи PlayStation та Python-скриптів у системи автоматизованого керування роботизованими платформами, що буде розглянуто в наступних розділах роботи.

РОЗДІЛ 2

КОНТРОЛЕРИ УПРАВЛІННЯ СИСТЕМИ PLAYSTATION ЯК ІНТЕРФЕЙС КЕРУВАННЯ

2.1. Загальна характеристика контролерів управління системи PlayStation

Контролери управління системи PlayStation є багатофункціональними пристроями введення, призначеними для інтерактивної взаємодії користувача з ігровими та прикладними системами. Завдяки високому рівню ергономіки, надійності та широкому набору сенсорних елементів, дані контролери набули популярності не лише у сфері відеоігор, але й у прикладних галузях, зокрема робототехніці, мехатроніці та системах дистанційного керування.



Рис.6.

Контролери серій DualShock та DualSense оснащені аналоговими стиками, цифровими кнопками, сенсорними панелями, а також датчиками руху, що

дозволяє використовувати їх як універсальний інтерфейс керування складними технічними об'єктами. Важливою особливістю є підтримка двостороннього зв'язку між контролером і керованою системою, що відкриває можливості для реалізації зворотного тактильного зв'язку.

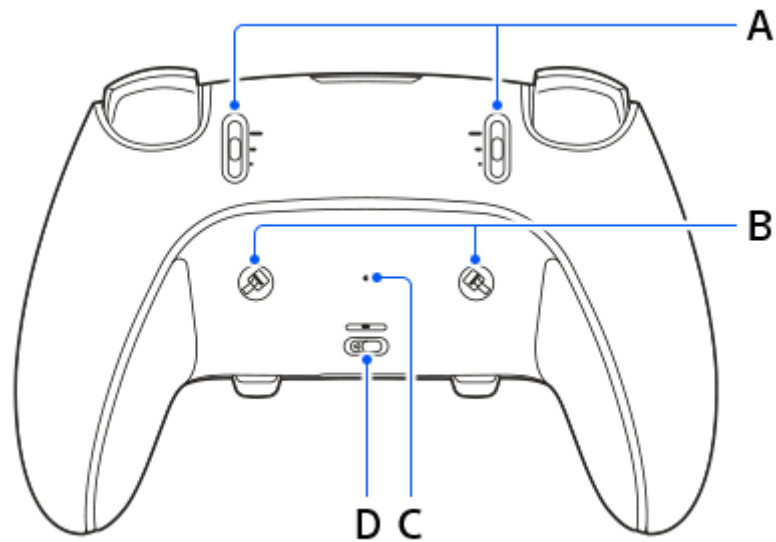


Рис.7.

Контролери PlayStation є універсальними пристроями введення, розробленими для високоточної та інтуїтивної взаємодії користувача з цифровими системами. Контролери серій DualShock і DualSense вирізняються ергономічною формою, великою кількістю органів керування та підтримкою аналогових і цифрових сигналів, що робить їх придатними для використання у систем керування роботизованими платформами.

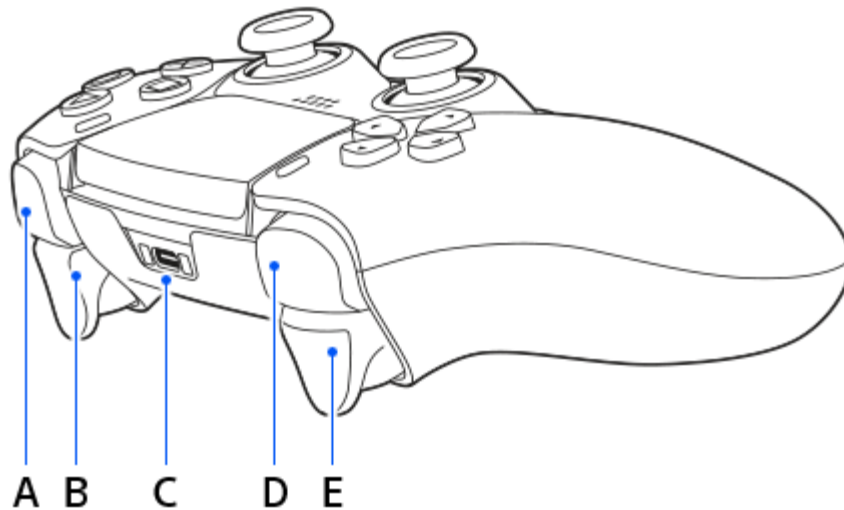


Рис.8.

DualShock є класичним контролером, який забезпечує базовий набір функцій, тоді як DualSense (PS5) має розширені можливості, зокрема адаптивні тригери та розвинену тактильну віддачу. Це дозволяє реалізувати зворотний зв'язок між роботизованою платформою та оператором, що є важливим у напіваавтономних системах керування.

Контролери PlayStation є ключовим елементом екосистеми ігрових консолей Sony Interactive Entertainment та відіграють важливу роль у взаємодії користувача з програмно-апаратним середовищем. Протягом кількох поколінь компанія Sony постійно вдосконалювала конструкцію, функціональність і сенсорні можливості контролерів, що призвело до появи таких знакових моделей, як **DualShock** та **DualSense**.

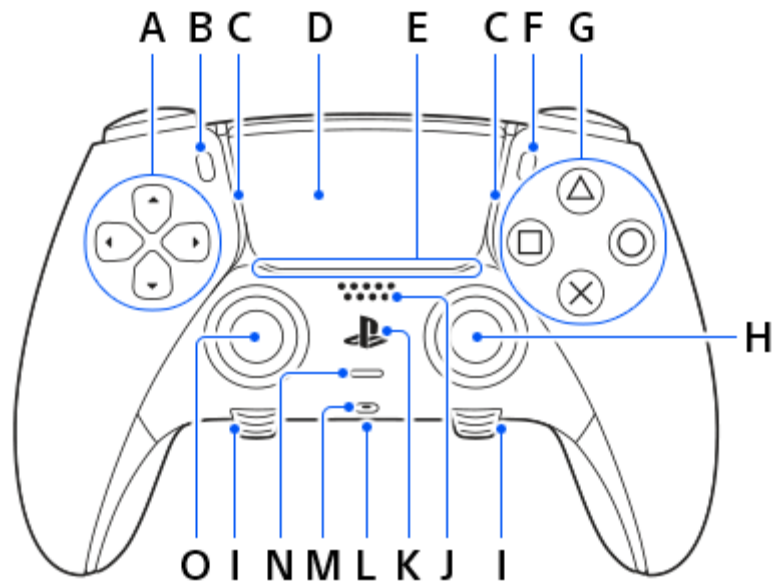


Рис.9.

Контролер DualShock

DualShock — це серія геймпадів, яка вперше була представлена компанією Sony у 1997 році та набула широкого поширення з виходом консолі **PlayStation One**, а згодом — **PlayStation 2**, **PlayStation 3** та **PlayStation 4** (DualShock 4).

Основні характеристики DualShock:

- симетричне розташування аналогових стіків;
- наявність цифрової хрестовини (D-pad);
- стандартний набір кнопок керування (Δ , $\circ$, $\times$, $\square$);
- два аналогові тригери (L2, R2) та плечові кнопки (L1, R1);
- вбудовані віброттори (функція зворотного зв'язку);

- підтримка дротового (USB) та бездротового (Bluetooth) з'єднання.



Рис.10.

У версії **DualShock 4** було додано:

- сенсорну панель (touchpad);
- гіроскоп і акселерометр;
- світлодіодну індикацію;
- вбудований динамік;
- роз'єм для навушників.

Завдяки універсальності, відносно простій архітектурі та відкритим протоколам обміну даними DualShock широко використовується не лише в іграх, але й у сфері **робототехніки, симуляторів, VR/AR-систем та навчальних проєктів**, зокрема у поєднанні з мовами програмування Python, C++ та середовищами ROS.

Контролер DualSense

DualSense — це контролер нового покоління, представлений разом із консоллю **PlayStation 5**. Він є еволюційним розвитком концепції DualShock, але з істотно розширеними функціональними та сенсорними можливостями.

Основні особливості DualSense:

- **адаптивні тригери (L2, R2)** з регульованим опором;
- **покращена тактильна віддача (haptic feedback)** замість класичної вібрації;
- високоточні датчики руху;
- вбудований мікрофон;
- покращена ергономіка корпусу;
- USB Type-C для заряджання та передачі даних;
- підтримка Bluetooth нового покоління.

DualSense забезпечує значно вищий рівень імерсивності, дозволяючи передавати користувачу складні тактильні сигнали, що імітують текстуру поверхонь, силу опору або динаміку руху. Це відкриває нові можливості не лише для ігор, а й для **керування роботизованими системами, тренажерами та кіберфізичними пристроями.**

Порівняльна характеристика DualShock і DualSense

Характеристика	DualShock	DualSense
Покоління консолі	PS1–PS4	PS5
Тип вібрації	Класична	Тактильна (Haptic)
Тригери	Аналогові	Адаптивні

Характеристика	DualShock	DualSense
Сенсорна панель	Так	Так (покращена)
Мікрофон	Ні	Так
Інтерфейс	USB, Bluetooth USB-C, Bluetooth	

Значення контролерів PlayStation у технічних дослідженнях

Завдяки високій надійності, точності сенсорів та широкій підтримці програмного забезпечення, контролери DualShock і DualSense активно використовуються в:

- дослідженнях людино-машинної взаємодії;
- освітніх робототехнічних проєктах;
- системах дистанційного керування;
- експериментальних інтерфейсах керування.

Таким чином, контролери PlayStation є не лише ігровими пристроями, а й багатофункціональними інструментами введення даних, які можуть бути ефективно інтегровані в сучасні інженерні та наукові системи.

2.2. Апаратна архітектура та функціональні можливості контролерів PlayStation

З точки зору апаратної архітектури контролери PlayStation являють собою вбудовані мікроконтролерні системи, що поєднують у собі модулі введення, обробки та передачі даних. Основними елементами контролера є мікроконтролер, аналого-цифрові перетворювачі, сенсори положення та руху, а також модулі бездротового зв'язку.

Аналогові стики дозволяють формувати безперервні керуючі сигнали, що є надзвичайно важливим для плавного керування швидкістю та напрямком руху роботизованих платформ. Наявність гіроскопів та акселерометрів забезпечує можливість керування на основі просторових рухів, що значно розширює функціональність інтерфейсу.

Контролер DualSense, який використовується в системі PlayStation 5, додатково оснащений адаптивними тригерами та покращеною системою тактильного зворотного зв'язку. Ці можливості можуть бути використані для інформування оператора про стан роботизованої платформи, зокрема перевантаження приводів або зміну режимів роботи.

Апаратна архітектура контролерів PlayStation базується на вбудованому мікроконтролері, який обробляє сигнали з аналогових стиків, кнопок, сенсорної панелі та інерціальних датчиків. Контролери оснащені акселерометрами та гіроскопами, що дозволяє фіксувати просторові переміщення та орієнтацію.

Сенсорні можливості контролерів дозволяють використовувати їх не лише як класичний пульт, а і як багатовимірний інтерфейс керування, що особливо актуально для мобільних роботів, маніпуляторів та дронів.

Апаратна архітектура та функціональні можливості контролерів PlayStation

Контролери PlayStation (DualShock та DualSense) є складними апаратно-програмними пристроями введення, що поєднують у собі мікроконтролерну систему, сенсорні модулі, інтерфейси зв'язку та елементи зворотного зв'язку. Їх архітектура розроблена з урахуванням вимог високої точності, низької затримки та ергономічної взаємодії з користувачем.

Апаратна архітектура контролерів PlayStation

Мікроконтролерний модуль

Центральним елементом апаратної архітектури контролерів PlayStation є вбудований мікроконтролер (MCU), який виконує такі функції:

- зчитування сигналів з аналогових і цифрових елементів керування;
- обробка даних із сенсорів руху;
- формування пакетів даних для передачі через інтерфейси USB або Bluetooth;
- керування системами тактильного зворотного зв'язку;
- контроль енергоспоживання та заряджання акумулятора.

У контролерах DualShock 4 і DualSense використовуються високопродуктивні мікроконтролери з ядром ARM, що забезпечують обробку даних у режимі реального часу та стабільну роботу при високих частотах опитування.

Сенсорні елементи введення

Контролери PlayStation містять велику кількість сенсорних компонентів, які забезпечують багатовимірне введення даних:

- Аналогові стіки — побудовані на основі потенціометрів або магніторезистивних датчиків, дозволяють визначати положення у двох осях (X, Y).
- Тригери (L2, R2) — аналогові елементи з можливістю вимірювання сили натискання.
- Кнопки — цифрові входи з миттєвою реакцією.

- Сенсорна панель (Touchpad) — ємнісний сенсор, що підтримує мультиточковий.
- Датчики руху — тривісний акселерометр та гіроскоп для визначення орієнтації та руху в просторі.

У DualSense точність сенсорів була підвищена, що дало змогу реалізувати складні сценарії керування та імерсивної взаємодії.

Система тактильного зворотного зв'язку

Однією з ключових функціональних відмінностей між DualShock і DualSense є реалізація зворотного зв'язку:

- DualShock використовує два ексцентрикові вібромотори (один високочастотний, один низькочастотний).
- DualSense оснащений лінійними актуаторами, які забезпечують точну тактильну віддачу з можливістю формування різних профілів вібрації.

Це дозволяє передавати користувачеві інформацію про силу, текстуру та інші фізичні характеристики віртуального середовища.

Адаптивні тригери (DualSense)

Контролер DualSense має спеціальні електромеханічні механізми у тригерах L2 та R2, які:

- змінюють опір натискання в режимі реального часу;
- імітують фізичні обмеження або навантаження;
- керуються програмно через інтерфейси консолі або ПК.

Такі тригери особливо корисні для симуляторів, робототехнічних тренажерів та систем дистанційного керування.

Система живлення

Контролери PlayStation оснащені вбудованими літій-іонними акумуляторами з такими функціями:

- заряджання через USB (micro-USB або USB Type-C);
- апаратний контроль рівня заряду;
- режими енергозбереження при простій.

