Міністерство освіти і науки України
Криворізький національний університет
Кафедра автоматизації, комп'ютерних наук і технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти – магістр

за освітньо-професійною програмою

«Кіберфізичні системи в промисловості, бізнесі та транспорті»

зі спеціальності 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка

тема роботи:

«Інтелектуальна система керування біоморфним роботом»

Виконав студент гр. АКІТР-24-1м _____ Розумець В. О.

Керівник _____ Курганов І. Д.

Нормоконтроль _____ Маринич І. А.

Завідувач кафедри _____ Рубан С. А.

Кривий Ріг 2025

КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет: інформаційних технологій

Кафедра: автоматизації, комп'ютерних наук і технологій

Ступінь вищої освіти: Магістр

Спеціальність: 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка

ЗАТВЕРДЖУЮ
Зав. кафедри: к.т.н. Рубан С. А.

_____

« » червня 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра

студенту групи АКІТР-24-1м Розумця Владислава Олександровича

1. **Тема роботи:** «Інтелектуальна система керування біоморфним роботом»
   Затверджено наказом по університету № 258с від 13.05.2025р.
2. **Термін здачі кваліфікаційної роботи:** 01.12.2025р.
3. **Склад кваліфікаційної роботи:** Пояснювальна записка обсягом 85с., додатки, презентація у Microsoft PowerPoint (10 слайдів) в електронному та друкованому вигляді
4. **Консультанти кваліфікаційної роботи:**
   _____Розділ 1-3_____доц. Курганов І.Д._____
   __Нормоконтроль_____доц. Маринич І.А._____

5. Календарний план:

№	Етапи роботи	Термін виконання
1	Вступ	10.07.2025
2	Розділ 1	25.07.2025
3	Розділ 2	25.08.2025
4	Розділ 3	25.09.2025
5	Висновки	25.10.2025
6	Оформлення кваліфікаційної роботи	20.11.2025
7	Підготовка презентації та графічного матеріалу	28.11.2025
8	Підготовка доповіді до захисту	01.12.2025

6. Дата видачі завдання: 28.06.2025р.

Керівник _____/Курганов І.Д./

7. Запевнення: *Я, Розумець Владислав Олександрович, запевняю, що ця кваліфікаційна робота виконанна самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.*

Із чинним Положеннням про академічну доброчесність *Криворізького національного університету ознайомлена.*

Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі умисних порушень робота не допускається до захисту або оцінюється незадовільно.

Здобувач _____/Розумець В.О./

АНОТАЦІЯ

Розумець В.О. Інтелектуальна система керування біоморфним роботом: кваліфікаційна робота магістра : 174 – Автоматизація, комп'ютерно – інтегровані технології та робототехніка. Кривий Ріг. Криворізький національний університет, 2025. 84 с.

Об'єктом дослідження є діагностика трубопроводів за допомогою робот-змії

У першому розділі розглянуто особливості застосування біоподібних роботів у неруйнівному контролі та класифіковано їх. Досліджено Програмне забезпечення робототехнічних систем.

У другому розділі представлено математичне та алгоритмічне моделювання процесів , проєктування біоморфного робота змії, а також висунуто пропозиції щодо вдосконалення цих систем.

У третьому розділі представлено формування конструктивної та керувальної частин біоморфного робота-змії. Наведено принципи роботи сервоприводів Dynamixel AX-12 та характеристику електронної системи на базі мікроконтролера ARM Cortex-M4. Розроблено програмну архітектуру системи управління, що включає моделювання сегментів, сенсорну підсистему, візуалізацію кінематики та анімацію руху. Інтегровано три спеціалізовані модулі: ARC-OPT для оптимізації траєкторій, BOLeRo для навчання поведінки та HyRoDyn для розрахунку гібридної динаміки. Реалізовано бібліотеку шаблонних рухів, систему оновлення сенсорних даних та механізм експорту результатів.

Ключові слова: автоматизована система управління, безпілотні літальні апарати, біоморфний робот, біоморфний робот-змія, магнітні гусеничні роботи, неруйнівний контроль,

операційні технології, система автоматичного управління, система управління.

ANNOTATION

Rozumets V.O. “Intelligent control system for a biomorphic robot.”

Qualification work for obtaining a master's degree in the educational and professional program “Cyber-physical systems in industry, business, and transport” in the specialty 174 – Automation, computer-integrated technologies, and robotics – Kryvyi Rih National University, Kryvyi Rih, 2025.

The object of the study is the diagnosis of pipelines using a snake robot.

The first chapter discusses the features of the use of bio-like robots in non-destructive testing and classifies them. The software of robotic systems is investigated.

The second chapter presents mathematical and algorithmic modeling of processes, the design of a biomorphic snake robot, and proposals for improving these systems.

The third chapter presents the formation of the structural and control parts of a biomorphic snake robot. The principles of operation of Dynamixel AX-12 servo drives and the characteristics of the electronic system based on the ARM Cortex-M4 microcontroller are presented. The software architecture of the control system has been developed, which includes segment modeling, a sensor subsystem, kinematics visualization, and motion animation. Three specialized modules have been integrated: ARC-OPT for trajectory optimization, BOLeRo for behavior training, and HyRoDyn for hybrid dynamics calculation. A library of template movements, a sensor data update system, and a result export mechanism have been implemented.

Keywords: automated control system, unmanned aerial vehicles, biomorphic robot, biomorphic snake robot, magnetic tracked robots, non-destructive testing,

operational technologies, automatic control system, control system.

ЗМІСТ

ВСТУП

Важливість теми дослідження. Недеструктивний контроль (НДК) охоплює різноманітні методи аналізу, які використовуються в цивільних, медичних та промислових галузях для оцінки властивостей матеріалів, тканин, деталей чи конструкцій без їхнього пошкодження. НДК відіграє ключову роль у забезпеченні надійності життєво важливих компонентів та загальної безпеки суспільства.

Сучасні елементи все частіше мають складні форми та геометрію, що ускладнює процес контролю. Застосування традиційних методів, що вимагають ручного втручання, породжує низку труднощів та обмежує можливості НДК.

Роботизація приносить значні переваги для НДК, даючи змогу відповідати зростаючим вимогам завдяки підвищенню швидкості та достовірності перевірок. Крім того, роботи можуть здійснювати контроль у зонах, куди операторам важко дістатися, а також усувають необхідність присутності людини у потенційно небезпечних умовах. Однак висока складність технологій та їхня значна вартість стримували широке впровадження роботів у сферу НДК. З цієї причини потенціал поєднання роботизованих платформ із сенсорами, виконавчими механізмами та програмним забезпеченням досі не розкрито повністю.

Попри це, подібні технології здатні кардинально змінити підходи до виконання роботизованого НДК. Роботи, як правило, працюють за заздалегідь визначеними траєкторіями, що формуються за допомогою автономних систем планування шляху. Останні досягнення в електроніці, робототехніці, сенсорних технологіях та програмному забезпеченні відкривають шлях до інноваційних рішень у сфері роботизованого НДК. Активно розвиваються також дослідження, сфокусовані на використанні біологічних засад у робототехніці для імітації природних моделей поведінки та вирішення комплексних завдань.

Сучасні цифрові інновації, зокрема штучний інтелект, цифрові двійники та Інтернет речей, значно розширюють можливості неруйнівного контролю (НДК). Вони покращують як процеси контролю якості на виробництві, так і

забезпечують безпеку експлуатації. Оскільки Індустрія 4.0 базується на даних, а НДК є їхнім основним джерелом, виник новий напрям – НДК 4.0. Він інтегрує традиційні методи неруйнівного контролю з передовими цифровими рішеннями.

Наша мета – оптимізувати систему управління біоморфним роботом, який буде використовуватися для завдань НДК в контексті Індустрії 4.0

Щоб досягти поставленої мети було визначено наступні завдання:

з'ясувати сутність поняття «біоподібний робот» та виявити особливості його використання у неруйнівному контролі;

дослідити класифікацію та конструктивні особливості біоподібних роботів; провести аналіз системи управління таким роботом;

визначити головні напрямки вдосконалення системи керування біоподібним роботом;

здійснити практичні кроки зі створення, розробки дизайну та моделювання біоподібного робота;

Об’єкт дослідження – Діагностика трубопроводів за допомогою робот-змії.

Предмет досліджень – це інтелектуальна система керування біоморфним роботом, підвищення ефективності використання неруйнівному контролі.

Отримані результати: уточнено термінологію «біоподібний робот» та визначено особливості його застосування у неруйнівному контролі; згруповано різновиди біоподібних роботів; висвітлено особливості системи керування біоподібним роботом; запропоновано ключові вектори вдосконалення системи керування; проведено проєктування та моделювання біоподібного робота.

Наукова новизна отриманих результатів.

Вперше надано визначення поняття біоморфного робота.

Розроблено систему управління біоморфного робота.

Практичне значення отриманих результатів. Підвищення продуктивності системи керування біоподібним роботом розширює сферу їхнього потенційного застосування як у НДК, так і в інших галузях.

У першому розділі розглянуто особливості застосування біоподібних роботів у неруйнівному контролі та класифіковано їх. Другий розділ присвячено

конструктивній характеристиці та розробці системи керування біоморфним роботом, а також висунуто пропозиції щодо вдосконалення цих систем. Третій розділ містить практичні матеріали з проєктування, розробки та 3-D моделювання біоподібного робота.

РОЗДІЛ 1

ЗАСТОСУВАННЯ БІОМОРФНИХ РОБОТІВ У ПРОЦЕСАХ НЕРУЙНІВНОГО КОНТРОЛЮ

1.1 Сфера та роль біоморфних роботів у неруйнівному моніторингу

Огляд різних підходів до визначення терміна робот, що імітує біологічне свідчить про те, що наразі частіше вживаються поняття біоробот та біоміметичний робот. Таким чином, робота, подібна до біологічної, трактується як пристрій чи система автоматизованого типу, розроблена для здійснення різноманітних функцій та операцій, чия конструкція ґрунтується на відтворенні будови чи функціональних аспектів живих тканин або організмів.

Недеструктивний контроль (НДК) є міждисциплінарним сукупністю аналітичних способів, які знаходять застосування як у науковій царині, так і у виробництві для оцінювання властивостей матеріалів та/або підтвердження цілісності елементів та споруд без їхнього пошкодження. НДК забезпечує надійність та відповідність виробленої продукції заданим вимогам, дозволяє контролювати виробничі етапи, сприяючи зниженню витрат і підтримці стабільної якості. За відсутності НДК, безпека та функціональність компонентів можуть опинитися під серйозною загрозою. Це перетворює НДК на критично важливий інструмент у запобіганні масштабним аваріям, зокрема авіаційним та залізничним катастрофам, вибуховим подіям та витокам на газо- та нафтопроводах, збоям у роботі атомних реакторів, морським пригодам та іншим техногенним інцидентам. Ключові сектори застосування НДК включають аерокосмічну, авіаційну та аеронавігаційну галузі; енергетику (атомну, вітрову, електричну); нафтогазову та нафтохімічну індустрії; хімічне виробництво; об'єкти інфраструктури (мости, будівлі, автошляхи); транспортні сектори (автомобільний, залізничний, морський); будівельну сферу та інші.