DualSense має збільшену ємність акумулятора та оптимізоване енергоспоживання.

Функціональні можливості контролерів PlayStation

Обробка та передача даних

Контролери підтримують:

- дротову передачу даних через USB з мінімальною затримкою;
- бездротову передачу через Bluetooth із шифруванням та стабільною синхронізацією;
- частоту опитування до кількох сотень Гц.

Це дозволяє використовувати контролери у системах реального часу.

Інтеграція з програмними системами

Контролери PlayStation можуть інтегруватися з:

- операційними системами Windows, Linux, macOS;
- мовами програмування Python, C/C++;
- фреймворками робототехніки (ROS, OpenCV, PyGame).

Це робить їх універсальними пристроями введення для інженерних застосувань.

Додаткові функції

До функціональних можливостей також належать:

- світлодіодна індикація станів;
- вбудований динамік;
- мікрофон (DualSense);
- аудіовихід для навушників;
- оновлення прошивки контролера.

Апаратна архітектура контролерів PlayStation поєднує в собі сучасні мікроконтролерні технології, високоточні сенсори та ефективні системи зворотного зв'язку. Це забезпечує не лише комфортну ігрову взаємодію, але й можливість застосування цих контролерів у наукових, навчальних та інженерних проєктах, зокрема в робототехніці та системах людино-машинної взаємодії.

2.3. Протоколи та інтерфейси зв'язку контролерів PlayStation

Для передачі даних між контролером PlayStation та керуючою системою використовуються стандартні протоколи зв'язку, зокрема USB та Bluetooth. Застосування універсальних інтерфейсів забезпечує сумісність контролерів з різними обчислювальними платформами, включаючи персональні комп'ютери, одноплатні мікрокомп'ютери та вбудовані системи.

При використанні USB-з'єднання контролер визначається системою як пристрій класу HID (Human Interface Device), що дозволяє зчитувати дані без необхідності встановлення спеціалізованих драйверів. Бездротовий зв'язок через Bluetooth забезпечує більшу мобільність, однак вимагає додаткових механізмів синхронізації та захисту передачі даних.

Надійність та низька затримка передачі даних є критично важливими характеристиками для систем керування роботами в реальному часі. Саме тому вибір способу підключення контролера має значний вплив на загальну ефективність системи керування.

Контролери PlayStation підтримують дротовий (USB) та бездротовий (Bluetooth) режими зв'язку. USB-з'єднання забезпечує мінімальну затримку та високу стабільність передачі даних, що є критично важливим для керування у реальному часі. Bluetooth-з'єднання забезпечує мобільність оператора, проте потребує додаткової обробки можливих затримок.

В операційних системах контролери визначаються як пристрої класу HID, що дозволяє працювати з ними без спеціалізованих драйверів.

Протоколи та інтерфейси зв'язку контролерів PlayStation

Контролери PlayStation (DualShock та DualSense) використовують сучасні дротові та бездротові інтерфейси зв'язку для обміну даними з ігровою консоллю або персональним комп'ютером. Вибір протоколів обумовлений необхідністю забезпечення високої швидкості передачі даних, мінімальної затримки, енергоефективності та стабільності з'єднання.

Дротовий інтерфейс USB

Загальна характеристика

Контролери PlayStation підтримують дротове підключення через інтерфейс USB, який використовується для:

- передачі даних керування;
- заряджання вбудованого акумулятора;
- оновлення вбудованого програмного забезпечення (прошивки).

Залежно від покоління контролера застосовуються різні типи роз'ємів:

- Mini-USB / Micro-USB — у ранніх версіях DualShock;
- USB Type-C — у контролері DualSense.

Протокол USB HID

Передача даних через USB здійснюється з використанням класу пристроїв HID (Human Interface Device). Цей протокол забезпечує:

- автоматичне розпізнавання контролера операційною системою;
- відсутність необхідності у встановленні додаткових драйверів;
- стандартизований формат обміну даними.

Контролер надсилає HID-звіти (HID reports), які містять:

- стани кнопок;
- координати аналогових стіків;
- значення тригерів;
- дані з акселерометра та гіроскопа;

- службову інформацію (рівень заряду, статус з'єднання).

Переваги USB-з'єднання

- мінімальна затримка передачі сигналів;
- висока стабільність зв'язку;
- одночасна передача даних і живлення;
- зручність для розробки та тестування програмних систем.

Бездротовий інтерфейс Bluetooth

Загальна характеристика Bluetooth

Для бездротового з'єднання контролери PlayStation використовують стандарт Bluetooth, який забезпечує мобільність та зручність користування.

Контролери підтримують різні версії Bluetooth залежно від моделі:

- DualShock 4 — Bluetooth 2.1/4.0;
- DualSense — Bluetooth 5.x.

Протокол Bluetooth HID

Як і у випадку з USB, бездротова передача даних здійснюється за допомогою профілю Bluetooth HID. Він дозволяє:

- передавати дані керування в режимі реального часу;
- використовувати стандартні засоби підключення в операційних системах;
- забезпечувати сумісність із широким спектром пристроїв.

Передані пакети містять аналогічну інформацію, що й при USB-підключенні, але можуть мати дещо більшу затримку через бездротовий канал.

Особливості Bluetooth-з'єднання

- наявність процедури спарювання (pairing);
- підтримка шифрування даних;
- динамічне регулювання енергоспоживання;
- можливі перешкоди в умовах зашумленого радіоефіру.

Частота опитування та затримки

Контролери PlayStation працюють із високою частотою опитування, що є критичним параметром для ігрових і робототехнічних застосувань:

- частота опитування через USB може досягати 250–1000 Гц;
- через Bluetooth — зазвичай 125–250 Гц.

Низька затримка забезпечує точну та передбачувану реакцію системи на дії користувача.

Приватні та розширені протоколи Sony

Окрім стандартних протоколів HID, компанія Sony використовує власні розширення протоколів, які дозволяють:

- керувати адаптивними тригерами DualSense;
- передавати складні профілі тактильного зворотного зв'язку;

- оновлювати прошивку контролера;
- здійснювати автентифікацію з консоллю PlayStation.

Ці протоколи частково документовані та активно досліджуються спільнотою розробників.

Інтерфейси взаємодії з програмним забезпеченням

Контролери PlayStation можуть взаємодіяти з програмним забезпеченням через:

- стандартні драйвери HID в ОС;
- бібліотеки для роботи з контролерами (SDL, PyGame, evdev);
- спеціалізовані API та драйвери (наприклад, DS4Windows).

Це дозволяє використовувати контролери не лише для ігор, але й для:

- керування роботами;
- симуляції процесів;
- наукових експериментів;
- навчальних систем.

Порівняння інтерфейсів USB та Bluetooth

Критерій	USB	Bluetooth
Тип з'єднання	Дротовий	Бездротовий
Затримка	Мінімальна	Вища

Критерій	USB	Bluetooth
Стабільність	Висока	Залежить від умов
Мобільність	Обмежена	Висока
Живлення	Так	Ні

Протоколи та інтерфейси зв'язку контролерів PlayStation поєднують у собі стандартизовані рішення (USB HID, Bluetooth HID) та фірмові розширення Sony. Такий підхід забезпечує високу сумісність, надійність і функціональність, що дозволяє використовувати контролери як універсальні пристрої введення в ігрових, інженерних та науково-дослідних системах.

2.4. Використання контролерів PlayStation у систем керування роботизованими платформами

Використання контролерів PlayStation у робототехнічних системах дозволяє реалізувати інтуїтивно зрозумілий та гнучкий інтерфейс керування. Кнопки та аналогові органи керування можуть бути відображені на відповідні функції платформи, такі як рух, поворот, керування маніпуляторами або перемикання режимів роботи.

Застосування таких контролерів є особливо доцільним у систем напіваавтономного керування, де оператор здійснює корекцію дій робота або бере керування на себе у складних умовах. Завдяки ергономічній формі та зручному розташуванню елементів керування зменшується втома оператора та підвищується точність виконання команд.

Крім того, контролери PlayStation можуть використовуватися як засіб навчання та тестування алгоритмів керування, оскільки вони забезпечують швидке налаштування та не потребують складної апаратної інтеграції.

Зчитування сигналів з контролерів здійснюється шляхом програмного опитування аналогових і цифрових каналів. Аналогові стики формують значення в діапазоні від -1 до $+1$, що дозволяє реалізувати пропорційне керування швидкістю та напрямком руху.

Python-бібліотеки забезпечують доступ до даних контролера у режимі реального часу, що дозволяє інтегрувати їх у системи автоматизованого керування без складної апаратної адаптації.

Використання контролерів PlayStation у системах керування роботизованими платформами

Сучасні роботизовані платформи потребують зручних, надійних та інтуїтивно зрозумілих засобів керування. Контролери PlayStation (DualShock і DualSense), спочатку розроблені для ігрових систем, завдяки своїй багатій сенсорній оснащеності, ергономічному дизайну та відкритим протоколам зв'язку, набули широкого застосування у сфері робототехніки, зокрема для керування мобільними роботами, маніпуляторами та автономними платформами.

Переваги використання контролерів PlayStation у робототехніці

Основними причинами застосування контролерів PlayStation у роботизованих системах є:

- ергономічність — форма контролера адаптована для тривалого використання оператором;

- велика кількість органів керування — аналогові стіки, тригери, кнопки та сенсорна панель;
- аналогове керування — можливість плавного регулювання швидкості та напрямку руху;
- наявність сенсорів руху — використання акселерометра та гіроскопа для альтернативних способів керування;
- доступність і низька вартість порівняно зі спеціалізованими промисловими пультами.

Ці фактори роблять контролери PlayStation ефективним інтерфейсом людино-машинної взаємодії.

Архітектура системи керування роботом з використанням контролера PlayStation

Типова структура системи керування включає такі компоненти:

1. Контролер PlayStation — пристрій введення команд оператора.
2. Обчислювальний модуль (ПК, одноплатний комп'ютер, мікроконтролер) — приймає та обробляє сигнали керування.
3. Комунікаційний інтерфейс — USB або Bluetooth для передачі даних.
4. Програмне забезпечення керування — реалізує алгоритми перетворення сигналів у команди керування.
5. Виконавчі механізми робота — електродвигуни, серводвигуни, приводи.

У такій архітектурі контролер виступає джерелом команд, а програмний рівень забезпечує їх інтерпретацію та передачу до апаратної частини робота.

Реалізація керування рухом роботизованих платформ

Найпоширенішим застосуванням контролерів PlayStation є керування рухом мобільних роботів. Типові сценарії включають:

- керування швидкістю та напрямком руху за допомогою аналогових стіків;
- окреме керування лівою та правою сторонами платформи (диференціальне керування);
- керування поворотом і прискоренням з використанням тригерів;
- перемикання режимів роботи кнопками контролера.

Завдяки аналоговим входам оператор може здійснювати точне та плавне керування, що є важливим для маневрування у складних умовах.

Керування маніпуляторами та роботизованими руками

Контролери PlayStation широко застосовуються для керування багатоступеневими роботизованими маніпуляторами:

- один аналоговий стік керує рухом у горизонтальній площині;
- інший — вертикальним переміщенням;
- тригери використовуються для регулювання сили захвату;
- кнопки відповідають за вибір активного суглоба або режиму керування.

У DualSense адаптивні тригери можуть імітувати зростання навантаження, що підвищує точність і безпеку керування.

Використання сенсорів руху та тактильного зворотного зв'язку

Акселерометр і гіроскоп контролерів дозволяють реалізувати жестове керування роботизованими платформами. Наприклад:

- нахил контролера відповідає зміні напрямку руху;
- обертання — керуванню поворотом маніпулятора.

Система тактильного зворотного зв'язку може використовуватися для:

- сигналізації про перевищення навантаження;
- індикації перешкод;
- підтвердження виконання команд.

Це значно покращує зворотний зв'язок між оператором і роботом.

Програмна реалізація керування

Для інтеграції контролерів PlayStation у робототехнічні системи застосовуються:

- мови програмування Python, C, C++;
- бібліотеки роботи з контролерами (PyGame, evdev, SDL);
- фреймворки робототехніки, зокрема ROS (Robot Operating System).

Програмне забезпечення виконує:

- зчитування HID-пакетів;
- фільтрацію шумів;
- масштабування сигналів;

- реалізацію алгоритмів безпеки (dead zone, обмеження швидкості).

Обмеження та проблеми використання

Попри численні переваги, використання контролерів PlayStation має певні обмеження:

- можливі затримки при Bluetooth-з'єднанні;
- відсутність сертифікації для промислового використання;
- залежність від заряду акумулятора;
- обмежений радіус дії бездротового зв'язку.

Тому в критично важливих системах контролери використовуються переважно на рівні операторського інтерфейсу.

Контролери PlayStation є ефективним, доступним і гнучким інструментом для керування роботизованими платформами. Їх багаті функціональні можливості, поєднані з простотою інтеграції у сучасні програмні середовища, дозволяють використовувати ці пристрої у навчальних, дослідницьких та прикладних робототехнічних системах.

2.5. Переваги та обмеження використання контролерів PlayStation як інтерфейсу керування

До основних переваг використання контролерів PlayStation у систем керування роботизованими платформами належать висока ергономічність, доступність, надійність та багатофункціональність. Контролери забезпечують точне зчитування аналогових сигналів і підтримують різні режими взаємодії з оператором.

Водночас існують і певні обмеження. Зокрема, контролери PlayStation не є промисловими пристроями, що може обмежувати їх використання у жорстких умовах експлуатації. Також необхідною є програмна адаптація сигналів контролера до конкретних вимог системи керування.

Незважаючи на зазначені обмеження, застосування контролерів PlayStation як інтерфейсу керування є перспективним напрямом розвитку систем автоматизованого керування, особливо у дослідницьких та навчальних проєктах.

Перевагами контролерів PlayStation є ергономічність, доступність, багатофункціональність і простота інтеграції. Основними обмеженнями є непристосованість до жорстких промислових умов та залежність від програмної обробки сигналів.

Переваги та обмеження використання контролерів PlayStation як інтерфейсу керування

Контролери PlayStation (DualShock і DualSense) дедалі частіше застосовуються не лише в ігрових системах, але й як універсальні інтерфейси керування в робототехніці, симуляторах та експериментальних кіберфізичних системах. Їх використання має як значні переваги, так і певні обмеження, які необхідно враховувати під час проєктування систем керування.

Переваги використання контролерів PlayStation

Ергономічність та зручність експлуатації

Контролери PlayStation розроблені з урахуванням анатомії рук користувача, що забезпечує:

- комфортне тривале використання без значної втоми;
- інтуїтивно зрозуміле розташування органів керування;

- швидке навчання операторів без спеціальної підготовки.

Це особливо важливо для операторських систем дистанційного керування.

Велика кількість органів керування

Контролери містять:

- два аналогові стіки;
- аналогові тригери;
- цифрові кнопки;
- сенсорну панель;
- датчики руху.

Така кількість елементів дозволяє реалізувати багатоканальне керування складними роботизованими системами без необхідності додаткових пристроїв.

Аналогове та плавне керування

Аналогові входи забезпечують:

- точне регулювання швидкості;
- контроль сили та положення;
- плавні переходи між режимами руху.

Це є суттєвою перевагою порівняно з дискретними клавіатурними інтерфейсами.

Наявність сенсорів руху та зворотного зв'язку

Контролери PlayStation оснащені:

- акселерометрами та гіроскопами для жестового керування;
- вібраційною або тактильною системою зворотного зв'язку;
- адаптивними тригерами (DualSense).

Ці можливості дозволяють реалізувати ефективний двосторонній обмін інформацією між оператором і системою керування.

Доступність та економічна доцільність

Порівняно зі спеціалізованими промисловими пультами керування, контролери PlayStation:

- є масовими та доступними;
- мають нижчу вартість;
- не потребують складного технічного обслуговування.

Це робить їх привабливими для навчальних та дослідницьких проєктів.

Простота інтеграції з програмним забезпеченням

Завдяки підтримці стандартів USB HID і Bluetooth HID контролери:

- сумісні з більшістю операційних систем;
- легко інтегруються з Python, C/C++;
- підтримуються популярними бібліотеками та фреймворками (ROS, SDL).

Обмеження використання контролерів PlayStation

Відсутність промислової сертифікації

Контролери PlayStation не призначені для:

- експлуатації в промислових умовах;
- роботи у середовищах з підвищеною вібрацією, пилом або вологістю;
- виконання критично важливих функцій без додаткових систем безпеки.

Обмежена надійність бездротового з'єднання

Використання Bluetooth може призводити до:

- затримок у передачі сигналів;
- втрати пакунків даних;
- впливу радіоперешкод.

У відповідальних застосуваннях зазвичай надається перевага дротовому підключенню.

Залежність від заряду акумулятора

Контролери мають вбудований акумулятор, що:

- потребує регулярного заряджання;
- може обмежувати тривалість безперервної роботи;
- створює ризик раптового вимкнення.

Обмежена точність та знос механічних елементів

Аналогові стіки та тригери:

- схильні до механічного зношування;
- можуть мати дрейф нульового положення;
- поступаються за точністю професійним індустріальним енкодерам.

Закритість окремих протоколів

Деякі функції (наприклад, адаптивні тригери DualSense):

- використовують фірмові протоколи Sony;
- мають обмежену документацію;
- вимагають сторонніх драйверів або реверс-інжинірингу.

Порівняльна оцінка

Критерій	Контролери PlayStation	Промислові пульти
Вартість	Низька	Висока
Ергономіка	Висока	Середня
Надійність	Середня	Висока
Сертифікація	Відсутня	Наявна
Гнучкість	Висока	Обмежена

Контролери PlayStation є ефективним і гнучким інтерфейсом керування для навчальних, дослідницьких та експериментальних систем. Вони поєднують ергономічність, функціональність і доступність, проте мають обмеження щодо надійності та сертифікації, що унеможлиблює їх пряме використання в критично важливих промислових системах без додаткових заходів безпеки.

Висновки до Розділу 2

У другому розділі було здійснено комплексний аналіз контролерів управління системи PlayStation як інтерфейсу керування роботизованими платформами. Розглянуто загальні характеристики, апаратну архітектуру, функціональні можливості, протоколи зв'язку, а також практичні аспекти використання контролерів DualShock і DualSense у робототехнічних системах.

Встановлено, що контролери PlayStation є багатофункціональними пристроями введення, які поєднують ергономічний дизайн, велику кількість аналогових і цифрових органів керування, інерціальні сенсори та засоби зворотного тактильного зв'язку. Це робить їх ефективним людино-машинним інтерфейсом для дистанційного та напіваавтономного керування роботизованими платформами.

Аналіз апаратної архітектури показав, що контролери PlayStation базуються на сучасних мікроконтролерних рішеннях із підтримкою високої частоти опитування, що забезпечує роботу в режимі реального часу. Наявність аналогових стиків, акселерометрів, гіроскопів, сенсорної панелі та адаптивних тригерів (у DualSense) дозволяє реалізувати багатовимірне та інтуїтивне керування складними рухами мобільних роботів і маніпуляторів.

Дослідження протоколів та інтерфейсів зв'язку підтвердило, що використання стандартів USB HID і Bluetooth HID забезпечує високу сумісність контролерів з різними операційними системами та програмними середовищами. Дротове

USB-з'єднання є найбільш доцільним для систем реального часу завдяки мінімальній затримці, тоді як Bluetooth-з'єднання підвищує мобільність оператора, але потребує додаткового врахування затримок і стабільності каналу зв'язку.

Показано, що контролери PlayStation широко застосовуються у навчальних, дослідницьких та експериментальних робототехнічних проєктах. Вони дозволяють швидко реалізувати системи керування рухом, маніпуляторами та допоміжними функціями без складної апаратної інтеграції, особливо у поєднанні з високорівневими мовами програмування та фреймворками робототехніки.

Разом з тим визначено низку обмежень використання контролерів PlayStation, зокрема відсутність промислової сертифікації, залежність від стану акумулятора, можливі затримки бездротового зв'язку та обмежену довговічність механічних елементів. Це зумовлює доцільність їх застосування переважно на рівні операторського інтерфейсу або в некритичних системах.

Узагальнюючи, можна зробити висновок, що контролери PlayStation є доступним, гнучким та функціонально насиченим інтерфейсом керування, який доцільно використовувати у системах автоматизованого керування роботизованими платформами навчального, дослідницького та прикладного призначення. Отримані в розділі результати створюють підґрунтя для подальшого розгляду програмної інтеграції контролерів PlayStation з Python-скриптами та практичної реалізації систем керування, що буде представлено в наступних розділах роботи.

РОЗДІЛ 3

МОВА ПРОГРАМУВАННЯ PYTHON У СИСТЕМАХ АВТОМАТИЗОВАНОГО КЕРУВАННЯ

3.1. Загальна характеристика мови програмування Python

Мова програмування Python є високорівневою інтерпретованою мовою загального призначення, яка широко застосовується у сучасних системах автоматизованого керування завдяки своїй простоті, гнучкості та розширюваності. Основною перевагою Python є зрозумілий синтаксис, що значно знижує поріг входження та скорочує час розробки програмних рішень.

У контексті автоматизованого керування Python використовується як засіб швидкої реалізації алгоритмів обробки даних, керування пристроями та інтеграції різнорідних програмно-апаратних компонентів. Завдяки відкритій архітектурі та великій кількості бібліотек Python дозволяє створювати масштабовані та модульні системи керування роботизованими платформами.

Python широко використовується у робототехніці завдяки простоті синтаксису, кросплатформності та великій кількості бібліотек. Він дозволяє швидко створювати прототипи систем керування та адаптувати їх до різних платформ.

3.2. Застосування Python у робототехнічних системах

Python набув широкого застосування у робототехніці як на рівні прототипування, так і в готових системах керування. Він використовується для реалізації алгоритмів руху, обробки сенсорних даних, планування траєкторій та взаємодії з користувачем. Особливою нагодою для використання Python є необхідність швидкої адаптації системи до змінних умов експлуатації.

Мова Python активно застосовується у рамок робототехнічних фреймворків, таких як Robot Operating System (ROS), де вона виконує роль інструмента для створення керуючих вузлів та сценаріїв автоматизації. Завдяки цьому Python стає ключовим елементом у систем, що поєднують автономні та напівавтономні режими керування.

Бібліотеки `pygame`, `inputs`, `evdev` та `hidapi` забезпечують доступ до сигналів контролерів PS. Вони дозволяють зчитувати аналогові осі, кнопки та сенсорні дані у режимі реального часу.

3.3. Бібліотеки Python для систем автоматизованого керування

Важливою перевагою Python є наявність спеціалізованих бібліотек, що забезпечують ефективну роботу з апаратними пристроями та сигналами керування. Для взаємодії з контролерами введення, зокрема контролерами PlayStation, використовуються бібліотеки `pygame`, `inputs`, `evdev` та `hidapi`, які дозволяють зчитувати дані у режимі реального часу.

Для математичної обробки сигналів та реалізації алгоритмів керування широко застосовуються бібліотеки `NumPy`, `SciPy` та `control`. Вони надають інструменти для аналізу динамічних систем, фільтрації сигналів та реалізації регуляторів. Для візуалізації результатів керування використовуються бібліотеки `Matplotlib` та `Plotly`, що дозволяє здійснювати моніторинг стану роботизованої платформи.

Python-скрипти виступають проміжною ланкою між контролером та апаратною частиною робота, перетворюючи сигнали введення у керуючі команди для приводів.

3.4. Реалізація алгоритмів керування на Python

Python дозволяє реалізовувати різноманітні алгоритми керування, включаючи класичні та інтелектуальні методи. Зокрема, на основі Python можуть бути реалізовані пропорційно-інтегрально-диференціальні регулятори, алгоритми стабілізації руху, а також системи адаптивного керування.

Використання Python у реальному часі потребує оптимізації програмного коду та раціонального використання обчислювальних ресурсів. Це досягається шляхом застосування багатопоточності, асинхронного програмування та використання модулів для роботи з низькорівневими інтерфейсами. Такий підхід дозволяє забезпечити стабільну роботу системи керування навіть при значних обчислювальних навантаженнях.

Для роботи у реальному часі застосовуються циклічні алгоритми, фільтрація сигналів та обмеження керуючих впливів. Це дозволяє уникнути різких змін швидкості та нестабільної поведінки платформи.

3.5. Переваги та обмеження використання Python у систем автоматизованого керування

До основних переваг використання Python у систем автоматизованого керування належать швидкість розробки, висока читабельність коду та широкий вибір бібліотек. Python дозволяє швидко інтегрувати апаратні контролери, реалізувати алгоритми керування та здійснювати тестування системи.

Разом з тим, Python має певні обмеження, пов'язані з інтерпретованою природою мови, що може впливати на продуктивність у жорстких режимах

реального часу. Для подолання цих обмежень часто застосовується гібридний підхід, при якому критично важливі модулі реалізуються мовами низького рівня, а Python використовується як інструмент високорівневої логіки керування.

Таким чином, Python є ефективним та перспективним засобом розробки систем автоматизованого керування роботизованими платформами.

Безпека забезпечується шляхом перевірки допустимих значень сигналів, реалізації аварійної зупинки та контролю стану системи.

Висновки до Розділу 3

У третьому розділі було розглянуто роль мови програмування Python у системах автоматизованого керування роботизованими платформами та обґрунтовано доцільність її використання як основного програмного інструмента для реалізації алгоритмів керування й інтеграції апаратних засобів.

Встановлено, що Python як високорівнева інтерпретована мова характеризується простотою синтаксису, кросплатформністю та високою швидкістю розробки програмних рішень. Ці властивості роблять її особливо придатною для створення, тестування та модифікації систем автоматизованого керування, зокрема на етапах прототипування та експериментальних досліджень.

Показано, що Python широко застосовується у робототехнічних системах для реалізації алгоритмів руху, обробки сенсорних даних, планування траєкторій і організації людино-машинної взаємодії. Інтеграція Python з робототехнічними фреймворками, зокрема ROS, дозволяє створювати модульні та масштабовані

системи керування, які підтримують автономні та напівавтономні режими роботи.

Проаналізовано бібліотеки Python, що використовуються у системах автоматизованого керування. Бібліотеки `pygame`, `inputs`, `evdev` та `hidapi` забезпечують ефективне зчитування сигналів з контролерів PlayStation у режимі реального часу, тоді як `NumPy`, `SciPy` та `control` надають інструменти для математичної обробки сигналів і реалізації алгоритмів керування. Засоби візуалізації, зокрема `Matplotlib`, дозволяють здійснювати моніторинг і аналіз стану роботизованої платформи.

Обґрунтовано можливість реалізації на Python як класичних (PID-регулятори, алгоритми стабілізації), так і адаптивних методів керування. Визначено, що для забезпечення роботи в режимі реального часу необхідним є застосування оптимізації коду, багатопоточності, асинхронних механізмів та обмеження керуючих впливів, що підвищує стабільність і безпеку роботи системи.

Водночас встановлено, що основним обмеженням Python є його інтерпретована природа, яка може знижувати продуктивність у жорстких реальних часових умовах. Для подолання цього недоліку доцільним є використання гібридного підходу, коли критично важливі модулі реалізуються мовами низького рівня, а Python використовується для реалізації високорівневої логіки та взаємодії з користувачем.

Узагальнюючи, можна зробити висновок, що мова програмування Python є ефективним, гнучким і перспективним інструментом для розробки систем автоматизованого керування роботизованими платформами. Її використання у поєднанні з апаратними контролерами та засобами забезпечення безпеки створює надійну основу для побудови сучасних робототехнічних систем, що підтверджує доцільність подальшого практичного впровадження запропонованих підходів.

РОЗДІЛ 4

ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗОВАНОГО КЕРУВАННЯ РОБОТИЗОВАНОЮ ПЛАТФОРМОЮ

4.1. Вимоги до системи автоматизованого керування роботизованою платформою

Проектування системи автоматизованого керування роботизованою платформою починається з формування комплексу функціональних та технічних вимог, які визначають архітектуру системи та принципи її реалізації. Основними вимогами є забезпечення стабільного керування рухом платформи, точного виконання команд оператора, а також можливість роботи в режимі реального часу.