Інтеграція традиційних методів НДК ускладнюється без застосування автоматизованого обладнання для ідентифікації складних частин, особливо коли йдеться про високоточний неруйнівний аналіз елементів із поверхнею, що має мінливу кривизну, неоднорідну товщину та складну геометрію. З огляду на це, роботизовані системи для НДК, які поєднують маніпуляційні технології з засобами неруйнівного моніторингу, здатні суттєво підвищити ефективність та точність інспекційних процедур. Роботизований НДК зводить разом фізичні засади методів контролю із високою маневреністю та просторовою прецизійністю керування позицією маніпулятора. В умовах обмежень, притаманних неруйнівному контролю, подібні системи здатні задавати необхідні траєкторії руху та кутові позиції для передавальних і приймальних сенсорів. Внаслідок цього, класичні підходи НДК зазнали трансформації: від обробки площинних поверхонь до аналізу криволінійних, від двовимірних схем до багатовимірних, від ручного керування — до інтелектуалізованих стратегій, формуючи унікальну міждисциплінарну технологію роботизованого НДК.

Автоматизація процесів НДК елементів конструкцій та вузлів є одним із пріоритетних завдань у численних промислових галузях. Це сприяє поліпшенню точності та швидкості інспектування, одночасно скорочуючи час перевірок та пов'язані з ними витрати на людський ресурс, що різко контрастує з ручними методиками. Використання робототехніки забезпечує підвищену гнучкість та автономність функціонування автоматизованих НДК систем. Застосування роботизованих інспекційних комплексів є корисним у широкому діапазоні виробничих сценаріїв — від НДК, що вбудовується у процес створення деталей складної конфігурації (наприклад, зубчастих коліс), до регулярного капітального огляду об'єктів великого розміру під час їхньої експлуатації (зокрема у нафтохімічному та аерокосмічному секторах).

Перевірка великої кількості деталей (наприклад, у автопромисловості) або масивних споруд (як-от корпуси літаків) вимагає значних часових та трудових ресурсів. Деякі методи НДК, зокрема візуальна інспекція, контроль вихровими струмами та ультразвуковий аналіз, добре піддаються автоматизації, що

стимулює зростання зацікавленості у розробці роботизованих систем неруйнівного контролю.

Останнім часом ми спостерігаємо стрімкий розвиток технологічних рішень, що залежить від вимог, цільових сфер застосування та вартості, асоційованої з різними типами робочих завдань. Неабиякої популярності набуло також використання концепцій, відомих як біоніки, біоміметики чи біомімікрії.

Біомімікрія трактується як процес відтворення властивостей або способів функціонування живих організмів (рослин і тварин) у процесі пошуку рішень їхніх природних проблем чи завдань.

Концепції «біоніка» та «біоміметика» щільно пов'язані з біомімікрією. Слово «біомімікрія» походить від двох грецьких коренів: «біос» — природа, та «мімезис» — уподібнення чи імітація. Цей термін охоплює створення нових систем для вирішення інженерних викликів шляхом копіювання природних рішень або запозичення принципів із природних структур і процесів. Біоніка та біоміметика зазвичай розглядаються як поняття зі схожим значенням, оскільки їхня концептуальна база подібна. Однак ключова відмінність полягає у походженні самих термінів. Термін «біоніка» був вперше запропонований Джеком Стілом у 1960 році й описує розробку сучасних технічних систем або функціональних модулів, які базуються на аналогах, що існують у природі [11]. Біоніка здебільшого стосується відтворення біологічних систем за допомогою технічних засобів, і цей термін часто фігурує у контексті протезування, штучного покриття шкіри тощо [15].

Концепцію «біоміметика» розкрив у 1969 році Отто Шмітт. Він визначав її як процес наслідування формування, внутрішньої будови або функціональності матеріалів чи речовин, створених біологічно, з метою продукування чи синтезу штучного еквівалента. Це явище може бути застосоване до структур, механічних принципів, процесів або функцій [19].

У промисловості вже набули поширення роботи-альпіністи та мініатюрні самохідні платформи для проведення НДК, здатні функціонувати на великих конструкціях, куди доступ персоналу є обмеженим чи несе ризик. Переваги, які

надають роботизовані НДК системи, представлені у таблиці 1.1.

Таблиця 1.1 – Переваги застосування роботіву НРК

№п/п	Переваги	Пояснення
1.	Підвищена точність	Мінімізація ризику людської помилки та забезпечення точних та стабільних результатів
2.	Дистанційна перевірка	Використання для небезпечних та важкодоступних місць
3.	Послідовність	Роботи підтримують постійну швидкість і методологію перевірки, зменшуючи мінливість результатів у порівнянні з інспекторами-людьми
4.	Ефективність	Прискорення процесу перевірки, що призводить до швидшого часу виконання робіт і підвищення продуктивності
5.	Повторюваність	Можуть багаторазово перевіряти одні й ті ж компоненти з однаковим рівнем точності, що робить їх ідеальними для контролю якості та довгострокового моніторингу
6.	Збір даних	Можливість збирання та запису значного обсягу даних під час перевірок, які можна аналізувати на предмет тенденцій та використовувати для прогнозованого технічного обслуговування
7.	Скорочення часу простою	Час простою для технічного обслуговування зводиться до мінімуму
8.	Рентабельність	Зменшуються витрати на оплату праці та підвищується загальна ефективність.
9.	Безпечність	Підвищення безпеки та зниження ризику нещасних випадків або впливу небезпечних речовин
10.	Операції 24/7	Роботи можуть працювати цілодобово, забезпечуючи безперервний моніторинг та інспекцію, що особливо цінно в таких галузях, як виробництво енергії та аерокосмічна промисловість

Як демонструє таблиця, впровадження роботів у НРК відкриває шлях до посилення контрольних можливостей, доступу до зон, куди важко дістатися, а також до зростання ефективності інспекцій, навіть незважаючи на те, що первісні фінансові вкладення у роботизовані комплекси можуть бути значними. Проте у перспективі тривалого часу вони сприяють відчутному скороченню витрат. Роботи здатні функціонувати безперервно, у режимі 24/7, що веде до зменшення

витрат на людський ресурс. Крім того, вони виконують обстеження оперативніше, чим скорочують час незапланованих зупинок виробництва та збільшують загальну продуктивність; забезпечують вищий рівень безпеки для персоналу; надають можливість отримувати дані у режимі реального часу (інформація може бути зібрана та оброблена безпосередньо на місці, що дає змогу оперативно приймати необхідні рішення). Ключовими сферами їхнього застосування є, насамперед, обмежені простори, середовища з екстремальними температурами, а також зони, де присутні небезпечні речовини.

Інженерні рішення, засновані на біомімікрії, вважаються важливим рушієм інновацій та швидко здобувають популярність не лише у високотехнологічних галузях, а й у більш традиційних сферах, зокрема у матеріалознавстві та нанотехнологіях. Було проведено численні дослідження, сфокусовані на розробці інтелектуальних матеріалів, модифікаторів поверхні, нанокомпозитів та іншої продукції на засадах біомімікрії. Нанотехнології є ще однією областю, де біоміметика слугує інструментом для генезації нових рішень. Біоміметичні підходи також виступають каталізаторами екологічно сталого розвитку, оскільки сприяють формуванню еко-стабільних технологій через запозичення принципів із природного середовища.

Наукові роботи у площині біоміметичної робототехніки наразі активно ведуться у численних дослідницьких центрах світу з метою створення нового класу біоміметичних роботів, які продемонструють значно вищий рівень компетенції та надійності при роботі у неструктурованих умовах порівняно з тими роботами, що використовуються сьогодні. Передбачається, що роботи, сконструйовані на основі біоміметичних засад, будуть більш гармонійно взаємодіяти з довкіллям та демонструватимуть більшу стабільність у складних обставинах. Фахівці, які проєктують такі системи, використовують переваги нових матеріалів, передових методів виробництва, інноваційних сенсорних модулів та виконавчих механізмів, що розширює можливості для імітації біологічних процесів і систем. Біоміметичний механізм являє собою напівавтономну чи повністю автономну систему, здатну функціонувати у

складному оточенні. Такий механізм може бути спроєктований таким чином, щоб зберігати здатність до адаптації до непередбачуваних змін та виконувати різноманітні функції.

Біоміметику умовно класифікують за трьома основними напрямками: форма й експлуатаційні характеристики; біокібернетика, системи сприйняття та власне робототехніка; нанобіоміметика [19].

Загалом, ключова увага приділяється саме функціональному механізму, а не обов'язково біологічній формі його втілення, що варто брати до уваги при реалізації проєктів.

Біоміметика — це сукупність взаємодій між біологією та інженерією, спрямована на розв’язання прикладних завдань шляхом аналізу природних систем та трансферу їхніх принципів у технічні рішення. На рисунку 1.1 представлене застосування біоміметики та механічний принцип функціонування робота, здатного пересуватися по вертикальній поверхні.

1. (а) робот для лазіння (б) гусениця (в) 3D-візуалізація

Рисунок 1.1 – Біоміметика та механічний принцип робота-лазача на стовпі [4]

З огляду на стрімкий прогрес у сфері новаторських матеріалів, передових конструкційних рішень та найновіших підходів до управління, біонічні роботи еволюціонують. Вони відходять від простого копіювання вигляду та рухів живих істот, прямуючи до створення біоміметичних систем, де електромеханічні конструкції інтегруються з фундаментальними біологічними засадами.

1.2 Роботизовані системи та їх типи

Серед ключових категорій роботизованих систем, що активно застосовуються у світі для виконання завдань НРК, виділяються наступні:

1. безпілотні літальні апарати (БПЛА) або дрони;
2. повзуни та альпіністи;
3. крабові системи;
4. магнітні гусеничні системи;автономні підводні апарати;
5. роботизовані руки;
6. колаборативні роботи;
7. моделі штучного інтелекту.

Дистанційно керовані транспортні засоби (ДКТЗ)

Сфера застосування: знаходять своє застосування для інспектування під водою (перевірка об’єктів інфраструктури, газо- та нафтопроводів, обшивок суден).

Характерні риси: Обладнані широким спектром датчиків та відеокамер. Вони спроможні фіксувати зображення, проводити ультразвукове вимірювання товщини, а також виконувати неруйнівний контроль методом магнітної дефектоскопії безпосередньо під водою. Керування цими апаратами здійснюється з поверхні (як правило, з надводної одиниці) за допомогою наземної системи контролю, де оператор у режимі реального часу ухвалює рішення та стежить за функціонуванням апарата. Одним із ключових напрямків розвитку для підводних апаратів неруйнівного контролю (ПАНК) є перехід до більшої "автономності" для виконання певних спеціалізованих завдань, як-от фіксація позиції, динамічне утримання місця, автоматичне визначення вектора руху чи підтримання заданої глибини [54].

Складнощі: Невизначеність параметрів (зокрема, додаткової маси, коефіцієнтів гідродинаміки тощо). Ця складність посилюється за рахунок модульності сучасних ПАНК (наприклад, можливість апарата працювати з

різноманітним інструментальним оснащенням чи напрямними елементами: з одним чи двома маніпуляторами, модулем для гідроструминного очищення, поворотним щітковим модулем, блоком для інспектування трубопроводів, пристроями для обрізання дротів і кабелів, а також роторним дисковим різаком та ін.).

Крім того, висока мінливість умов підводного середовища створює значні перешкоди для роботи апарата, такі як течії та вплив хвиль на мілководді.

Керування за допомогою ковзного режиму дає змогу ефективно долати окреслені проблеми, тому цей метод вважається одним із багатообіцяючих прийомів для керування підводними апаратами та адаптації до змін у їхній моделі, оскільки невизначеності та зовнішні чинники можуть бути нівельовані складовою ковзного керування. Однак усім відомо, що традиційний метод ковзного керування (SMC) провокує виникнення високочастотних складових у системі, що призводить до інтенсивного перемикання в рушійних механізмах, яке, у свою чергу, прискорює їхнє зношування та скорочує загальний термін служби.

Безпілотні літальні апарати (БПЛА) або дрони.

Використання: повітряна інспекція великомасштабних конструкцій (мости, вітрові турбіни, промислові об’єкти).

Особливості: обладнані камерами з високою роздільною здатністю та тепловізійними технологіями (Рис.1.2). Вони здатні захоплювати візуальні дані, інфрачервоні зображення та інші матеріали, які допомагають виявляти дефекти або аномалії. Для підйому використовують аеродинамічні сили, можуть літати автономно або під дистанційним керуванням [37].

Рисунок 1.2 – Ультразвуковий дрон Voliro T для НРК

Безпілотні літальні апарати (БПЛА), які також широко відомі як дрони, можуть бути класифіковані за низкою критеріїв, що відображають їхні технічні та експлуатаційні особливості. Зокрема, класифікація здійснюється за принципом польоту, цілями виконуваної місії, злітною масою, типом силової установки, способом керування, максимальною висотою та дальністю польоту, конструктивним виконанням, функціональним призначенням, методом запуску й посадки, типом та масою корисного навантаження, рівнем автономності, габаритними розмірами, витривалістю у повітрі та тривалістю польоту. Загальноприйняті категорії БПЛА, що узагальнюють ці ознаки, наведені у таблиці 1.2.

Окрім різноманітності класифікаційних підходів, БПЛА мають низку суттєвих переваг, які зумовлюють їх широке застосування у цивільній та спеціалізованій сферах. До ключових плюсів належать невелика маса та зручність транспортування, що дозволяє оперативно розгортати систему у польових умовах.

Сучасні дрони оснащуються розширеними функціями навігації та орієнтування на місцевості, зокрема із використанням супутникових систем і цифрових карт. Налаштовувані комплекти сенсорів дозволяють адаптувати БПЛА під конкретні завдання, забезпечуючи миттєве отримання інформації про поточну обстановку. Додатковою перевагою є круговий огляд на 360 градусів за допомогою відеосистем, що значно підвищує ситуаційну обізнаність оператора.

Конструкція корпусу БПЛА, як правило, відзначається високою міцністю

та наявністю захисту від впливу зовнішніх факторів, зокрема магнітних полів і вологи, відповідно до стандарту IP65.

Таблиця 1.2 – Класифікація БПЛА

№ п/п	Принцип класифікації	Характеристика
1.	Маса	висотно-довготривалі БПЛА, міні-БПЛА, мікро БПЛА, тактичні БПЛА, БПЛА середньої висоти та довготривалості, тощо
2.	Корисне навантаження	можуть нести різне корисне навантаження, таке як камери, датчики, системи зв'язку, вантажі, зброя,та інші
3.	Ціль	відповідно до їх цілі, наприклад пошук і порятунок, спостереження, розвідка, атака, транспортування, тощо.
4.	Силова установка	Живляться від паливних, сонячних електричних або інших джерел.
5.	Конфігурація	нахил-ротор, моноротор, нахил-крило мультиротор та інші.
6.	Діапазон висоти	висотні БПЛА,низьковисотні БПЛА та стратосферні БПЛА
7.	Принцип дії	Різноманіття безпілотних літальних апаратів (БПЛА) охоплює апарати з фіксованим крилом, поворотним крилом, гібридні конструкції, а також апарати з махаючим крилом й інші типи, принципи польоту яких ґрунтуються на різних механізмах руху.
8.	Радіус дії	БПЛА середньої дальності, БПЛА малої дальності, БПЛА великої дальності
9.	Спосіб запуску	запускаються з землі, повітря, моря, можуть мати інші типи способів запуску
10.	Витривалість	БПЛА з довгою витривалістю, короткою витривалістю, БПЛА наддовгої витривалості тощо.
11.	Рівень автономності	можуть мати різні рівні автономності, такі як керовані людиною, напівавтономні, повністю автономні та інші
12.	Призначення	цивільне,військове, комерційне, наукове, промислове, тощо.
.13	Управління	Пілотованими, дистанційно, напівавтономними, автономними або мати інші типи управління

Крабові системи

Привід робота оснащений шістьма підпружиненими важелями, що забезпечують його переміщення (див. рис. 1.3 б). У поєднанні з планетарною трансмісією, 32 радіальними кульковими підшипниками та 12 гумовими колесами, ця система гарантує максимальне зчеплення в трубах. Це сприяє ефективному проведенню контролю. Підпружинені важелі також забезпечують стабільне центрування робота в трубі, дозволяючи йому адаптуватися до нерівностей та долати перешкоди.

Унікальна конструкція роботів дозволяє їм успішно проходити круті повороти (всього 0,8x D) та переміщатися вертикально в межах трубопроводів (див. рис. 1.3 б). Ці установки спеціально створені задля ефективної навігації в трубопроводах із діаметром від 50 до 1000 мм (див. рис. 1.3 в) [37].

а) Навігаційні можливості б) Робот в) Робот в трубопроводі

Рисунок 1.3 – Робот Endo Control Crab Crawler Robot [37]

Guardian S – це універсальний робот, портативна мобільна платформа IoT, що має різноманітні датчики, які постачають інформацію в реальному часі, забезпечуючи безпеку оператора.

Рисунок 1.4 – Модель Sarcos Guardian S [9, 37]

Компанія Sarcos Robotics явила світові справді захоплюючу ідею робота, котрий має здатність просочуватися у вкрай важкодоступні закутки. Цей пристрій за обрисами вельми подібний до змії. Умовно кажучи, його конструкцію можна розкласти на три умовні сегменти: передній, серединний та задній. І передній, і задній відсіки обладнані гусеничним рушієм, тоді як центральний вузол являє собою еластичний механізм, який забезпечує роботу здатність до маневрування у заплутаних і тісних просторах.

Розроблений з урахуванням роботи в умовах повної непередбачуваності, апарат під назвою Guardian S призначений для пересування нерівними, складними теренами та через вузькі тунелі. Прикметною рисою є також його значна автономність роботи без потреби живлення та здатність функціонувати без дротів на пристойних дистанціях, що виводить на новий рівень ефективність проведення оглядів та дистанційного спостереження [9].

Деталізація технічних параметрів цього робота представлена у Додатку А.

Магнітні гусеничні роботи (МГР)

ВЗастосування: для інспектування поверхонь чорних металів; зокрема у нафтогазовій галузі.

Характеристики: МГР розроблені для роботи на рівних та помірно вигнутих сталевих площинах.

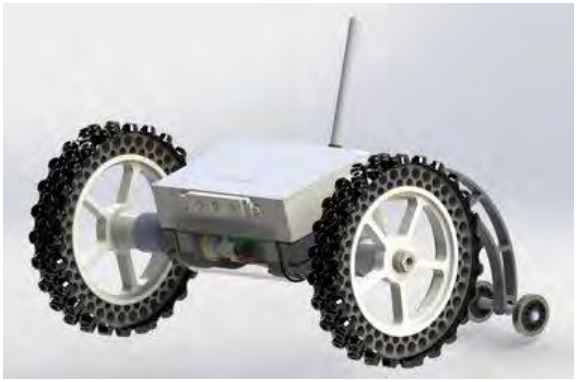


Рисунок 1.5 – 3D-візуалізація Magnet Crawler II (ескіз CAD: Фелікс Бернхард, DFKI) [37]

Залежно від потреб, на цю платформу можна легко встановити повний комплект змінних приладів, що базуються на гідроабразивній чи піскоструминній обробці. Вони здатні пересуватися зовнішніми поверхнями труб та ємностей, здійснюючи магнітопорошкову дефектоскопію та ультразвукові виміри товщини з метою виявлення корозійних уражень та інших недоліків.

Magnet-Crawler II – це мобільна, невелика роботизована одиниця для переміщення по магнітних поверхнях. Апарат використовує два мотор-редуктори для руху своїх магнітних роликів, а еластичний "хвіст" забезпечує йому кращу стабільність. У передній частині робота розташована система Odroid, об'єднана з поворотним модулем камери 720p, платою вводу/виводу, парою світлодіодних прожекторів та одним світлодіодним променем для підсвічування. Живлення Magnet-Crawler II забезпечують літій-полімерні акумулятори, а керування відбувається через бездротову мережу WLAN. Відеодані з робота передаються на термінал управління оператора. Робот ефективно застосовується для обстеження обшивок кораблів. Технічні параметри Magnet-Crawler II представлені у таблиці 1.3.

Таблиця 1.3 – Технічні характеристики Magnet-Crawler II

Показники	Характеристика
Датчики:	3-осьовий датчик прискорення
Розмір:	330 x 300 x 130 мм
Вага:	1230 г
Живлення:	11,1 В – літій-полімерна батарея 800 мАг (час роботи близько 30 хв.)
Привід/двигун:	2 мотор-редуктори 12 В постійного струму з енкодерами

Джерело: [37]

Автономні підводні апарати (АПА)

Застосування: дляінспектування корпусів суден, трубопровідних систем і підводних конструкцій

Характеристики: це запрограмовані роботизовані апарати, що здатні, залежно від своєї конструктивної реалізації, дрейфувати, пересуватися чи ковзати морськими просторами без прямого керування оператором у режимі реального часу. Деякі пристрої підтримують періодичний або постійний зв'язок з операторами через супутникові сигнали чи підводні акустичні системи для забезпечення певного рівня контролю. На відміну від апаратів, що керуються дистанційно і прив'язані до судна-носія, ці автономні системи не мають фізичного з'єднання з оператором. Вони отримують настанови або запрограмовані заздалегідь операторами, які можуть знаходитися на судні або навіть на суші. На відміну від інших роботизованих рішень, які майже миттєво передають відеодані через кабелі до командних пунктів на кораблях, автономні апарати зберігають усі накопичені відомості, включаючи зображення та інші дані сенсорів, на вбудованих комп'ютерах до моменту їхнього вилучення після повернення апарату на поверхню після завершення місії.

Габарити цих апаратів можуть коливатися від декількох сотень до кількох тисяч фунтів. Вони здатні переміщатися з водневої поверхні на значні глибини океану і назад, а також можуть завмирати, утримувати позицію чи змінювати

положення подібно до того, як це роблять дирижаблі чи гелікоптери у повітрі. Повністю незалежні місії забезпечуються бортовим живленням. Ця енергія необхідна для роботи рушіїв чи двигунів, що забезпечують рух апарату у воді, а також для живлення датчиків, встановлених на борту. Переважна більшість використовує спеціалізовані акумуляторні батареї, хоча деякі моделі можуть застосовувати паливні елементи або накопичувати енергію від сонячних панелей. Окремі типи, як-от планери, мінімізують потреби в енергії, використовуючи для руху сили тяжіння та плавучості.

Роботизована рука

Застосування: універсальні роботи, придатні для використання у широкому спектрі галузей.