До ключових вимог також належать надійність, масштабованість та адаптивність системи керування. Система повинна підтримувати різні режими роботи, зокрема ручний, напівавтономний та автоматичний, а також забезпечувати безпечну взаємодію з оператором. Важливим аспектом є можливість інтеграції зовнішніх пристроїв введення, таких як контролери управління системи PlayStation, та програмних модулів, реалізованих мовою Python.

Система повинна бути стабільною, масштабованою, безпечною та здатною працювати у реальному часі.

4.2. Архітектура програмно-апаратного комплексу системи керування

Система автоматизованого керування роботизованою платформою має багаторівневу архітектуру, що включає апаратний рівень, програмний рівень та рівень взаємодії з оператором. Апаратний рівень охоплює виконавчі

механізми, сенсорні пристрої та контролери керування приводами. Програмний рівень відповідає за обробку даних, реалізацію алгоритмів керування та координацію роботи системи.

Інтерфейс взаємодії з оператором реалізується за допомогою контролера PlayStation, який забезпечує зручне та інтуїтивне введення керуючих команд. Python-скрипти виконують функцію проміжної ланки між контролером та апаратними компонентами платформи, здійснюючи перетворення сигналів введення у керуючі впливи для виконавчих механізмів.

Така архітектура дозволяє легко модифікувати систему, додавати нові функції та адаптувати її до різних типів роботизованих платформ.

Архітектура включає контролер введення, обчислювальний модуль, виконавчі механізми та сенсорну підсистему.

4.3. Проектування алгоритмів керування рухом роботизованої платформи

Алгоритми керування є центральним елементом системи автоматизованого керування та визначають поведінку роботизованої платформи у процесі експлуатації. Проектування алгоритмів включає розробку математичних моделей руху, вибір методів керування та налаштування параметрів регуляторів.

У межах даної системи керування передбачається використання класичних алгоритмів, таких як PID-регулювання, для стабілізації швидкості та напрямку руху платформи. Керуючі сигнали, отримані від контролера PlayStation, масштабуються та фільтруються з метою усунення шумів та забезпечення плавності руху.

Особлива увага приділяється обробці аналогових сигналів з джойстиків, які дозволяють реалізувати пропорційне керування швидкістю та кутом повороту платформи.

Контролер формує сигнали → Python обробляє → система керування передає команди двигунам.

4.4. Інтеграція контролера PlayStation та Python-скриптів у системи керування

Інтеграція контролера PlayStation у системи автоматизованого керування здійснюється шляхом зчитування сигналів введення через стандартні інтерфейси та їх подальшої обробки Python-скриптами. Для цього використовуються спеціалізовані бібліотеки, що забезпечують доступ до даних контролера у режимі реального часу.

Python-скрипти реалізують логіку керування, обробку подій та взаємодію з апаратними модулями платформи. Вони забезпечують гнучке налаштування відповідності між елементами керування контролера та функціями платформи, що дозволяє адаптувати систему до потреб оператора.

Такий підхід забезпечує високу гнучкість системи керування та спрощує процес її модернізації без необхідності внесення змін у апаратну частину.

Використовуються диференційні алгоритми, пропорційне та PID-керування.

4.5. Забезпечення надійності та безпеки функціонування системи

Одним з важливих аспектів проектування системи автоматизованого керування є забезпечення її надійності та безпеки. Система повинна коректно реагувати на збої зв'язку, помилки введення та несправності апаратних компонентів.

Для цього передбачається реалізація механізмів контролю стану системи, аварійного зупинення платформи та обмеження керуючих впливів. Python-скрипти можуть містити модулі діагностики та логування, що дозволяє здійснювати аналіз роботи системи та своєчасно виявляти потенційні проблеми.

Забезпечення безпечної взаємодії між оператором та роботизованою платформою є необхідною умовою ефективної експлуатації системи автоматизованого керування.

Реалізація включає модульну структуру, головний цикл керування та механізми безпеки.

Висновки до Розділу 4

У четвертому розділі було розглянуто процес проектування системи автоматизованого керування роботизованою платформою з урахуванням вимог до функціональності, надійності, безпеки та роботи в режимі реального часу. Сформульовано основні технічні та функціональні вимоги, які визначають структуру системи та принципи її побудови.

Встановлено, що ефективна система автоматизованого керування повинна забезпечувати стабільний рух роботизованої платформи, точне виконання команд оператора та підтримку різних режимів роботи, зокрема ручного, напіваавтономного й автоматичного. Особливу увагу приділено вимогам до

масштабованості та адаптивності, що дозволяє модифікувати систему без суттєвих змін апаратної частини.

Проаналізовано архітектуру програмно-апаратного комплексу, яка базується на багаторівневому підході та включає контролер введення, обчислювальний модуль, сенсорну підсистему та виконавчі механізми. Показано, що використання Python-скриптів як проміжної ланки між контролером PlayStation та апаратними компонентами платформи забезпечує гнучкість системи та спрощує її розширення і модернізацію.

Обґрунтовано вибір алгоритмів керування рухом роботизованої платформи, зокрема застосування пропорційного та PID-керування для стабілізації швидкості та напрямку руху. Доведено доцільність масштабування та фільтрації сигналів з аналогових органів керування контролера PlayStation, що дозволяє підвищити плавність руху та зменшити вплив шумів і перешкод.

Розглянуто особливості інтеграції контролера PlayStation та Python-скриптів у систему автоматизованого керування. Показано, що використання стандартних інтерфейсів та спеціалізованих бібліотек Python забезпечує зчитування сигналів у режимі реального часу та гнучке налаштування відповідності між елементами керування і функціями роботизованої платформи.

Окрему увагу приділено питанням надійності та безпеки функціонування системи. Визначено, що реалізація механізмів контролю стану, аварійного зупинення, обмеження керуючих впливів і засобів діагностики є необхідною умовою безпечної експлуатації роботизованої платформи. Модульна структура програмного забезпечення та наявність головного циклу керування забезпечують стабільність роботи системи в різних умовах експлуатації.

Узагальнюючи результати розділу, можна зробити висновок, що запропонована архітектура та принципи проєктування системи автоматизованого керування роботизованою платформою забезпечують ефективну, гнучку та безпечну роботу системи. Це створює практичну основу для подальшої реалізації та експериментальної перевірки розроблених рішень у реальних умовах.

РОЗДІЛ 5

ТЕСТУВАННЯ ТА АНАЛІЗ ЕФЕКТИВНОСТІ СИСТЕМИ

5.1. Мета та завдання тестування системи автоматизованого керування

Тестування системи автоматизованого керування роботизованою платформою є необхідним етапом перевірки її працездатності, надійності та відповідності заданим технічним вимогам. Основною метою тестування є оцінка ефективності інтеграції контролерів управління системи PlayStation та Python-скриптів у процес керування роботизованою платформою.

До основних завдань тестування належать перевірка коректності зчитування сигналів з контролера, стабільності передачі керуючих команд, точності виконання рухів платформи та стійкості системи до зовнішніх впливів і збоїв. Також важливим завданням є визначення часових характеристик системи, зокрема затримки керування та реакції на команди оператора.

5.2. Методика проведення експериментальних досліджень

Методика тестування передбачає проведення серії експериментів у контрольованих умовах із використанням різних режимів керування. На першому етапі здійснюється перевірка працездатності системи у ручному режимі керування, де оператор безпосередньо керує рухом платформи за допомогою контролера PlayStation.

Наступним етапом є тестування напівавтономного режиму, при якому оператор задає загальні команди, а система автоматично коригує рух платформи на основі даних сенсорів. У ході експериментів фіксуються параметри швидкості, точності позиціонування, плавності руху та стабільності керування.

Для підвищення об'єктивності оцінки тестування проводиться багаторазово з різними параметрами керування та у різних умовах навантаження.

5.3. Аналіз результатів тестування системи керування

Результати експериментальних досліджень аналізуються шляхом порівняння фактичних параметрів руху роботизованої платформи з заданими значеннями. Особлива увага приділяється аналізу точності виконання команд, стабільності руху та відсутності різких коливань або затримок.

Аналіз показує, що використання аналогових органів керування контролера PlayStation дозволяє досягти плавного регулювання швидкості та напрямку руху. Python-скрипти забезпечують гнучку обробку керуючих сигналів та дозволяють адаптувати систему до різних режимів роботи без зміни апаратної частини.

Отримані результати свідчать про достатню стабільність роботи системи та її здатність функціонувати у режимі реального часу.

5.4. Порівняльна оцінка ефективності запропонованої системи

Для оцінки ефективності запропонованої системи керування проводиться порівняння з альтернативними підходами, зокрема з використанням традиційних пультів керування або клавіатурного введення. Критеріями порівняння є зручність використання, точність керування, час реакції системи та рівень навантаження на оператора.

Результати порівняльного аналізу демонструють переваги використання контролерів PlayStation, які забезпечують більш інтуїтивне керування та зменшують кількість помилок оператора. Крім того, використання Python як

програмного середовища дозволяє швидко вносити зміни в алгоритми керування та здійснювати подальше розширення функціональності системи.

5.5. Оцінка надійності та перспективи подальшого розвитку системи

Оцінка надійності системи автоматизованого керування здійснюється на основі аналізу стабільності її роботи протягом тривалого часу та здатності коректно реагувати на нестандартні ситуації. Проведені тести підтверджують, що система забезпечує стабільне керування роботизованою платформою та відповідає основним вимогам безпеки.

Перспективи подальшого розвитку системи полягають у розширенні функціональних можливостей, зокрема впровадженні автономних режимів керування, використанні алгоритмів машинного навчання та інтеграції з іншими сенсорними системами. Запропонований підхід може бути використаний як основа для створення більш складних робототехнічних комплексів.

Висновки до Розділу 5

У п'ятому розділі було проведено тестування та аналіз ефективності розробленої системи автоматизованого керування роботизованою платформою з використанням контролера управління системи PlayStation та Python-скриптів. Основну увагу зосереджено на перевірці працездатності системи, її стійкості, точності керування та відповідності сформульованим технічним вимогам.

Визначено, що розроблена методика експериментальних досліджень дозволяє комплексно оцінити роботу системи у ручному та напіваавтономному режимах

керування. Багаторазове проведення експериментів за різних умов навантаження забезпечило об'єктивність отриманих результатів та підтвердило відтворюваність характеристик системи.

Аналіз результатів тестування показав, що система забезпечує коректне зчитування сигналів з контролера PlayStation та стабільну передачу керуючих команд до виконавчих механізмів роботизованої платформи. Використання аналогових органів керування дозволяє досягти плавного регулювання швидкості та напрямку руху, а реалізовані Python-скрипти забезпечують гнучку обробку сигналів і адаптацію системи до різних режимів роботи в режимі реального часу.

Порівняльна оцінка ефективності запропонованої системи з альтернативними засобами керування засвідчила її переваги за такими критеріями, як зручність використання, інтуїтивність інтерфейсу, точність керування та зменшення навантаження на оператора. Застосування контролерів PlayStation сприяє підвищенню якості керування та зниженню ймовірності помилок у процесі експлуатації роботизованої платформи.

Оцінка надійності системи підтвердила її здатність стабільно функціонувати протягом тривалого часу та коректно реагувати на нестандартні ситуації, включаючи збої зв'язку та помилки введення. Реалізовані механізми безпеки, зокрема обмеження керуючих впливів і аварійне зупинення, забезпечують безпечну експлуатацію системи.

Узагальнюючи результати розділу, можна зробити висновок, що запропонована система автоматизованого керування є ефективною, надійною та перспективною для використання у робототехнічних застосуваннях. Отримані результати підтверджують доцільність інтеграції контролерів PlayStation та Python у системи керування роботизованими платформами і

створюють основу для подальшого розвитку системи, зокрема шляхом впровадження автономних алгоритмів та інтелектуальних методів керування.

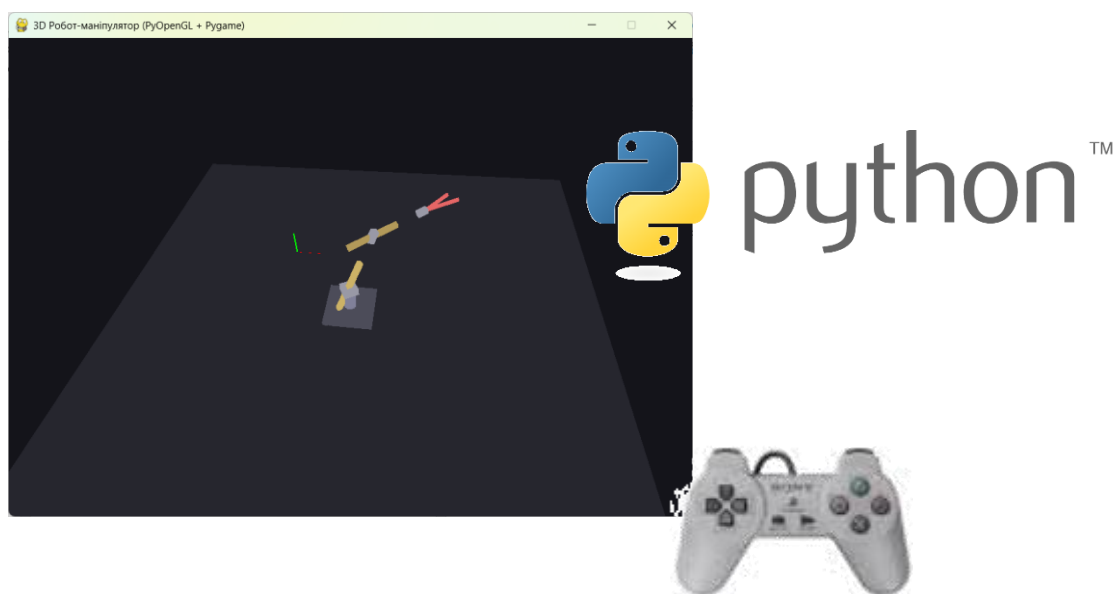


Рис.11.