Характеристики: біонічні маніпулятори переважно створюються з метою імітації людської руки та відтворення функціональних можливостей і характеристик, властивих людським діям. Порівняно зі звичайними механічними захватами, роботизовані біонічні маніпулятори спроможні ефективно виконувати поставлені завдання в різноманітних складних умовах завдяки високій продуктивності, низькому енергоспоживанню та значній пристосованості до середовища.

Роботи з біонічними руками поділяються, головним чином, на однорукі та дворукі біонічні системи. Дворукі біонічні роботи не лише можуть координувати роботу обох кінцівок, як це роблять однорукі антропоморфні роботи, але й демонструють високу стійкість до збоїв. Це суттєво покращує експлуатаційні показники та розширює сферу застосування біонічних дворуких роботів при роботі з комплексними завданнями [10].

Колаборативні роботи

Характеристики: розроблені для роботи пліч-о-пліч з інспекторами, доповнюючи їхні можливості та підвищуючи рівень безпеки. Вони можуть працювати з використанням ультразвукового контролю чи вихрострумових зондів, поки оператор здійснює загальне управління процесом обстеження.

Моделі штучного інтелекту

Характеристики: інтегруються з різноманітними системами візуалізації: від стандартних камер відеоспостереження та портативних камер до складних чотириногих платформ та безпілотних літальних апаратів (Дрон.1.6).

Рисунок 1.6 – Модель штучного інтелекту Levatas [37]

Спеціально створені повністю робочі моделі можуть застосовуватися разом із Boston Dynamics Spot, роботом-собакою, для зчитування аналогових датчиків, фіксації теплових аномалій та визначення змін.

1.3 Дослідження програмного забезпечення робототехнічних систем

Програмне забезпечення біоподібних роботів має ключове значення для їхнього функціонування. Нижче наведено опис основного.

ARC-OPT (Адаптивне керування роботом за допомогою оптимізації) є комплексною платформою для оптимізаційного керування робототехнічними системами. Ця система дозволяє користувачам легко визначати, втілювати та оптимізувати складні реактивні завдання роботів, які складаються з множинних паралельних підзадач. ARC-OPT надає широкий спектр моделей роботів та потужних розв'язувачів, що уможливлюють реалізацію різноманітних стратегій керування, включаючи ієрархічні, зважені та гібридні підходи, для ефективного вирішення складних робочих завдань.

з великою кількістю ступенів свободи, зокрема гуманоїдах [6].

Bagel (Мова програмування біологічно натхненна мова на основі графів) використовує графову структуру для опису програм. Кожен вузол у графі

представляє собою окремий алгоритм, який приймає дані на вхід і передає їх на вихід. Зв'язки (ребра) між вузлами показують, як дані рухаються від одного алгоритму до іншого. Граф Bagel також може мати загальні входи та виходи, що дозволяє розглядати цілий граф як один великий "Бублик-вузол" на вищому рівні. Це означає, що спосіб взаємодії з окремими алгоритмами та спосіб взаємодії з їхніми комбінаціями (складеними графами) виглядає однаково. Таким чином, одні й ті самі графи можна використовувати як для створення нових алгоритмів, так і для побудови складних систем. Крім того, можливість об'єднувати кілька вхідних даних в один за допомогою різних функцій злиття робить Bagel ефективним для моделювання ієрархічних структур, що базуються на поведінці.

Оптимізація поведінки та навчання роботів (BOLeRo) надає інструменти для формування поведінкових моделей роботів [16]. Це охоплює представлення поведінки, навчання з підкріпленням, оптимізацію «чорного ящика», еволюційні алгоритми та імітаційне навчання. Забезпечує інтерфейс C++ і Python, що дозволяє бути ефективним там, де це критично, і водночас гнучким і зручним там, де продуктивність не є визначальною. Оскільки бібліотека надає інтерфейс C++, її легко інтегрувати в більшості робототехнічних фреймворків, наприклад у операційну систему роботів (ROS) або конструктор роботів (Rock).

CAD-2-SIM дозволяє спростити процес переходу від комп'ютерного проектування до моделювання. Цей інструмент призначений для автоматичного перенесення даних про механізми з систем автоматизованого проєктування (CAD) до програм для симуляції [17]. Він дозволяє експортувати ключові числові параметри, що визначають кінематику та динаміку механізму, без необхідності ручного введення. Це означає, що дані можуть бути безпосередньо передані з CAD-системи у формати, сумісні з різними симуляційними платформами. Така пряма інтеграція прискорює та стабілізує процес розробки. CAD-2-SIM спирається на нотацію Шет-Уікера та графову схему перерахування.

Hybrid Robot Dynamics (HyRoDyn) — це модульний програмний інструментарій, реалізований мовою C++, призначений для розв’язання задач

кінематики та динаміки складних послідовно-паралельних гібридних роботів, популярність яких останнім часом суттєво зросла. Такі роботи поєднують властивості та структурну складність як послідовних, так і паралельних архітектур, що вимагає спеціалізованих методів аналізу та керування. На відміну від класичних підходів, які спираються на неявне розв'язання циклічних обмежень, HyRoDyn застосовує цілісну стратегію роботи зі складними механізмами, використовуючи замкнені аналітичні розв'язки для обмежень паралельних циклів, характерних для гібридних конструкцій [25]. Бібліотека HyRoDyn може бути розширена кінематиками шляхом додавання нових підмеханізмів, а також підтримує інтеграцію довільних паралельних механізмів за допомогою чисельних явних методів, зберігаючи при цьому гнучкість і високу ефективність. Загалом HyRoDyn виступає надійною альтернативою загальновживаним кінематико-динамічним пакетам.

Machina Arte Robotum Simulans (MARS) — це кросплатформне програмне середовище для моделювання та візуалізації, орієнтоване на робототехнічні дослідження. Воно включає базовий модуль моделювання, графічний інтерфейс на основі Qt, тривимірну візуалізацію з використанням OSG та фізичний рушій ODE. Модульна архітектура MARS забезпечує високу гнучкість використання, зокрема можливість запуску фізичної симуляції без графічного інтерфейсу або візуалізації [27]. Система підтримує підключення користувацьких плагінів, а також використання широкого набору вже наявних модулів.

Maja Machine Learning Framework (MMLF) — це універсальний фреймворк для задач навчання з підкріпленням, розроблений мовою Python. Він надає набір алгоритмів RL разом із еталонними тестовими середовищами та орієнтований на зручне розширення. MMLF дозволяє автоматизувати процес бенчмаркінгу різних агентів і спрощує проведення порівняльних експериментів [29].

NDLCom — це протокол зв'язку на рівні вузла, призначений для обміну даними в децентралізованих роботизованих системах. Він забезпечує передавання невеликих пакетів інформації між мікроконтролерами, FPGA та центральним комп'ютером через з'єднання типу «точка-точка». Кожен вузол

пересилає повідомлення відповідно до адреси, вказаної в заголовку пакета. Реалізації механізмів приймання, маршрутизації та декодування доступні мовами C/C++ і VHDL [33], а візуалізація, збереження та експорт даних здійснюються за допомогою графічного інтерфейсу.

Phobos — це відкритий плагін для середовища Blender, який забезпечує створення, редагування та експорт моделей роботів для симулятора MARS. Процес формування коректних симуляційних моделей є складним і часто потребує ручного редагування громіздких конфігураційних файлів. Phobos спрощує цю задачу, надаючи інтерактивні візуальні засоби побудови моделей і підтримуючи експорт у формати URDF та SMURF, сумісні з MARS.

pySPACE — це модульна програмна платформа для обробки сегментованих часових рядів і векторних ознак. Вона розроблена з урахуванням розподіленого виконання та емпіричної оцінки ланцюгів обробки сигналів і може використовуватися як для порівняльного аналізу, так і для онлайн-застосувань. Система автоматично завантажує, обробляє та зберігає набори даних, а вузли обробки сигналів і складні операції легко комбінуються. Паралельне виконання можливе на багатоядерних системах або в обчислювальних кластерах.

reSPACE (reconfigurable Signal Processing and Classification Environment) — це реконфігуроване середовище для обробки та класифікації сигналів, орієнтоване на спрощення розробки апаратних прискорювачів на базі FPGA для вбудованих і мобільних систем. Воно використовує модельно-орієнтований підхід, що дозволяє скоротити час створення апаратних рішень, з особливим акцентом на задачі обробки сигналів, зображень та машинного навчання [26].

Rock — це програмний фреймворк для розроблення роботизованих систем, базова компонентна модель якого ґрунтується на Orocos RTT (Real Time Toolkit). Він надає повний набір інструментів для створення, інтеграції та експлуатації високопродуктивних і надійних роботизованих рішень у промислових і наукових застосуваннях. Rock містить велику бібліотеку готових драйверів і компонентів, а також підтримує просте розширення новими модулями [40].

React — це відкрита бібліотека JavaScript, розроблена компанією Meta для

створення користувацьких інтерфейсів. Вона базується на компонентному підході, який забезпечує повторне використання та незалежність елементів інтерфейсу. Ключовою особливістю React є застосування Віртуального DOM, що підвищує продуктивність завдяки оновленню лише змінених частин сторінки. Для опису структури компонентів використовується синтаксис JSX, який поєднує можливості JavaScript та HTML-розмітки, роблячи React основним інструментом для створення сучасних вебзастосунків.

Висновки до розділу 1

На основі проведеного аналізу можна стверджувати, що біоподібний робот — це автоматичний пристрій або система, призначені для виконання різних задач та операцій, спроєктовані з урахуванням імітації структури або функціональних властивостей біологічних матеріалів або природних систем. Під час проєктування біоміметичних роботів основним є не відтворення матеріалів чи структур природи, а розуміння принципів проєктування та фізико-хімічних механізмів, що визначають оптимізовану структурну й геометричну організацію біологічних систем та її зв’язок із численними функціями. Для НРК застосовують різні види біоподібних роботів залежно від функціональних задач та специфічних вимог. Програмне забезпечення є одним з ключових елементів роботи робота.

РОЗДІЛ 2

СИСТЕМИ КЕРУВАННЯ РОБОТАМИ БІОМОРФНОГО ТИПУ

2.1 Комплекс керування для біоморфних роботів

Структура біоподібного робота буде залежати безумовно від типу та виконуваних функцій. Нижче на Рис. 2.1 – 2. 3 наведено кілька прикладів.

Експериментальна робот-платформа SmartHex, зображена на Рис. 2.1, побудована на основі біонічної структури та принципу мінімізації ваги. Її комплектація включає камеру Kinect, силовий модуль, панель управління та систему датчиків. Камери Kinect застосовувалися для навігації, зокрема для виявлення перешкод та розпізнавання рельєфу. Силовий модуль забезпечував живлення всієї системи. Плата управління реалізовувала зв'язок між головним обчислювальним пристроєм та цифровою сервосистемою. Система датчиків призначалася для збору даних про кути орієнтації та споживаний струм.

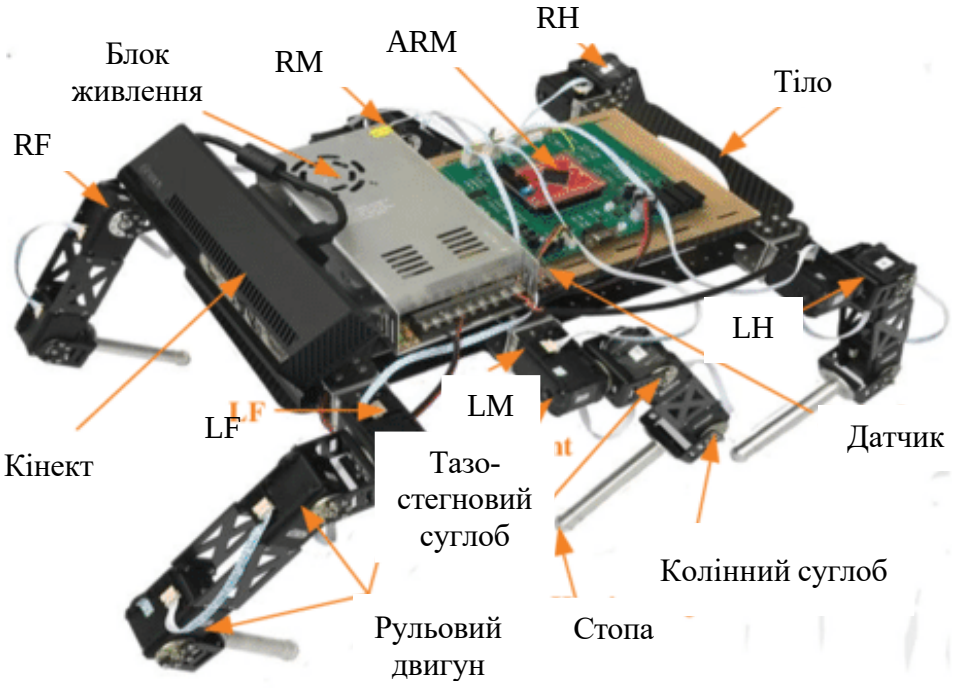


Рисунок 2.1 – Струкутра біоміметичного робота павука [52]

Аналіз будови біоподібних роботів показує, що будь-який робот складається з таких основних компонентів: корпусу (тіла), системи управління, сенсорної системи, джерела живлення, засобів зв’язку та виконавчих механізмів. Система управління — це комплекс, який відповідає за контроль вихідного стану робота. По суті, це пристрій або сукупність пристроїв, що виконують функції керування, видачі команд і регулювання роботи іншого пристрою або системи за допомогою зворотного зв’язку. Завдяки цьому вона забезпечує управління і регулювання роботи керованого об’єкта. Система складається з різних елементів, пов’язаних між собою для спільного виконання процесу керування. Її структура формується через взаємодію компонентів, що дозволяє створити конфігурацію, здатну управляти системою. Головними елементами системи управління є виконавчі механізми, датчики, задаючий (еталонний) вхід та сама керована система. Саме під системою розуміють процес або установку, яку потрібно контролювати, виконавчий механізм перетворює сигнал керування у силовий, датчик вимірює вихідні параметри системи, а еталонний вхід задає бажаний результат роботи.

Завдяки системам управління забезпечується контроль рухів і функцій робота, що дозволяє йому виконувати поставлені завдання відповідно до задуму.

Основними компонентами системи є (Рис. 2.2): датчик: здійснює вимірювання змінної системи; контролер: виконує операції порівняння та обчислення; модуль керування: обладнання для реалізації керуючого впливу в процесі, керуючий елемент отримує сигнали від контролера та відповідно до них виконує задані дії.

Датчики в мобільних роботах забезпечують взаємодію із зовнішнім середовищем, прийняття рішень у реальному часі та автономне виконання визначених завдань.

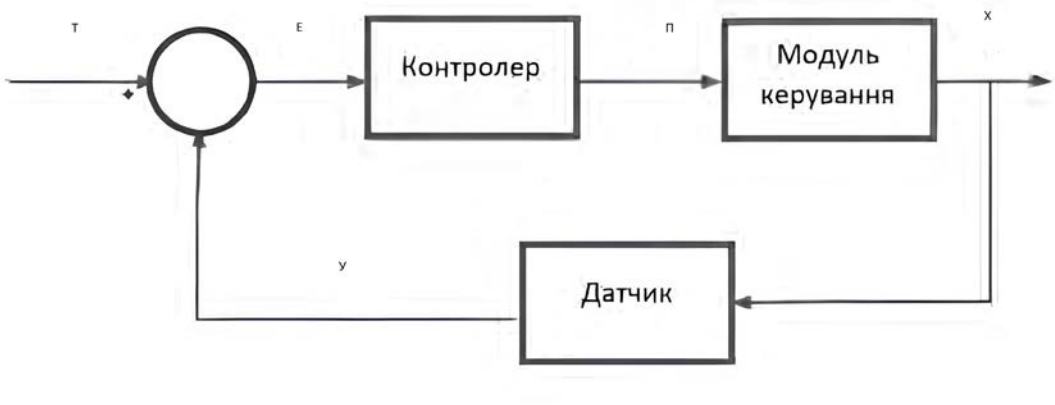


Рисунок 2.2 – Схема управління

Датчики зазвичай поділяють на дві основні категорії: внутрішні та зовнішні. Внутрішні забезпечують отримання даних про стан самого робота, тоді як зовнішні датчики збирають відомості про навколишнє середовище. Існує декілька типів датчиків, що наведені в Табл. 2.1.

Таблиця 2.1 – Основні типи датчиків та їх характеристики

№ п/п	Датчики	Особливості
1.	Положення	Ці компоненти забезпечують можливість визначення місцезнаходження робота в межах заданої області. Вони є критично важливими для реалізації функцій автономної навігації та побудови оптимальних маршрутів. Застосовуються в різноманітних типах перемикачів, зокрема в тактильних перемикачах бампера, кнопкових перемикачах та кінцевих вимикачах.
2.	Наближення	Для безконтактного виявлення об'єктів поблизу, що є критично важливим для уникнення перешкод та зіткнень в автономній навігації, використовуються датчики. До них належать фоторезистори та ультразвукові датчики.
4.	Зображення	Використовуються роботом для отримання візуальної інформації про навколишнє середовище. Такі датчики забезпечують виявлення об’єктів, підтримують автономну навігацію та сприяють ефективному плануванню маршруту руху.
5.	Температури	Пристрої, що дають змогу роботу вимірювати температуру навколишнього середовища.
6.	Освітленості	Пристрої, що дозволяють роботу визначати рівень освітленості навколишнього середовища. Для цього використовуються різні типи датчиків освітленості,

Вихід, який генерує роботизована система, називають її станом і позначають через x. На стан впливають попередні значення стану, вплив (сигнали), що подаються на виконавчі механізми, а також чинники фізичного середовища. Стан може описуватися положенням, швидкістю, кутовою швидкістю, силою та іншими величинами. Роботи не мають можливості точно виміряти стан x, проте здатні отримувати його оцінку за допомогою встановлених датчиків. Такі оцінки позначаються через y. Завдання інженера-робототехніка полягає у виборі достатньої кількості датчиків або в належному їх калібруванні, щоб забезпечити співвідношення y ~ x.

2.2 Напрями удосконалення системи управління біоморфними роботами в умовах Індустрії 4.0

У межах індустрії 4.0 виникає четверта промислова революція, яка характеризує перехід від використання окремих машин із комп'ютерним управлінням до розумного виробництва де матеріали, машини,та персонал поєднані цифровими технологіями й здатні динамічно реагувати на зміни виробничих процесів. Виникнення цього явища стало можливим завдяки злиттю операційних та інформаційних технологій. Така інтеграція сприяє зростанню гнучкості виробництва та відкриває шлях до створення унікальних продуктів, що відповідають специфічним запитам клієнтів [26]. Технології, що складають основу Індустрії 4.0, можна поділити на дві категорії: фізичні та цифрові. Фізичні технології, в першу чергу, стосуються трансформації виробничих процесів таких як адитивне виробництво, тоді як цифрові технології охоплюють сучасні інформаційно-комунікаційні системи, моделювання та інші суміжні рішення [31, 48].

Паралельно Індустрія 4.0 змінює підходи до застосування різних систем і технологій, включно з робототехнікою. Вона охоплює нові технології, які інтегрують фізичну, цифрову та біологічну сфери, впливаючи на різноманітні дисципліни, галузі, економіки та державні структури. Штучний інтелект уже

широко присутній — від суперкомп’ютерів, безпілотників і віртуальних асистентів до 3D-друку, інтелектуальних сенсорів, носимих пристроїв та мікрочіпів, менших за піщинку [32]. У межах Індустрії 4.0 виділяють шість основних керівних принципів: оперативну сумісність; віртуалізацію; децентралізоване прийняття рішень; обробку даних у реальному часі; орієнтацію на сервіс; модульність. Відповідно, поєднання технологій для оптимізації часу, підвищення точності та мінімізації помилок є тим, що вирізняє Індустрію 4.0 серед попередніх промислових революцій.

Віртуальна реальність (ВР) відіграє значну роль в Індустрії 4.0 та відкриває нові можливості для використання у робототехнічних застосуваннях завдяки концепції віртуального виробництва. На сьогодні вона застосовується для моделювання руху машин і багатотілесних систем, використовуючи віртуальні об’єкти та подаючи інформацію у формі стандартних графіків і числових даних. Індустрія 4.0 також надає особливу увагу штучному інтелекту та великим даним. Роботизовані системи контролю здатні опрацьовувати великі масиви даних, а отже, дослідження у сфері роботизованого НРК мають включати адаптацію нових методів візуалізації та аналітичної обробки даних.

В Індустрії 4.0 іншими ключовими аспектами є системна інтеграція та хмарні обчислення. Вони забезпечують можливість підключати практично будь-які елементи виробництва — обладнання до віддалених переглядачів, системи між собою, а також між різними підприємствами та навіть глобальними мережами. Стандартизовані протоколи машинної комунікації, такі як OPC UA, забезпечують взаємодію між системами. Моделі обробки даних сенсорів мають вбудовані інтелектуальні механізми, що дозволяють адаптацію до широкого діапазону реальних умов та середовищ. Моделі штучного інтелекту можуть працювати з різноманітними платформами візуального спостереження — від камер відеонагляду й одномонокулярних камер до сучасних чотириногих роботів і дронів.

Такий підхід безпосередньо вплинув на розвиток інструментів MATLAB, орієнтованих на автоматизацію роботизованого НРК, із можливістю складного

планування руху, уникнення перешкод і реалізації зовнішньої синхронізації між роботами та пов’язаними зовнішніми системами НРК. Реалізація повного зовнішнього контролю роботизованим обладнанням дає змогу синхронізувати збирання даних НРК з відповідними координатами на всіх ділянках траєкторії та закладає потенціал для подальшого розширення функціональних можливостей, важливих для задач інспекції НРК.

Потрібне нове програмне забезпечення, яке забезпечує можливість гнучкого планування траєкторій для контролю складних криволінійних поверхонь, що часто зустрічаються у сучасному інженерному виробництві. Отже, вимоги до сучасних систем обробки даних в умовах Індустрії 4.0 формують нові виклики й завдання, а також створюють додаткові можливості для застосування датчиків у НРК. Водночас, щоб реалізувати ці можливості, необхідно перейти від традиційних підходів НРК до систем, здатних легко взаємодіяти з існуючими мережами та функціонувати як джерело даних для хмарних і периферійних обчислень. Унаслідок цього НРК стає ключовим постачальником інформації для використання великих даних, аналітичних методів та інструментів машинного навчання.