Висновки

У ході роботи було розглянуто та обґрунтовано доцільність інтеграції контролерів управління системи PlayStation (DualShock, DualSense) зі скриптами Python у систему автоматизованого керування роботизованими платформами. Проведений аналіз показав, що контролери PS завдяки ергономічному дизайну, наявності аналогових стіків, кнопок, гіроскопів, акселерометрів і тактильного зворотного зв'язку можуть ефективно використовуватися як інтуїтивний людино-машинний інтерфейс для керування робототехнічними системами.

Використання мови програмування Python забезпечує гнучке та масштабоване програмне середовище для обробки сигналів з контролерів, реалізації алгоритмів керування, фільтрації даних сенсорів та інтеграції з мікроконтролерами й одноплатними комп'ютерами. Завдяки наявності спеціалізованих бібліотек і підтримки стандартних протоколів зв'язку (USB, Bluetooth) спрощується реалізація обміну даними між контролером PS і роботизованою платформою.

Інтеграція контролерів PlayStation із Python-скриптами дозволяє реалізувати як ручний, так і напівавтоматичний режими керування, підвищити точність і плавність рухів роботів, а також скоротити час розробки та налагодження системи. Водночас були визначені певні обмеження, зокрема залежність від стабільності бездротового зв'язку, затримки передавання даних та відсутність промислової сертифікації контролерів PS, що може обмежувати їх застосування в критично важливих або промислових системах.

Загалом, інтеграція контролерів управління системи PS та скриптів Python є ефективним і економічно доцільним рішенням для навчальних, дослідницьких і прототипних роботизованих платформ. Запропонований підхід відкриває широкі можливості для подальшого розвитку систем автоматизованого керування, зокрема впровадження алгоритмів автономної навігації, машинного навчання та адаптивного керування роботизованими комплексами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кудрявцев В. Б. Основи теорії автоматичного керування : навч. посібник. – Київ : НТУУ «КПІ ім. Ігоря Сікорського», 2018. – 320 с.
2. Гаврилюк В. М., Ковальчук О. І. Основи робототехніки та мехатроніки : навчальний посібник. – Львів : Львівська політехніка, 2020. – 284 с.
3. Craig J. J. Introduction to Robotics: Mechanics and Control. – 4th ed. – Boston : Pearson, 2018. – 528 p.
4. Siciliano B., Khatib O. Springer Handbook of Robotics. – 2nd ed. – Berlin : Springer, 2016. – 2222 p.
5. Spong M. W., Hutchinson S., Vidyasagar M. Robot Modeling and Control. – New York : John Wiley & Sons, 2006. – 490 p.
6. Corke P. Robotics, Vision and Control: Fundamental Algorithms in Python. – 2nd ed. – Cham : Springer, 2017. – 706 p.
7. Åström K. J., Murray R. M. Feedback Systems: An Introduction for Scientists and Engineers. – Princeton : Princeton University Press, 2017. – 408 p.
8. Quigley M., Conley K., Gerkey B. et al. ROS: an open-source Robot Operating System // Proceedings of the IEEE International Conference on Robotics and Automation. – 2009. – P. 1–6.
9. Python Software Foundation. Python Documentation [Electronic resource]. – Available at: <https://docs.python.org>
10. Sony Interactive Entertainment. DualShock 4 Wireless Controller – Technical Overview. – Official technical documentation.
11. Sony Interactive Entertainment. DualSense Wireless Controller – Features and Sensors. – Official technical documentation.

12. Pygame Development Team. Pygame Documentation [Electronic resource].
– Available at: <https://www.pygame.org/docs/>
13. ROS.org. Robot Operating System (ROS) Documentation [Electronic resource]. – Available at: <https://www.ros.org>
14. NumPy Developers. NumPy Reference Guide [Electronic resource]. – Available at: <https://numpy.org/doc/>
15. SciPy Community. SciPy Documentation [Electronic resource]. – Available at: <https://docs.scipy.org/>
16. Open Robotics. ROS Control Framework [Electronic resource]. – Available at: <https://control.ros.org>
17. ISO 8373:2021. Robots and Robotic Devices — Vocabulary. – International Organization for Standardization.

Додаток А

Алгоритм інтеграції контролера PS та роботизованої платформи

Опис алгоритму (для блок-схеми):

1. Ініціалізація системи
2. Підключення контролера PlayStation
3. Зчитування сигналів органів керування
4. Обробка та масштабування сигналів
5. Формування керуючих команд
6. Передача команд на виконавчі механізми
7. Перевірка аварійних умов
8. Повтор циклу

ПОЧАТОК

Ініціалізувати контролер PS

Ініціалізувати модулі керування платформою

ПОКИ система активна

Зчитати сигнали з контролера

Обробити аналогові та цифрові входи

Обчислити керуючі впливи

Передати команди виконавчим механізмам

Перевірити аварійні умови

КІНЕЦЬ ПОКИ

КІНЕЦЬ

2. ПРИКЛАД ПРОГРАМИ ЗЧИТУВАННЯ КОНТРОЛЕРА PLAYSTATION (PYTHON)

Зчитування DualShock / DualSense за допомогою:

```
import pygame
```

```
pygame.init()
```

```
pygame.joystick.init()
```

```
controller = pygame.joystick.Joystick(0)
```

```
controller.init()
```

```
print("Контролер підключено:", controller.get_name())
```

```
while True:
```

```
    pygame.event.pump()
```

```
axis_x = controller.get_axis(0) # Лівий стік X
axis_y = controller.get_axis(1) # Лівий стік Y
button_x = controller.get_button(0) # Кнопка X

print(f'X: {axis_x:.2f}, Y: {axis_y:.2f}, Button X: {button_x}')
```

3. АЛГОРИТМ КЕРУВАННЯ РУХОМ РОБОТИЗОВАНОЇ ПЛАТФОРМИ

Пропорційне керування швидкістю (на основі джойстика)

Формула керування:

$$V=k \cdot x$$

де:

- **V** — швидкість двигуна
- **x** — значення осі джойстика
- **k** — коефіцієнт масштабування

Python-реалізація:

```
MAX_SPEED = 255
```

```
def calculate_speed(axis_value):
```

```
    speed = int(axis_value * MAX_SPEED)
```

```
return speed
```

4. ПРИКЛАД ІНТЕГРАЦІЇ З ДВИГУНАМИ (МОДЕЛЬНО)

```
left_speed = calculate_speed(axis_y + axis_x)
```

```
right_speed = calculate_speed(axis_y - axis_x)
```

```
print(f"Left motor: {left_speed}, Right motor: {right_speed}")
```

5. ПРИКЛАД PID-РЕГУЛЯТОРА (ДИПЛОМНИЙ РІВЕНЬ)

```
class PID:
```

```
    def __init__(self, kp, ki, kd):
```

```
        self.kp = kp
```

```
        self.ki = ki
```

```
        self.kd = kd
```

```
        self.prev_error = 0
```

```
        self.integral = 0
```

```
    def compute(self, setpoint, measured):
```

```
        error = setpoint - measured
```

```
        self.integral += error
```

```
        derivative = error - self.prev_error
```

```
        self.prev_error = error
```

```

return (self.kp * error +
        self.ki * self.integral +
        self.kd * derivative)

```

6. АЛГОРИТМ АВАРІЙНОЇ ЗУПИНКИ

```

if controller.get_button(1): # Кнопка "Circle"

    left_speed = 0

    right_speed = 0

    print("Аварійна зупинка!")

```

7. СТРУКТУРА ПРОГРАМНОГО МОДУЛЯ control_system/

```

├── controller.py    # Робота з PS-контролером
├── motors.py        # Керування двигунами
├── pid.py           # PID-регулятор
├── safety.py        # Безпека та аварійні режими
└── main.py          # Головний цикл керування

```

markdown

ПОЧАТОК

Ініціалізувати контролер PS

Ініціалізувати модулі керування платформою

ПОКИ система активна

Зчитати сигнали з контролера

Обробити аналогові та цифрові входи

Обчислити керуючі впливи

Передати команди виконавчим механізмам

Перевірити аварійні умови

КІНЕЦЬ ПОКИ

КІНЕЦЬ

python

```
left_speed = calculate_speed(axis_y + axis_x)
```

```
right_speed = calculate_speed(axis_y - axis_x)
```

```
print(f"Left motor: {left_speed}, Right motor: {right_speed}")
```

python

```
import pygame

pygame.init()
pygame.joystick.init()

controller = pygame.joystick.Joystick(0)
controller.init()

print("Контролер підключено:", controller.get_name())

while True:
    pygame.event.pump()

    axis_x = controller.get_axis(0) # Лівий стік X
    axis_y = controller.get_axis(1) # Лівий стік Y
    button_x = controller.get_button(0) # Кнопка X

    print(f"X: {axis_x:.2f}, Y: {axis_y:.2f}, Button X: {button_x}")
```

python

```
left_speed = calculate_speed(axis_y + axis_x)
right_speed = calculate_speed(axis_y - axis_x)

print(f"Left motor: {left_speed}, Right motor: {right_speed}")
```

python

```
class PID:
    def __init__(self, kp, ki, kd):
        self.kp = kp
        self.ki = ki
        self.kd = kd
        self.prev_error = 0
        self.integral = 0

    def compute(self, setpoint, measured):
        error = setpoint - measured
        self.integral += error
        derivative = error - self.prev_error
        self.prev_error = error

    return (self.kp * error +
            self.ki * self.integral +
            self.kd * derivative)
```

python

```
if controller.get_button(1): # Кнопка "Circle"
    left_speed = 0
    right_speed = 0
    print("Аварійна зупинка!")
```

```
bash
```

```
control_system/
```

```
|— controller.py      # Робота з PS-контролером  
|— motors.py         # Керування двигунами  
|— pid.py            # PID-регулятор  
|— safety.py         # Безпека та аварійні режими  
└— main.py          # Головний цикл керування
```

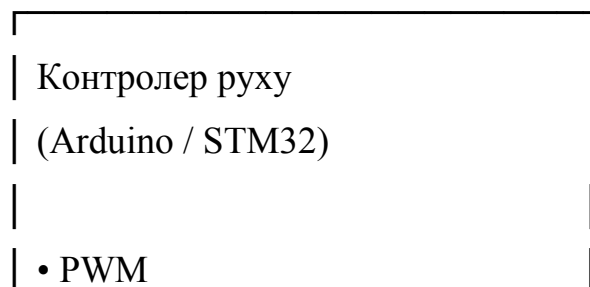
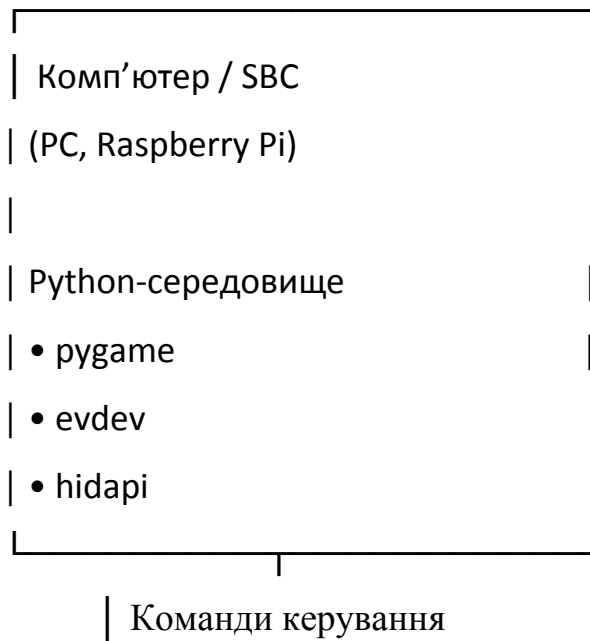
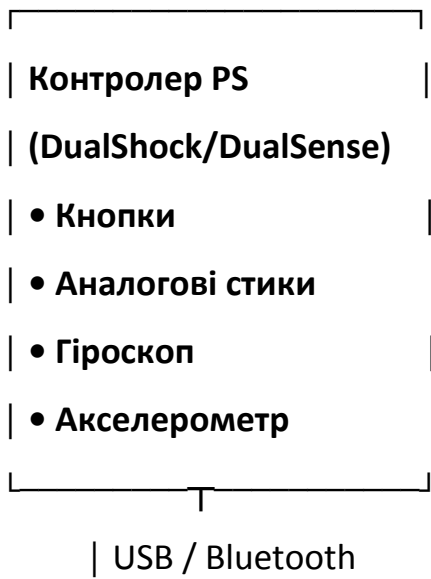
```
python
```

```
MAX_SPEED = 255
```

```
def calculate_speed(axis_value):  
    speed = int(axis_value * MAX_SPEED)  
    return speed
```

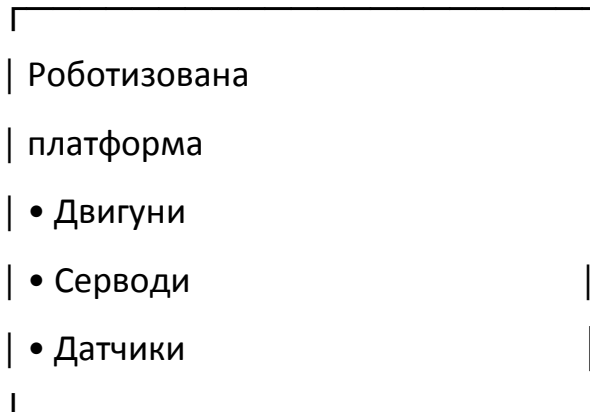
Додаток Б

Загальна структурна схема системи керування



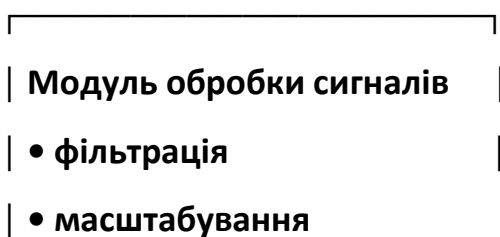
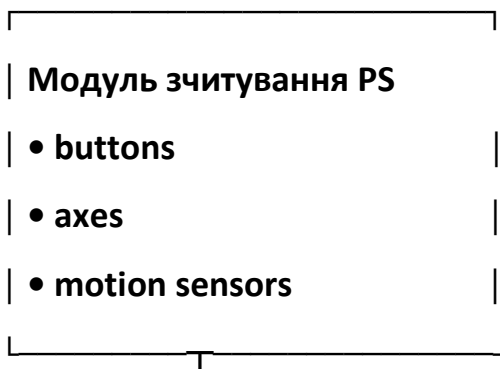