Розширення та покращення бібліотек роботів є актуальним, оскільки сучасні роботизовані інспекційні системи генерують значні обсяги даних. Тому дослідження роботизованого НРК повинні включати розробку новітніх підходів до їхнього аналізу та візуалізації. Для сучасних програм НРК відхилення від вихідних параметрів систем автоматичного управління та вимоги до точних і оглядових перевірок, які залежать від виміряних даних, потребують гнучкості у формуванні траєкторій, що виходить за межі можливостей наявного програмного забезпечення для автономного програмування роботів [52].

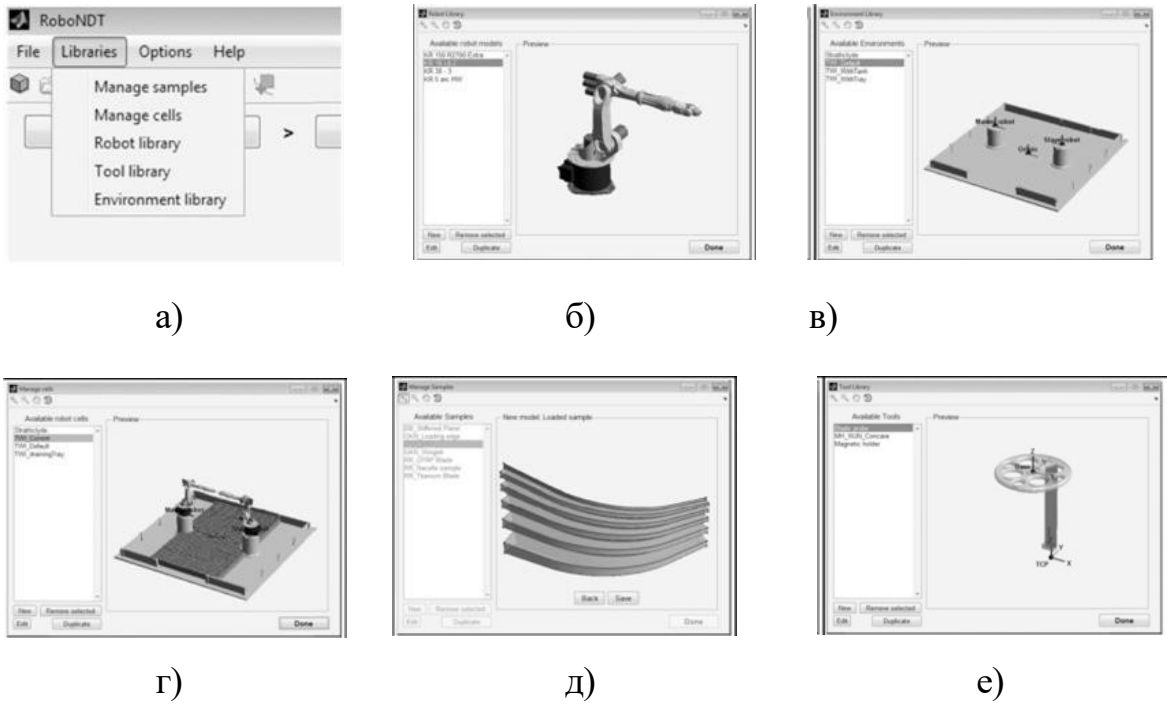


Рисунок 2.3 – Меню бібліотек (а), бібліотека роботів (б), бібліотека середовища (в), керування комірками (г), керування зразками (д) та бібліотека інструментів (е)

Удосконалення програмного забезпечення.

Ключовими завданнями є формування та коригування траєкторії руху інструмента, високошвидкісне отримання даних НРК, інтеграція результатів вимірювань поверхні та подання загальної візуалізації отриманої інформації у форматі, зручному для користувача.

Відповідно, сучасні науково-дослідні ініціативи в межах NDT 4.0 спрямовані на започаткування технологічного прориву у бік створення інтелектуальних сенсорних систем НРК. Такі системи забезпечують формування релевантних даних та інформації для цифрової пам’яті продукту на всіх етапах та на всіх рівнях його повного життєвого циклу.

2.3 Принципові підходи до конструювання біоподібного робота

Природа пропонує надзвичайно ефективні методи пересування, що розвивалися мільйони років і можуть слугувати чудовим джерелом інженерних ідей. Одним із таких прикладів є зміїна локомоція. Здатність змій успішно

пересуватися в найрізноманітніших умовах – від води до пустель і лісів – підтверджує ефективність їхнього унікального способу руху. Це наводить на думку, що змієподібні стратегії можуть бути оптимальним рішенням для пересування в складних, непередбачуваних середовищах.

Окрім результативного пересування у великій кількості середовищ, змієподібний роботизований механізм із властивостями, подібними до біологічних змій, надає низку переваг у порівнянні з традиційними роботизованими платформами. Ці переваги — малий поперечний розмір корпусу, внутрішня стабільність, висока надлишковість і значна здатність до адаптації — свідчать, що змієподібні роботи, спроєктовані на основі біологічних принципів, становлять перспективний напрям наукових досліджень щодо створення роботизованої системи, здатної працювати в широкому спектрі умов. Більше того, рух змії є відносно енергоефективним типом локомоції, що потребує менших витрат енергії порівняно з іншими біологічними формами, які мають схожі габарити, масу та швидкість. Найхарактернішими властивостями роботів-змій є їх здатність до адаптації, універсальність, масштабованість та надійність.

Отже, біоподібний робот-змія (БПРЗ) було обрано для дослідження, враховуючи такі показники:функціональні можливості (експлуатація як на суші, так і у воді); відмінна прохідність;модульність;герметичність і надійний захист від проникнення пилу чи рідин; стійкість до механічних впливів.

Це універсальні механізми, що складаються з багатьох ланок та з’єднань, які забезпечують гнучкий рух, імітуючи манеру пересування реальної змії, та на відміну від колісних, гусеничних і крокуючих систем, гарантують високу стійкість і зазвичай мають більший запас міцності, реалізуючи різноманітні способи пересування.

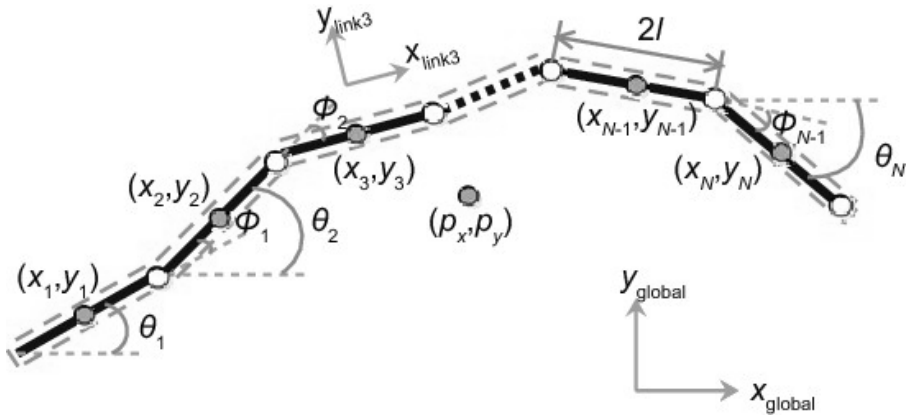


Рисунок 2.4– Схема скелета БПРЗ [21]

Ще однією перевагою БПРЗ є їхня модульна будова та висока надмірність. Існує багато ситуацій, коли робот повинен функціонувати у віддалених або важкодоступних умовах, що робить ремонт пошкодженого робота вкрай витратним. Водночас конструкція модульного БПРЗ складається з низки ідентичних модулів, завдяки чому його виготовлення та відновлення є відносно малозатратними.

Звісно, до недоліків передусім належить обмежена вантажопідйомність та складність системи керування, обумовлена великою кількістю модулів. Механізм БПРЗ зазвичай містить багато послідовно з'єднаних модулів, здатних згинатися в одній або кількох площинах. Велика кількість ступенів свободи ускладнює керування БПРЗ, однак забезпечує потенційну здатність до пересування у складних умовах та дозволяє перевершувати мобільність колісних, гусеничних та крокуючих роботів. БПРЗ характеризуються стійкістю руху та високою адаптивністю порівняно з класичними промисловими роботами. Останніми роками інтерес до БПРЗ зростає завдяки широким можливостям їх застосування [47].

Найшвидшим і найпоширенішим змієподібним способом руху, який застосовують біологічні змії, є латеральна/бічна хвилеподібність. За цієї схеми поступальний рух досягається завдяки поширенню хвиль від головної частини до хвоста, використовуючи нерівності поверхні та часткове заглиблення тіла в ґрунт. Саме цей тип руху найчастіше реалізується у БПРЗ. Змії використовують

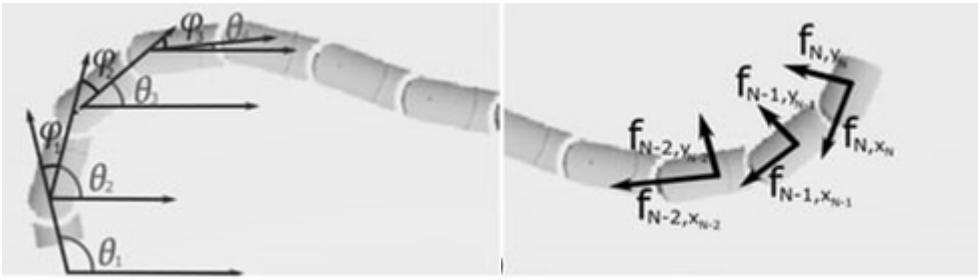
нерівності рельєфу як точки опори, що дозволяє просуватися вперед за допомогою бічних хвиль. Також встановлено, що ключовою умовою для виникнення бічної хвилястості у природних змій є наявність анізотропних сил тертя, які дають змогу рухатися вперед навіть на відносно гладких поверхнях.

Другий тип руху — це «бічне звивання», яке здебільшого реалізується біологічними зміями, що пересуваються однорідними поверхнями. Частина тіла змії залишається майже нерухомою, тоді як інша піднімається і переміщується вперед, створюючи характерну петлеподібну траєкторію. Це пересування зазвичай спостерігається у змій, що рухаються м’якими та слизькими поверхнями, наприклад піском пустель. У цьому способі змія використовує дві окремі ділянки тіла як точки статичного контакту, щоб підняти та зігнути інші сегменти.

Ще один тип пересування — прямолінійна (педальна) локомоція, також відома як гусеничний рух, найбільш характерний для масивних змій. Попередні дослідження встановили, що під час такого руху ребра виконують роль подібну до ніг, хоча змія фактично не спирається на них. За прямолінійного руху, подібно до дощових червів, змія просувається прямо, і, на відміну від інших способів, бокова взаємодія з довкіллям майже не впливає на рух. Натомість основою пересування є хвиля скорочення і розслаблення, що проходить вентральними м’язами разом із рухом окремих піднятих частин тіла.

Рух робота зумовлений зміною його внутрішньої геометрії, аналогічно до руху природних змій. Суглоби, які з’єднують сегменти робота, задаються набором кутів, що визначають його форму. Хвилеподібний характер пересування дозволяє формувати простіші математичні моделі, які описують основні властивості руху БПРЗ і добре підходять для аналізу та проєктування систем керування.

Анізотропна сила тертя сприяє руху БПРЗ, забезпечуючи нижчі коефіцієнти тертя у поздовжньому напрямку суглобів порівняно з поперечним. Така різниця коефіцієнтів дозволяє суглобам вільно ковзати вперед.



Кути з'єднання і тяга сили тертя в нормальному і тангенціальному напрямку.

Рисунок 2.5 – Кут з’єднання та сили тертя БПРЗ [5]

БПРЗ мають володіти можливістю перемикання між різними режимами руху, щоб адаптуватися до середовища та долати складні умови. Для реалізації імітації зміїного пересування необхідно застосовувати режими локомоції, що ґрунтуються на серпентиноїдній кривій Хіросе та її тривимірних модифікаціях [50]. Ці криві можуть бути подані як синусоїдальні хвилі (Рис. 3.3), що поширюються вздовж тіла, де одна відповідає за горизонтальний рух, а інша — за можливий вертикальний рух, як наведено нижче (2.1):

$$\alpha(n,t) = \begin{cases} A_x \sin(\omega_x t + n\,\delta_x), & n \text{ — odd}, \\ A_y \sin(\omega_y t + n\,\delta_y + \phi), & n \text{ — even}. \end{cases} \tag{2.1}$$

де n – число двигунів; Ax *і* Ay – амплітуди хвиль; δx і δy – просторова частота; ωx і ω^y – тимчасова частота;

φ різниця фаз між синусоїдальними.

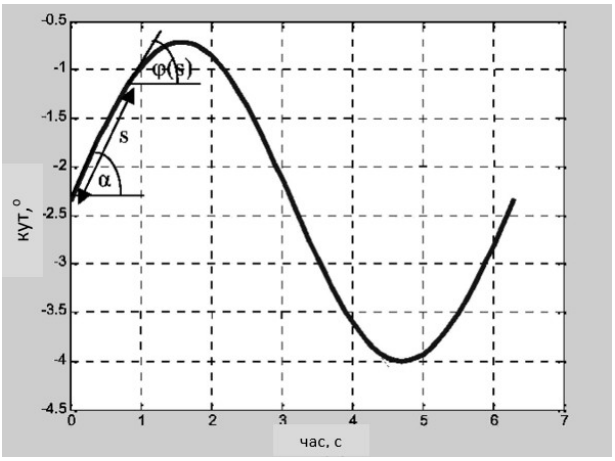


Рисунок 2.6 – Синусопідальний кривий рух БПРЗ

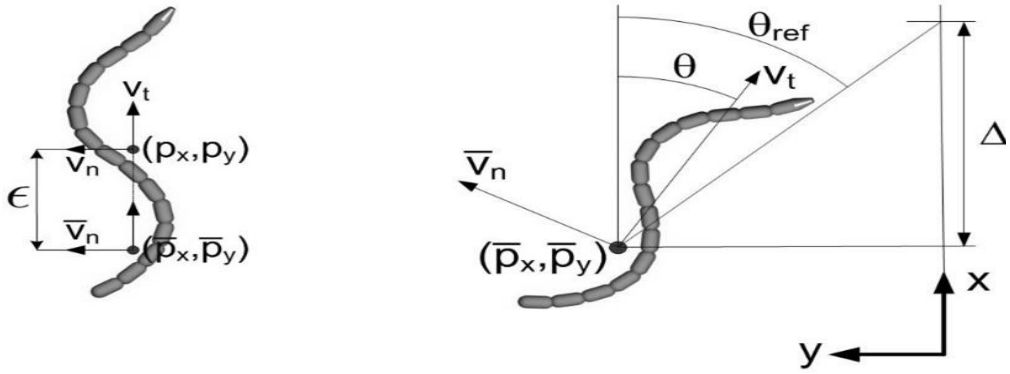
Для БПРЗ, що переміщуються по суші, запропоновано закон керування наведенням за лінією прямої видимості (line of sight – LOS), який забезпечує експоненційну стабілізацію необхідної прямолінійної траєкторії за умови заданого параметра відстані випередження. Для БПРЗ, що функціонують у водному середовищі, закон керування повинен компенсувати течії невідомого напрямку та величини. З цією метою запропоновано інтегральний закон керування прямою видимістю (ILOS), який демонструє експоненційну стабілізацію потрібної прямолінійної траєкторії при заданих умовах щодо дистанції випередження та інтегральних коефіцієнтів підсилення [44]. Для окремих задач також необхідне регулювання поступальної швидкості руху робота. Замість використання налаштувань параметрів схеми ходи на основі зв’язку між ними та швидкістю, що відповідає керуванню швидкістю в розімкнутому циклі, доцільно включати зворотний зв’язок швидкості в закон керування, розв’язуючи задачу маневрового керування. Закони маневрового керування, побудовані на основі біологічно натхненних віртуальних голономних обмежень, пропонуються для БПРЗ, що рухаються як по суші, так і під водою.

Таким чином, блок керування БПРЗ складається з двох ключових частин.

1. контролера моделі ходи, який забезпечує рух БПРЗ уперед відповідно до бічної хвилястості моделі ходи;

2. контролера курсу, який спрямовує БПРЗ на потрібний маршрут, а потім веде його уздовж нього.

Контролер моделі ходи. При цьому бічна хвилястість реалізується шляхом керування суглобами БПРЗ згідно із синусоїдальним опорним сигналом. Це означає, що координати суглобів експоненційно наближаються до опорних координат.



а) перетворення координат БПРЗ

б) закон наведення по лінії прямої видимості (LOS)

Рисунок 2.7– Схема руху БПРЗ з врахуванням LOS [51]

Контролер курсу. Для спрямування БПРЗ на потрібний прямий шлях, використовуємо закон прямої видимості (LOS).

Контроль пересування БПРЗ та синхронне планування ходи є складним завданням через їхню багатозв’язану тілесну структуру. У цьому випадку потрібні одночасні рухи, при цьому послідовність ходи не повинна залежати від зовнішніх перешкод. Одним із методів контролю, який можна застосувати до біоміметичних систем, є CPG (Central Pattern Generators). Контроль пересування на основі CPG має такі переваги: вихідні сигнали стабільні та демонструють граничні цикли; вони здатні керувати рухом робота без використання складної динамічної моделі; замкнуту структуру сенсорного зворотного зв’язку можна легко інтегрувати.

CPG присутні у багатьох хребетних і можуть моделюватися як контролер пересування. Крім того, такі моделі не вимагають динамічної моделі системи, на відміну від традиційних контролерів. Імітуючи спинний мозок, контролер пересування на основі CPG забезпечує ритмічні та стабільні виходи. Простою та ефективною моделлю CPG є ACPO (Amplitude-Controlled Phase Oscillator). ACPO, що базується на осциляторах Kuramoto (вперше запропонована Йошікі Курамото і є математичною моделлю для опису синхронізації), здатна виробляти ритмічні виходи незалежно від зовнішніх перешкод завдяки своїм функціям та поведінці граничного циклу [52]. Цей тип осцилятора визначається наступним чином:

$$\dot{\emptyset}_i = \omega_i + \sum_j^N (\mu_{ij} r_j \sin(\emptyset_j - \emptyset_i - \varphi_{ij})) \quad (2.1),$$

$$\ddot{r}_i = a_r \left(\frac{a_r}{4} (R_i - r_i) - \dot{r}_i \right) \quad (2.2),$$

$$\ddot{x}_i = a_x \left(\frac{a_x}{4} (X_i - x_i) - \dot{x}_i \right) \quad (2.3),$$

$$\theta_i = x_i + r_i \cos(\emptyset_i) \quad (2.4),$$

Де $e\theta_i$, r'_i, x_i знзнаходяться змінні стану, які визначають фазу, амплітуду та зміщення коливань, відповідно. R_i, X_i *і* w_i є бажаною амплітудою, зміщенням, і частота відповідно. N кількість осциляторів з *i*-м і *j*-м осциляторами. a_r *і* a_x є позитивними константи швидкості конвергенції. Виходи генератора виражається як . параметри і є зв'язком ваги та різниці фаз між *i*-м та *j*-м осциляторами. також визначає швидкість корекції *j*-го зв'язку. Спроектована модель побудована як двонаправлена мережа ланцюгового типу. Звідси:

$$\theta_i^{\infty}(t) = X_i + R_i \cos(w_i t + \varphi_0) \quad (2.5),$$

Тут φ_0 початкова фаза осцилятора. Для будь-яких початкових умов виходи збігаються до заданих параметрів керування, при цьому на виходи не впливають зовнішні збурення. Таким чином, амплітуду та фазовий зсув коливань можна легко регулювати, а вихідні сигнали пов’язаних осциляторів формують стаціонарні та стабільні моделі змієподібного руху

Сенсорний механізм. При проєктуванні БПРЗ слід враховувати такі критерії, як функціональність, простота конструкції та економічна доцільність. Важливе значення має вибір сенсорного механізму, конструювання модулів та розробка системи керування, що забезпечує ефективність пересування робота по нерівній поверхні з наявністю перешкод.

Для створення ефективної системи керування БПРЗ у реальних умовах датчики є необхідним компонентом. На відміну від людської ходьби, де доступна візуальна зворотна інформація від очей, біологічна змія збирає дані про навколишнє середовище через контактні сили. Оскільки саме ці сили між тілом робота та оточенням генерують рух, аналогічний сенсорний механізм підходить і для БПРЗ. Такі дані можна отримати за допомогою датчиків сили на поверхні

ланок робота або датчиків крутного моменту на кожному суглобі.

Стратегії керування БПРЗ. На відміну від багатьох простих механічних систем, де ефективно застосовуються класичні методи керування на основі моделі, ці підходи непридатні для реального часу у БПРЗ. Це пояснюється тим, що БПРЗ без пасивних коліс та анізотропного тертя, подібного до природного аналога, належать до категорії роботизованих систем із недостатнім приводом, для яких стандартні методи керування на основі моделі є надмірно складними та обчислювально витратними.

Тому більшість стратегій керування для безколісних БПРЗ базуються на генерації біологічно натхненних періодичних команд для кутів суглобів (шаблонів ходи), що дозволяє досягти необхідного типу руху шляхом корекції відповідних параметрів.

1. Стратегії керування в структурованому середовищі. Для управління БПРЗ у структурованому середовищі пропонуються моделі ходи з різними властивостями, такими як міцність. Більшість цих моделей мають подібну структуру. Наприклад, на плоских поверхнях значна частина контролерів базується на синусоїдальних паттернах ходи, які генерують кути суглобів наступним чином [17]:

$$q\ i = A\ sin(\omega t + \phi(i-1)) \qquad (2.6),$$

де $q\ i$; i = 1,2, … , N – кути з’єднання, N – кількість зв’язків, ω – часова частота, ϕ – просторова частота, A – амплітуда синусоїдальної хвилі.

Замість застосування нелінійних осциляторів, подібних до (2.6), використано біоінспірований осцилятор – центральний генератор шаблонів (CPG). Ця система складається з мережі взаємопов’язаних «нейронів», здатних формувати ритмічні моделі руху без потреби у зовнішніх сигналах. Використання таких осциляторів дозволяє мережі CPG генерувати ритмічну опору для кожного суглоба. Таким чином, за допомогою послідовних ритмічних сигналів із певним фазовим зсувом можна створювати опорні кути суглобів для реалізації змієподібної локомоції.