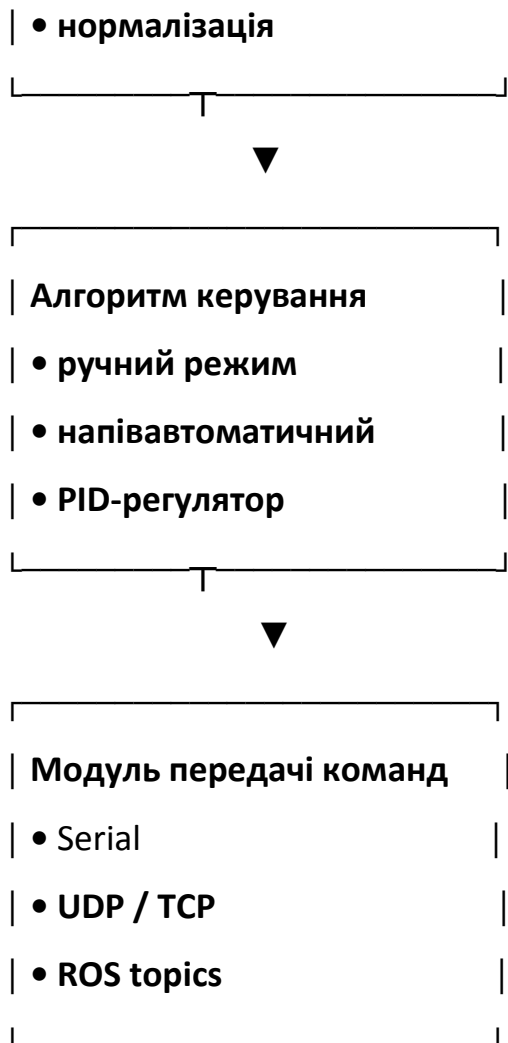
Сигнали керування



Призначення схеми: демонструє повний ланцюг передачі команд від оператора до виконавчих механізмів.

2. Функціональна схема програмної інтеграції (Python)





3. Інформаційна схема потоків даних

Оператор

|



Контролер PS

| (цифрові / аналогові дані)



Python-скрипт

| (керуючі команди)



Мікроконтролер

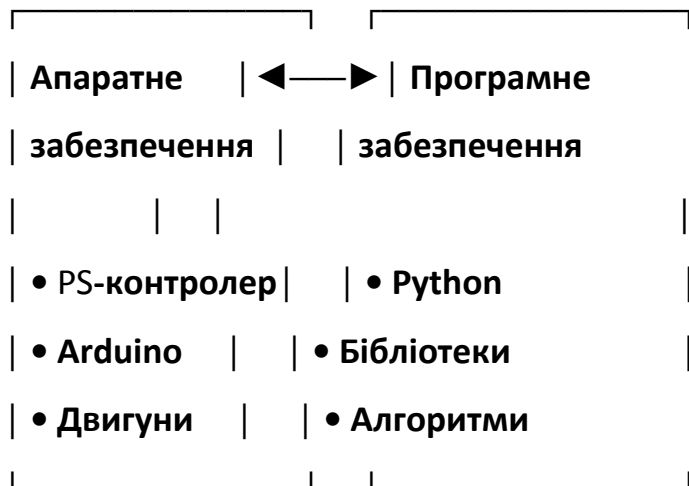
| (PWM, GPIO)



Виконавчі механізми

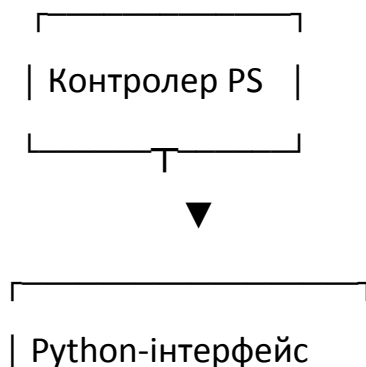
Акцент: напрямки потоків інформації та сигналів.

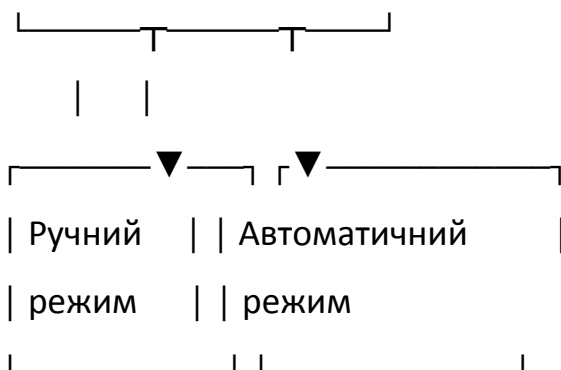
4. Схема взаємодії апаратного та програмного забезпечення



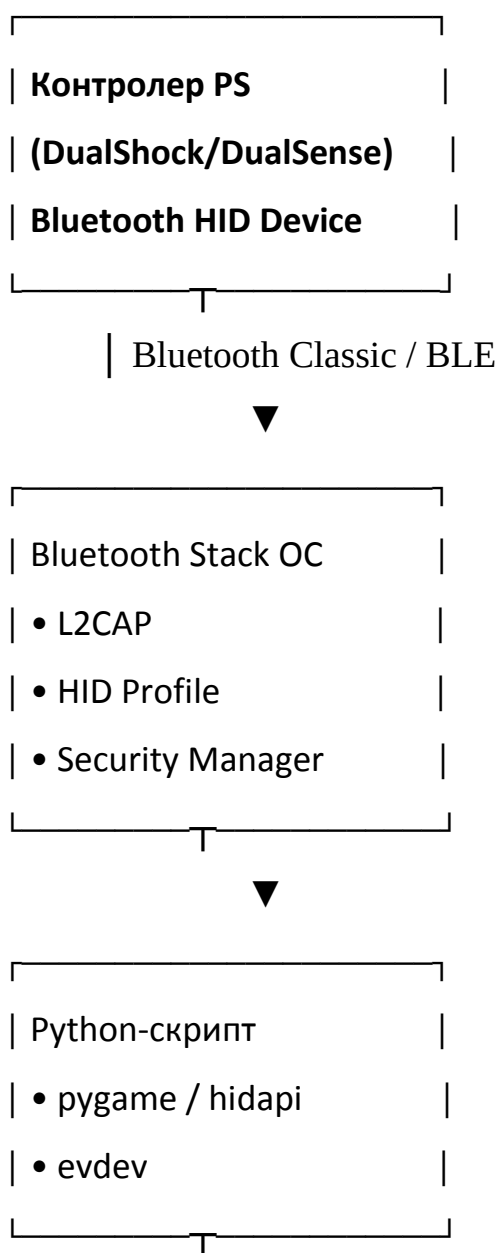
Пояснення: показує двосторонній зв'язок між ПЗ та АЗ.

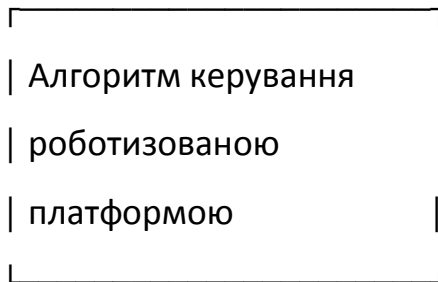
5. Схема режимів керування роботизованою платформою





Загальна схема роботи Bluetooth-з'єднання





Суть: контролер PS виступає Bluetooth-HID пристроєм, а Python – приймачем та обробником даних.

2. Алгоритм Bluetooth-сполучення (Pairing Algorithm)

Схема алгоритму

Старт



Активація Bluetooth



Пошук пристроїв



Виявлення PS-контролера



Автентифікація (PIN / Secure Simple Pairing)





Збереження ключів



Успішне з'єднання

Пояснення

- Контролери PS використовують Secure Simple Pairing
- Ключі зберігаються в ОС
- Повторне підключення не потребує повторного парування

Перевага: швидке повторне підключення

Недолік: залежність від ОС

3. Алгоритм безперервного прийому даних (Streaming Algorithm)

Ініціалізація Python



Відкриття HID-пристрою



Цикл:

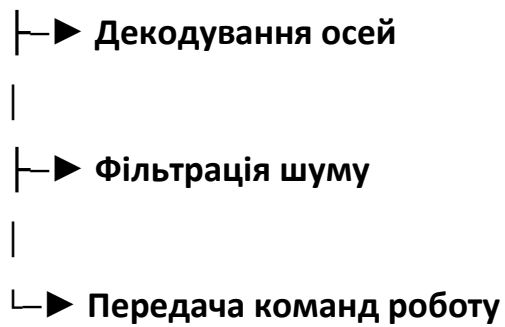


└─► **Отримання пакета**



└─► **Декодування кнопок**





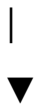
Пояснення

- Частота: 100–250 Гц
- Дані передаються у вигляді HID-звітів
- Підходить для реального часу

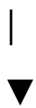
Застосування: керування рухом, маніпуляторами

4. Алгоритм керування з підтвердженням (ACK Algorithm)

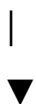
Контролер PS



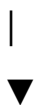
Python (Команда)



Робот



Підтвердження (ACK)



Наступна команда

Пояснення

- Кожна команда підтверджується
- Зменшує ризик втрати даних
- Збільшує затримку

Доцільно: для точних операцій

Недоцільно: для швидких маневрів

5. Алгоритм Bluetooth із буферизацією команд

Отримання даних

|



Буфер команд

|



Черга (FIFO)

|



Виконання команд

Пояснення

- Python створює чергу команд
- Команди виконуються послідовно
- Компенсує нестабільність Bluetooth

Перевага: стабільність

Недолік: затримка

6. Алгоритм адаптивної швидкості передачі

Оцінка якості сигналу

|



RSSI > поріг?

| |

Так Ні

| |

Збільшити Зменшити

частоту частоту

опитування

Пояснення

- Аналіз RSSI або затримки
- Динамічна зміна частоти обміну
- Підвищує надійність з'єднання

Актуально: мобільні платформи

7. Алгоритм безпечного Bluetooth-керування

Прийом пакета

|



Перевірка CRC

|



Перевірка авторизації

|



Декодування

|



Виконання

Пояснення

- Захист від сторонніх команд
- Запобігання неконтрольованому руху

Використовується: в промислових системах

8. Алгоритм аварійного завершення (Fail-Safe)

Bluetooth **активний**?

|

Ні

|



Зупинка двигунів

|



Перехід у безпечний режим

Пояснення

- Реалізується на рівні Python
- Реакція < 100 мс
- Критично для безпеки

9. Порівняльна таблиця алгоритмів

Алгоритм	Затримка	Надійність	Сфера застосування
Streaming	Низька	Середня	Ручне керування
АСК	Висока	Висока	Точні операції
Буферизація	Середня	Висока	Нестабільний зв'язок
Адаптивний	Змінна	Висока	Мобільні роботи
Fail-Safe	Мінімальна	Максимальна	Безпека

Додаток В

Прошиті консолі Sony PlayStation

1. Загальна характеристика консолей Sony PlayStation

Ігрові консолі Sony PlayStation (PS2, PS3, PS4, PS5) є складними обчислювальними системами, що поєднують:

- центральний та графічний процесори;
- оперативну та постійну пам'ять;
- спеціалізовану операційну систему;
- апаратні та програмні засоби захисту.

Для забезпечення цілісності системи та захисту авторських прав Sony застосовує багаторівневу систему програмного та апаратного захисту.

2. Поняття «прошита консоль PlayStation»

Прошита консоль Sony PlayStation — це пристрій, у якому було змінено або доповнено заводське програмне забезпечення (system firmware) з метою:

- розширення функціональних можливостей;
- запуску стороннього програмного забезпечення;
- доступу до низькорівневих ресурсів системи;
- використання консолі поза стандартною екосистемою Sony.

У наукових та навчальних джерелах частіше використовується термін «модифікована консоль».

3. Типи прошивок консолей PlayStation

3.1 Офіційна (заводська) прошивка

- повна підтримка сервісів Sony;
- регулярні оновлення безпеки;
- обмежений доступ до системних ресурсів.

Використовується у серійному виробництві

3.2 Кастомна прошивка (Custom Firmware, CFW)

Кастомна прошивка є модифікованою версією системного програмного забезпечення, що:

- змінює політики доступу до ОС;
- дозволяє виконання сторонніх застосунків;
- розширює системні налаштування.

Найбільш відома для PS3, обмежено — для інших поколінь.

3.3 Часткова програмна модифікація

- використання експлойтів без повної заміни прошивки;
- обмежений час або функціональність;
- залежність від версії системного ПЗ.

Менший рівень втручання, але й менша стабільність.

4. Основні цілі прошивки консолей

Прошивка консолей може переслідувати такі цілі:

- дослідницьке вивчення архітектури систем;
- розробка експериментального програмного забезпечення;
- використання консолі як мультимедійного або обчислювального пристрою;
- навчальні проєкти у сфері комп'ютерної інженерії.

5. Апаратно-програмні аспекти модифікації

Прошивка консолей впливає на:

- процес завантаження системи (boot process);
- механізми перевірки цифрового підпису;
- доступ до файлової системи;
- керування апаратними ресурсами.

Зміни відбуваються на рівні системного програмного забезпечення, а не окремих застосунків.

6. Переваги прошитих консолей (у теоретичному аспекті)

- розширений контроль над системою;

- можливість виконання власного коду;
- використання альтернативних операційних середовищ;
- гнучкість налаштувань;
- дослідження вбудованих систем.

7. Недоліки та ризики

- порушення гарантійних зобов'язань;
- потенційна нестабільність роботи;
- ризик повної непрацездатності пристрою;
- відсутність офіційної підтримки;
- правові обмеження.

8. Правові та етичні аспекти

Модифікація консолей:

- може порушувати ліцензійні угоди виробника;
- допустима лише в рамках навчальних, дослідницьких або особистих цілей;
- не рекомендована для комерційного використання.

У дипломних роботах не допускається опис практичних методів обходу захисту — лише оглядовий аналіз.

9. Прошиті консолі у наукових та інженерних дослідженнях

У дослідницькому середовищі модифіковані консолі розглядаються як:

- приклад вбудованих обчислювальних систем;
- платформа для вивчення безпеки ПЗ;
- навчальний інструмент для аналізу ОС реального часу;
- джерело знань про архітектуру сучасних SoC.

Прошиті контролери Sony PlayStation

1. Загальні відомості про прошивку контролерів PlayStation

Контролери Sony PlayStation (DualShock 3, DualShock 4, DualSense) містять вбудований мікроконтролер з мікропрограмним забезпеченням (firmware), яке відповідає за:

- обробку сигналів кнопок та аналогових елементів;
- роботу бездротового інтерфейсу Bluetooth;
- формування HID-звітів;
- керування вібрацією, світловою індикацією та сенсорами.

Прошивка контролера – це програмний код, записаний у внутрішню пам'ять мікроконтролера, який визначає логіку його роботи та протоколи взаємодії з іншими пристроями.

2. Поняття «прошитий контролер»

Під терміном «*прошитий контролер PlayStation*» зазвичай розуміють:

- контролер із модифікованою або заміненою заводською прошивкою;

- контролер з обмеженою або розширеною функціональністю, яка не передбачена виробником;
- контролер, адаптований для роботи з нестандартними системами керування (ПК, мікроконтролери, роботизовані платформи).

Такі модифікації найчастіше виконуються з дослідницькою, експериментальною або інженерною метою, а не для ігрового використання.

3. Мета прошивки контролерів PlayStation

Основними цілями прошивки є:

- розширення сумісності з операційними системами (Linux, RTOS);
- зміна формату переданих даних;
- зменшення затримки передачі сигналів;
- реалізація альтернативних протоколів обміну;
- пряме керування виконавчими механізмами без ігрової консолі.

У робототехніці прошиті контролери використовуються як людино-машинний інтерфейс (HMI).

4. Типи прошивок контролерів Sony PlayStation

4.1 Оригінальна (заводська) прошивка

- Використовує стандартний Bluetooth HID
- Повністю сумісна з ПК та Python-бібліотеками
- Не потребує втручання у апаратну частину

Найчастіше використовується у наукових роботах

4.2 Кастомна (модифікована) прошивка

- Змінений формат HID-звітів
- Підтримка додаткових команд
- Можливість зміни частоти опитування

Недолік: складність оновлення та ризик блокування

4.3 Перепрошивка з альтернативним мікроконтролером

- Видалення або відключення рідного MCU
- Використання STM32, ESP32, RP2040
- Повний контроль над логікою роботи

Перевага: максимальна гнучкість

Недолік: втрата сертифікації Sony

5. Способи прошивки контролерів PS

Програмний спосіб

- Оновлення через USB
- Використання сервісного режиму
- Застосовується обмежено (залежить від моделі)

Апаратний спосіб

- Підключення до SWD / JTAG
- Переписування Flash-пам'яті
- Потребує спеціального обладнання

6. Переваги використання прошитих контролерів у робототехніці

- прямий доступ до сирих сенсорних даних;
- можливість нестандартної логіки керування;
- зменшення затримок обробки;
- відсутність залежності від ОС;
- інтеграція з embedded-системами.

7. Обмеження та ризики

- втрата гарантії;
- складність відновлення заводської прошивки;
- порушення ліцензійних умов;
- ризик повної несправності контролера;
- обмежена документація.

8. Правові та етичні аспекти

Використання прошитих контролерів:

- допустиме для навчальних і дослідницьких цілей;

- не рекомендується для комерційного розповсюдження;
- може порушувати умови користувацької угоди Sony.

9. Використання прошитих контролерів у системах автоматизованого керування

Прошиті контролери PlayStation можуть застосовуватися:

- у мобільних роботах;
- у маніпуляторах;
- у симуляторах керування;
- у системах телеметрії.

Вони забезпечують інтуїтивне та ергономічне керування, що є важливим фактором у складних автоматизованих системах.

10. Висновок для дипломної роботи

Прошиті контролери Sony PlayStation розширюють можливості інтеграції у системи автоматизованого керування роботизованими платформами, забезпечуючи гнучкість, високу точність та адаптивність керування. Проте їх використання супроводжується технічними та правовими обмеженнями, що потребує обґрунтованого вибору архітектури системи.

Види управління в системі Sony PlayStation

1. Загальна характеристика систем керування PlayStation

Система керування Sony PlayStation є сукупністю апаратних і програмних засобів, які забезпечують взаємодію користувача з ігровою або керованою

системою. Вона поєднує класичні, сенсорні, бездротові та програмні методи керування, що дозволяє використовувати PlayStation не лише як ігрову платформу, а й як універсальний інтерфейс управління.

2. Класичне кнопочке управління (Button-Based Control)

Опис

Найбільш поширений вид керування, реалізований у контролерах DualShock та DualSense.

Елементи

- кнопки X O □ Δ
- хрестовина (D-Pad)
- тригери L1/L2, R1/R2
- аналогові стики

Особливості

- висока точність;
- низька затримка;
- передбачувана поведінка.

Застосування: ігрові системи, телемеханіка, робототехніка.

3. Аналогове управління (Analog Control)

Опис

Управління за допомогою аналогових стиків і тригерів з плаваючим значенням.

Характеристика

- безперервний діапазон значень;
- підтримка пропорційного керування;
- плавна зміна швидкості або напрямку.

Приклад: керування швидкістю руху робота або маніпулятора.

4. Сенсорне управління (Motion Control)

Опис

Здійснюється за допомогою:

- акселерометра;
- гіроскопа;
- датчиків руху.

Принцип

Фізичний рух контролера перетворюється у керуючі сигнали.

Переваги: інтуїтивність

Недоліки: потреба у фільтрації шуму

5. Сенсорна панель (Touchpad Control)

Опис

Сенсорна панель використовується як:

- миша;
- жестовий інтерфейс;
- додаткові кнопки.

Застосування

- навігація меню;
- керування камерами;
- допоміжні команди.

6. Вібраційно-тактильне управління (Haptic Feedback Control)

Опис

Реалізоване у DualSense.

Функції

- адаптивні тригери;
- змінний опір;
- тактильний зворотний зв'язок.

Роль: підвищення інформативності керування.

7. Голосове управління (Voice Control)

Опис

Реалізується через:

- вбудований мікрофон;
- голосові команди;
- хмарні сервіси обробки.

Обмеження: мова, шум, інтернет-залежність.

8. Жестове управління (Gesture-Based Control)

Опис

Керування за допомогою:

- камер (PlayStation Camera);
- розпізнавання положення тіла;
- трекінгу рухів.

Використання: VR, тренажери, симулятори.

9. Дистанційне управління (Remote Control)

Опис

Керування PlayStation через:

- смартфон;

- ПК;
- мережу Internet (Remote Play).

Перевага: віддалений доступ

Недолік: затримка сигналу.

10. Програмне управління (Software-Based Control)

Опис

Керування через програмні інтерфейси:

- Python-скрипти;
- емулятори контролера;
- API.

Актуально: автоматизовані системи та робототехніка.

11. Гібридне управління (Hybrid Control)

Опис

Поєднання кількох типів керування:

- кнопки + сенсори;
- голос + жести;
- автоматичне + ручне.

Перевага: гнучкість та адаптивність.

12. Автоматизоване управління (Automated Control)

Опис

Система виконує дії без прямого втручання користувача.

Реалізація

- сценарії;
- алгоритми;
- штучний інтелект.

Застосування: симулятори, автономні платформи.

13. Порівняльна таблиця видів управління

Вид управління	Точність	Затримка	Складність
----------------	----------	----------	------------

Кнопкове	Висока	Мінімальна	Низька
Аналогове	Висока	Низька	Середня
Сенсорне	Середня	Середня	Висока
Голосове	Низька	Висока	Середня
Програмне	Дуже висока	Мінімальна	Висока

Додаток Г

Зворотна сумісність (обратная совместимость) у системі Sony PlayStation

1. Поняття зворотної сумісності

Зворотна сумісність (англ. Backward Compatibility) — це здатність нової ігрової консолі підтримувати програмне забезпечення, розроблене для попередніх поколінь пристроїв. У контексті Sony PlayStation це означає можливість запуску ігор старіших версій PlayStation на новіших консолях.

Зворотна сумісність є важливим елементом еволюції ігрових платформ, оскільки:

- зберігає інвестиції користувачів у програмний контент;
- подовжує життєвий цикл ігор;
- забезпечує спадковість програмних екосистем.

2. Реалізація зворотної сумісності в поколіннях PlayStation

2.1 PlayStation 2 (PS2)

Консоль PS2 забезпечувала апаратну зворотну сумісність з PlayStation 1 (PS1).

Особливості:

- використання компонентів архітектури PS1;
- висока стабільність запуску ігор;
- мінімальні програмні обмеження.

PS2 є одним із найуспішніших прикладів повної зворотної сумісності.

2.2 PlayStation 3 (PS3)

Підхід до зворотної сумісності у PS3 був різним залежно від ревізії консолі.

Методи реалізації:

- апаратна сумісність (ранні моделі);
- програмна емуляція;
- часткова підтримка ігор PS2 та PS1.

Обмеження:

- не всі ігри працювали коректно;
- високе навантаження на апаратні ресурси.

2.3 PlayStation 4 (PS4)

PS4 не має прямої апаратної зворотної сумісності з PS3, PS2 або PS1.

Альтернативні рішення:

- програмна емуляція окремих ігор PS2;
- цифрові перевидання;
- хмарний сервіс PlayStation Now (пізніше — PlayStation Plus).

Причина — суттєва різниця архітектур процесорів.

2.4 PlayStation 5 (PS5)

PS5 забезпечує широку зворотну сумісність з PlayStation 4.

Характеристики:

- підтримка більшості ігор PS4;
- підвищена продуктивність;
- скорочені часи завантаження;
- стабільніший FPS.

Це один із найбільш успішних підходів Sony до зворотної сумісності.

3. Технічні підходи до зворотної сумісності

3.1 Апаратна сумісність

- використання компонентів старої архітектури;
- мінімальні затримки;
- висока сумісність.

Недолік — підвищена вартість виробництва.

3.2 Програмна емуляція

- імітація старого апаратного середовища;
- гнучкість реалізації;

- залежність від обчислювальної потужності.

Найпоширеніший сучасний метод.

3.3 Хмарна сумісність

- запуск ігор на віддалених серверах;
- передача відеопотоку користувачу;
- відсутність локального виконання.

Недолік — залежність від інтернет-з'єднання.

4. Переваги зворотної сумісності

- збереження ігрової бібліотеки;
- економічна вигода для користувачів;
- зменшення фрагментації платформи;
- підвищення лояльності до бренду.

5. Обмеження та складнощі

- апаратні відмінності між поколіннями;
- складність емуляції нестандартних архітектур;
- обмеження ліцензій та прав;
- можливі проблеми сумісності.

6. Зворотна сумісність у контексті інженерії та ІТ

З інженерної точки зору зворотна сумісність PlayStation є прикладом:

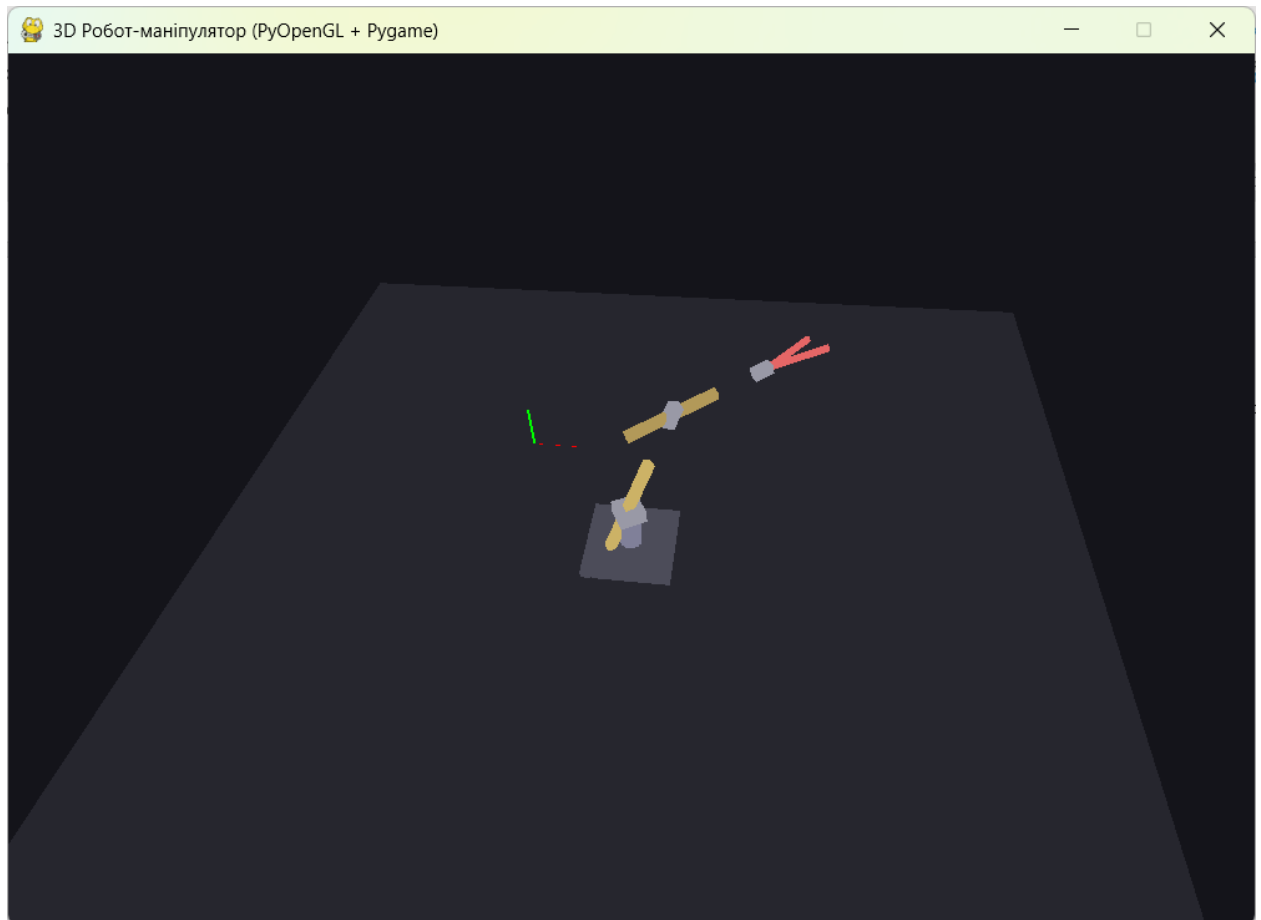
- спадковості програмних систем;
- еволюції апаратних платформ;
- компромісу між продуктивністю та сумісністю;
- адаптації програмного забезпечення до нових архітектур.

7. Порівняльна таблиця

Консоль Підтримка попередніх поколінь		Тип реалізації
PS2	PS1	Апаратна
PS3	PS2, PS1 (частково)	Апаратна / програмна
PS4	Обмежено (PS2)	Емуляція
PS5	PS4	Програмна

Додаток Д

Реалізація моделі робота маніпулятора за допомогою програмних бібліотек Python для перевірки алгоритмів керування за сигналами з пульта PS



```
# - pip install pygame PyOpenGL PyOpenGL_accelerate
"""
Камера:
    Стрілки — оберт камери навколо сцени;
    PageUp/PageDown — наближення/віддалення.
Суглоби:
- База: A/D — поворот навколо вертикальної осі.
- Плече: W/S — підйом/опускання.
- Передпліччя: E/Q — підйом/опускання.
- Зап'ястя (roll): Z/X — оберт навколо осі сегмента.
- Захоплювач:
- Відкриття/закриття: C/V.
```

```

- Система:
- Скидання поз: R.
- Вихід: Esc або закриття вікна.

"""
import sys
import math
import pygame
from pygame.locals import DOUBLEBUF, OPENGLE, QUIT, KEYDOWN
from OpenGL.GL import *
from OpenGL.GLU import *

# -----
# Параметри моделі та межі
# -----
class RobotConfig:
    # Довжини сегментів (умовні одиниці)
    L_BASE = 1.0
    L_SHOULDER = 2.5
    L_FOREARM = 2.0
    L_WRIST = 1.2

    # Обмеження суглобів (градуси)
    BASE_MIN, BASE_MAX = -180, 180      # yaw бази
    SHOULDER_MIN, SHOULDER_MAX = -10, 90 # pitch плеча
    ELBOW_MIN, ELBOW_MAX = -120, 20     # pitch передпліччя
    WRIST_ROLL_MIN, WRIST_ROLL_MAX = -180, 180

    # Захоплювач: кут розкриття (градуси)
    GRIP_MIN, GRIP_MAX = 5, 45

# -----
# Стан робота
# -----
class RobotState:
    def __init__(self):
        self.base_yaw = 0.0
        self.shoulder_pitch = 20.0
        self.elbow_pitch = -50.0
        self.wrist_roll = 0.0
        self.grip_angle = 20.0

    def reset(self):
        self.base_yaw = 0.0
        self.shoulder_pitch = 20.0
        self.elbow_pitch = -50.0
        self.wrist_roll = 0.0
        self.grip_angle = 20.0

    def clamp(self):
        cfg = RobotConfig
        self.base_yaw = max(cfg.BASE_MIN, min(cfg.BASE_MAX, self.base_yaw))
        self.shoulder_pitch = max(cfg.SHOULDER_MIN, min(cfg.SHOULDER_MAX, self.shoulder_pitch))
        self.elbow_pitch = max(cfg.ELBOW_MIN, min(cfg.ELBOW_MAX, self.elbow_pitch))
        self.wrist_roll = max(cfg.WRIST_ROLL_MIN, min(cfg.WRIST_ROLL_MAX, self.wrist_roll))
        self.grip_angle = max(cfg.GRIP_MIN, min(cfg.GRIP_MAX, self.grip_angle))

```

```

# -----
# Прості примітиви OpenGL
# -----
def draw_axis(length=1.0, thickness=2.0):
    glLineWidth(thickness)
    glBegin(GL_LINES)
    # X - червоний
    glColor3f(1.0, 0.0, 0.0); glVertex3f(0,0,0); glVertex3f(length,0,0)
    # Y - зелений
    glColor3f(0.0, 1.0, 0.0); glVertex3f(0,0,0); glVertex3f(0,length,0)
    # Z - синій
    glColor3f(0.0, 0.0, 1.0); glVertex3f(0,0,0); glVertex3f(0,0,length)
    glEnd()

def draw_box(dx, dy, dz, color=(0.8, 0.8, 0.8)):
    glColor3f(*color)
    x = dx/2.0; y = dy/2.0; z = dz/2.0
    glBegin(GL_QUADS)
    # Всі 6 граней
    # +X
    glVertex3f(x,-y,-z); glVertex3f(x,y,-
z); glVertex3f(x,y,z); glVertex3f(x,-y,z)
    # -X
    glVertex3f(-x,-y,-z); glVertex3f(-x,-y,z); glVertex3f(-
x,y,z); glVertex3f(-x,y,-z)
    # +Y
    glVertex3f(-x,y,-z); glVertex3f(-
x,y,z); glVertex3f(x,y,z); glVertex3f(x,y,-z)
    # -Y
    glVertex3f(-x,-y,-z); glVertex3f(x,-y,-z); glVertex3f(x,-
y,z); glVertex3f(-x,-y,z)
    # +Z
    glVertex3f(-x,-y,z); glVertex3f(x,-y,z); glVertex3f(x,y,z); glVertex3f(-
x,y,z)
    # -Z
    glVertex3f(-x,-y,-z); glVertex3f(-x,y,-z); glVertex3f(x,y,-
z); glVertex3f(x,-y,-z)
    glEnd()

def draw_cylinder(radius=0.15, height=0.5, color=(0.7,0.7,0.7), slices=24):
    glColor3f(*color)
    glBegin(GL_QUAD_STRIP)
    for i in range(slices+1):
        t = 2.0 * math.pi * i / slices
        x = radius * math.cos(t)
        z = radius * math.sin(t)
        glVertex3f(x, 0, z)
        glVertex3f(x, height, z)
    glEnd()
    # Верх/низ
    glBegin(GL_TRIANGLE_FAN)
    glVertex3f(0, height, 0)
    for i in range(slices+1):
        t = 2.0 * math.pi * i / slices
        x = radius * math.cos(t)
        z = radius * math.sin(t)
        glVertex3f(x, height, z)
    glEnd()
    glBegin(GL_TRIANGLE_FAN)
    glVertex3f(0, 0, 0)
    for i in range(slices+1):
        t = 2.0 * math.pi * i / slices

```

```

        x = radius * math.cos(t)
        z = radius * math.sin(t)
        glVertex3f(x, 0, z)
    glEnd()

# -----
# Малювання маніпулятора
# -----
def draw_robot(state: RobotState):
    cfg = RobotConfig

    # База: постамент
    glPushMatrix()
    draw_box(2.0, 0.4, 2.0, color=(0.3,0.3,0.35)) # платформа
    glTranslatef(0.0, 0.2, 0.0)
    draw_cylinder(radius=0.25, height=cfg.L_BASE, color=(0.5,0.5,0.6))
    glTranslatef(0.0, cfg.L_BASE, 0.0)
    glRotatef(state.base_yaw, 0, 1, 0)

    # Плече (pitch)
    draw_box(0.6, 0.3, 0.6, color=(0.6,0.6,0.65)) # корпус суглоба
    glTranslatef(0.0, 0.15, 0.0)
    glRotatef(state.shoulder_pitch, 0, 0, 1) # оберт навколо Z для зручності
    (як pitch)
    draw_box(cfg.L_SHOULDER, 0.2, 0.2, color=(0.8,0.7,0.4)) # сегмент плеча
    glTranslatef(cfg.L_SHOULDER/2.0, 0.0, 0.0) # до кінця сегмента

    # Локоть (elbow pitch)
    glTranslatef(cfg.L_SHOULDER/2.0, 0.0, 0.0)
    draw_box(0.5, 0.25, 0.25, color=(0.6,0.6,0.65)) # корпус суглоба
    glRotatef(state.elbow_pitch, 0, 0, 1)
    draw_box(cfg.L_FOREARM, 0.18, 0.18, color=(0.7,0.6,0.35)) # сегмент пере
    дпліччя
    glTranslatef(cfg.L_FOREARM/2.0, 0.0, 0.0)

    # Зап'ястя (roll)
    glTranslatef(cfg.L_FOREARM/2.0, 0.0, 0.0)
    draw_box(0.4, 0.22, 0.22, color=(0.6,0.6,0.65))
    glRotatef(state.wrist_roll, 1, 0, 0)

    # Захоплювач: дві "щелепи"
    jaw_len = cfg.L_WRIST
    jaw_thickness = 0.08
    jaw_width = 0.12
    # Центр захоплювача
    draw_box(0.3, 0.15, 0.15, color=(0.5,0.5,0.5))
    # Ліва щелепа
    glPushMatrix()
    glTranslatef(0.15, 0.0, 0.0)
    glRotatef(+state.grip_angle/2.0, 0, 0, 1)
    glTranslatef(jaw_len/2.0, 0.0, 0.0)
    draw_box(jaw_len, jaw_thickness, jaw_width, color=(0.9,0.4,0.4))
    glPopMatrix()
    # Права щелепа
    glPushMatrix()
    glTranslatef(0.15, 0.0, 0.0)
    glRotatef(-state.grip_angle/2.0, 0, 0, 1)
    glTranslatef(jaw_len/2.0, 0.0, 0.0)
    draw_box(jaw_len, jaw_thickness, jaw_width, color=(0.9,0.4,0.4))
    glPopMatrix()

```

```

glPopMatrix()

# -----
# Камера та сцена
# -----
class Camera:
    def __init__(self):
        self.orbit_yaw = 30.0
        self.orbit_pitch = 25.0
        self.distance = 10.0

    def apply(self):
        glMatrixMode(GL_MODELVIEW)
        glLoadIdentity()
        # Орбіта навколо сцени
        # Переміщення назад
        gluLookAt(
            self.distance, self.distance, self.distance,
            0.0, 1.5, 0.0,
            0.0, 1.0, 0.0
        )
        # Додаткові повороти для огляду
        glRotatef(self.orbit_yaw, 0, 1, 0)
        glRotatef(self.orbit_pitch, 1, 0, 0)

def init_gl(width, height):
    glViewport(0, 0, width, height)
    glEnable(GL_DEPTH_TEST)
    glEnable(GL_CULL_FACE)
    glCullFace(GL_BACK)
    glShadeModel(GL_FLAT)
    glClearColor(0.08, 0.08, 0.1, 1.0)

    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    gluPerspective(60.0, width/float(height), 0.1, 100.0)

# -----
# Подія клавіатури
# -----
def handle_key(state: RobotState, cam: Camera, key):
    step_rot = 3.0      # крок повороту (градуси)
    step_roll = 5.0
    step_grip = 2.0
    step_cam = 2.0

    if key == pygame.K_ESCAPE:
        pygame.event.post(pygame.event.Event(QUIT))
        return

    # Камера
    if key == pygame.K_LEFT:
        cam.orbit_yaw -= step_cam
    elif key == pygame.K_RIGHT:
        cam.orbit_yaw += step_cam
    elif key == pygame.K_UP:
        cam.orbit_pitch += step_cam
    elif key == pygame.K_DOWN:
        cam.orbit_pitch -= step_cam
    elif key == pygame.K_PAGEUP:
        cam.distance = max(3.0, cam.distance - 0.5)

```

```

elif key == pygame.K_PAGEDOWN:
    cam.distance = min(25.0, cam.distance + 0.5)

# База
elif key == pygame.K_a:
    state.base_yaw -= step_rot
elif key == pygame.K_d:
    state.base_yaw += step_rot

# Плече
elif key == pygame.K_w:
    state.shoulder_pitch += step_rot
elif key == pygame.K_s:
    state.shoulder_pitch -= step_rot

# Локоть
elif key == pygame.K_e:
    state.elbow_pitch += step_rot
elif key == pygame.K_q:
    state.elbow_pitch -= step_rot

# Зап'ястя (roll)
elif key == pygame.K_z:
    state.wrist_roll -= step_roll
elif key == pygame.K_x:
    state.wrist_roll += step_roll

# Захоплювач
elif key == pygame.K_c:
    state.grip_angle += step_grip
elif key == pygame.K_v:
    state.grip_angle -= step_grip

# Скидання
elif key == pygame.K_r:
    state.reset()

state.clamp()

# -----
# ОСНОВНИЙ ЦИКЛ
# -----
def main():
    pygame.init()
    width, height = 1000, 700
    pygame.display.set_mode((width, height), DOUBLEBUF | OPENGLE)
    pygame.display.set_caption("3D Робот-маніпулятор (PyOpenGL + Pygame)")

    init_gl(width, height)

    clock = pygame.time.Clock()
    state = RobotState()
    cam = Camera()

    running = True
    while running:
        for event in pygame.event.get():
            if event.type == QUIT:
                running = False
            elif event.type == KEYDOWN:
                handle_key(state, cam, event.key)

```

```

glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
cam.apply()

# Підлога/сцена
glPushMatrix()
glTranslatef(0.0, -0.01, 0.0)
draw_box(20.0, 0.02, 20.0, color=(0.15,0.15,0.18))
glPopMatrix()

# Ось світова
glPushMatrix()
glTranslatef(-3.0, 0.0, -3.0)
draw_axis(length=1.2, thickness=2.0)
glPopMatrix()

# Робот
glPushMatrix()
glTranslatef(0.0, 0.0, 0.0)
draw_robot(state)
glPopMatrix()

pygame.display.flip()
clock.tick(60)

pygame.quit()
sys.exit()

if __name__ == "__main__":
    main()

```