Стратегії управління в неструктурованому середовищі**.** БПРЗ повинні ефективно пересуватися в нерівних умовах, відчувати оточення та відповідно

коригувати форму тіла та рухи. Загалом виділяють два типи нерівностей, з якими стикаються БПРЗ у реальному середовищі:

1. бічні нерівності, коли перешкоди розташовані збоку від робота;

2. вертикальні нерівності, що виникають під час руху по нерівних поверхнях.

Пересування в середовищах із бічними нерівностями. Для першого типу нерівностей БПРЗ можуть уникати перешкод або використовувати їх як точки штовхання. Використовуючи інформацію від тактильних сенсорів на боці суглобів, пропонується алгоритм пересування з урахуванням перешкод, де кожен сегмент тіла точно слідує за модулем голови. Коли головний модуль контактує з новою точкою штовхання, алгоритм підгонки кривої з подальшою дискретизацією обчислює опорні кути для інших суглобів до наступного контакту.

Інший підхід до адаптивного пересування у таких умовах – керування на основі форми. У стандартній роботі траєкторії для суглобів БПРЗ формуються на основі параметризованих функцій ходи. Фактичну форму робота можна приблизно описати за оцінкою параметрів ходи та використати для досягнення адаптивного пересування.

Пересування по поверхнях із нерівностями. Невеликі нерівності на поверхні можна моделювати через зміну коефіцієнта тертя. Для створення адаптивної схеми пересування застосовується зворотний зв’язок за силами тертя та кутом нахилу тіла для корекції параметрів ходи. Подібні методи застосовуються разом із CPG, де налаштування параметрів осцилятора змінює частоту, амплітуду, фазові зсуви або форму вихідної хвилі для досягнення бажаного руху.

Схема керування роботом наведена в Додатку В: вхідними параметрами є бажана амплітуда, зсув та фазові співвідношення для кожного суглоба. Кожен осцилятор відповідає окремій ланці робота. Після задання параметрів мережа CPG генерує синусоїдальні кути, які через ШІМ-сигнали подаються на серводвигуни. Проектована мережа включає чотири регуляторні системи:

регулятор амплітуди, частоти та фази, регулятор зсуву та вихідну систему.

Відчуття є ключовим для адаптації в робототехніці, оскільки інтелектуальний контролер збирає дані з навколишнього середовища та застосовує їх для адаптивного керування поведінкою системи. Розробка сенсорного механізму для БПРЗ є складнішою, ніж для гуманоїдних роботів, через труднощі розміщення бортових камер під час постійного руху модулів. Тому більшість БПРЗ для пересування в неструктурованому середовищі оснащені датчиками крутного моменту, сили або тиску для визначення взаємодії з оточенням. Деякі роботи використовують бічні контактні перемикачі для визначення бокового контакту, інші – датчики тиску під модулями, а сучасні моделі застосовують FSR, тензометричні датчики та спеціалізовані системи вимірювання крутного моменту на основі кулачкового механізму для підвищення чутливості.

Математична модель БПРЗ, як правило, залежить від його конструкційних особливостей. Для класифікації різних конструкцій БПРЗ визначають основні характеристики, такі як тип з’єднань, кількість ступенів свободи, наявність або відсутність пасивних коліс.

Більшість БПРЗ складаються з ланок, з’єднаних обертовими шарнірами з однією або двома ступенями свободи. У деяких роботів ланки є телескопічними (призматичні з’єднання). Щоб забезпечити необхідні властивості тертя для бокової хвилястості, деякі БПРЗ обладнані пасивними колесами. При їх використанні динаміку взаємодії БПРЗ з поверхнею часто спрощують або ігнорують.

Математичне моделювання різних БПРЗ зазвичай розділяють на кінематику та динаміку. Кінематика описує геометричні аспекти руху.

Для безколісних БПРЗ тертя між роботом та поверхнею значною мірою впливає на рух. Тому для таких БПРЗ динаміка має бути змодельована для відповідних типів пересування, наприклад бічної хвилястості.

Переходи між ковзанням (за законом тертя Кулона) і контактами із землею моделюються як миттєві події. Це дозволяє створити точну модель просторового

кулонівського тертя, де враховуються як напрямки сили тертя, так і справжня фаза прилипання. Для безколісних роботів-змій важливо точно описати контакт з поверхнею як під час застрягання, так і при ковзанні. Ця особливість відрізняє безколісних БПРЗ від, наприклад, ножових механізмів, які зазвичай намагаються «прилипнути» до землі, а не ковзати по ній.

Динаміка БПРЗ описується через рівняння руху, що включає рівняння Ньютона–Ейлера для безударної частини та рівняння удару. Вибір координат забезпечує зручне представлення рівнянь системи. Узгоджені закони сили дозволяють формулювати кожен конститутивний закон одним рівнянням і уникати явного перемикання між рівняннями (наприклад, при зіткненні з поверхнею), навіть у гібридній системі. Це ефективно, оскільки сегменти робота часто контактують з землею під час, наприклад, поворотів або бічного руху.

Більшість математичних моделей, що описують динаміку БПРЗ, обмежувалися планарним (2D) рухом. Тривимірні (3D) моделі БПРЗ були розроблені відносно недавно. 3D-моделі спрощують тестування та розробку 3D-шаблонів змієподібного руху, таких як синус-ліфтинг і боковий рух. Для моделювання 15-ланкового БПРЗ замість явних динамічних рівнянь використовувався фізичний механізм Open Dynamics Engine (ODE) [15]. Таке програмне забезпечення дозволяє швидко змінювати геометрію робота та скорочує час підготовки робочої моделі [15].

Моделювання БПРЗ здійснюється за допомогою Matlab-Simulink із застосуванням інструменту SimScape, що дозволяє ефективно та швидко проводити багатотілесне моделювання та формулювати й вирішувати рівняння руху будь-якої механічної системи.

При розробці змієподібного робота перевага віддавалась модульній конструкції, оскільки така архітектура спрощує заміну або додавання модулів, тим самим змінюючи ступені свободи. Поточна конфігурація складається з модулів для кожної ланки тіла, а також окремого головного і хвостового модулів.

Модель БПРЗ включає n ланок, з’єднаних n − 1 карданними шарнірами з двома ступенями свободи (тобто обертальні шарніри). Нехай u∈ R 6n − в

– вектор, що містить поступальні та обертальні швидкості всіх ланок робота-змійки. Диференціальні міри du та dt зараз можуть вільно трактуватися як «можлива диференціальна зміна» u і часу t. Використання диференціальних мір дозволяє моделювати миттєві зміни швидкості при контакті БПРЗ із поверхнею. Системні рівняння для БПРЗ тепер можна записати як:

$$Mdu - hdt - dR = \tau Cdt \quad (2.7),$$

яке називається рівністю мір, де M ∈ R 6n ×6 n – матриця мас, h ∈ R 6n складається з плавних сил, τC ∈ R6n містить усі моменти шарнірного приводу, а dR ∈ R6n враховує нормальні контактні сили/імпульси від землі, сили/імпульси кулонівського тертя та двосторонні сили/імпульси обмеження в суглобах.

Модель БПРЗ складається з n циліндричних ланок, які з'єднані n −1 карданними шарнірами, кожне з яких має два ступені свободи. Відстань Li між двома сусідніми карданними шарнірами дорівнює довжині ланки i , а радіус кожної ланки LSCi . Кожна ланка моделюється як циліндр довжиною 2LGSi з двома сферами радіусом LSC i, прикріпленими до кінців ланки.

Положення та орієнтація ланки i описуються немінімальними абсолютними координатами:

$$q1 = \left[\frac{IrGi}{pi}\right] \in R^7 \quad (2.8),$$

Де $I_{rGi} = [xi\ yi\ zi]^T \in R^3$ є положенням центру ваги ланки i та вектор

pi= [eio et/i]T, де et/I = [ei1 ei2 ei3], містить чотири параметри Ейлера, які використовуються для опису обертання.

Параметри Ейлера утворюють одиничний кватерніонний вектор з обмеженням pT/ipi = 1 . Координати є немінімальними, оскільки кожна ланка описується 6 координатами, і абсолютними, оскільки позиція та орієнтація ланки i задані безпосередньо відносно I . Швидкість ланки i визначається як

$$ui = \left[\frac{IvGi}{Bi\varpi IBi}\right] \in R^6 \quad (2.9),$$

де I v G i - швидкість поступального руху CG ланки i , яка дорівнює

Iv Gi = IrG i, коли вона існує (тобто для руху без удару);

BiωIBi є кутовою швидкістю системи Bi відносно кадру I , поданого в кадрі B i .

Перетворення I r = R I B r можна виконати за допомогою матриці обертання R B i = H i H i де

$$\mathrm{X}^{i} = \begin{bmatrix} -\boldsymbol{e}_i & \tilde{\boldsymbol{e}}_i + e_{i_0}\mathbf{I} \end{bmatrix}, \; \bar{\mathbf{H}}_i = \begin{bmatrix} -\boldsymbol{e}_i & -\tilde{\boldsymbol{e}}_i + e_{i_0}\mathbf{I} \end{bmatrix} \tag{2.10}$$

а верхній індекс ˜ позначає кососиметричну форму вектора, тобто

$$\mathrm{e}^{\tilde{}} = \begin{bmatrix} 0 & -e_{i_3} & e_{i_2} \\ e_{i_3} & 0 & -e_{i_1} \\ -e_{i_2} & e_{i_1} & 0 \end{bmatrix} \tag{2.11}$$

Похідна за часом матриці обертання знайдена

$$\dot{\mathbf{R}}^{I}_{B_i} = \mathbf{R}^{I}_{B_i}\, {}_{B_i}\tilde{\boldsymbol{\omega}}_{IB_i} = {}_{I}\tilde{\boldsymbol{\omega}}_{IB_i}\, \mathbf{R}^{I}_{B_i} \tag{2.12}$$

Координати $= \begin{bmatrix} \boldsymbol{q}_1^{\mathrm{T}} & \cdots & \boldsymbol{q}_n^{\mathrm{T}} \end{bmatrix}$ (положення та орієнтація) і швидкості T усі ланки зібрані у вектори q і

$$\mathrm{u} = \begin{bmatrix} \boldsymbol{u}_1^{\mathrm{T}} & \cdots & \boldsymbol{u}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2.13}$$

Кожен карданний шарнір має 2 DOF, які керуються двома шарнірними приводами. Приводи моделюються як керовані крутні моменти, прикладені навколо осей обертання для шарніра.

B i +1 швидкість обертання навколо ex і додатний керуючий крутний момент τ h i для надання додатної швидкості обертання навколо e B y i , обидва відносно ланки i . Загальний крутний момент τ C i ∈ R 3 , прикладений до ланки i , становить

$$\boldsymbol{\tau}_{C_i} = \begin{bmatrix} 0 \\ \tau_{h_i} \\ 0 \end{bmatrix} - \mathbf{R}^{B_i}_{B_{i-1}} \begin{bmatrix} 0 \\ \tau_{h_{(i-1)}} \\ 0 \end{bmatrix} + \mathbf{R}^{B_i}_{B_{i+1}} \begin{bmatrix} \tau_{v_i} \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} \tau_{v_{(i-1)}} \\ 0 \\ 0 \end{bmatrix} \tag{2.14}$$

для i = 1 ,...,n , де матриця відносного обертання

$$\mathrm{R}^{B_i}_{B_{i+1}} = \left(\mathbf{R}^{I}_{B_i}\right)^{\mathrm{T}} \mathbf{R}^{I}_{B_{i+1}}, \tag{2.15}$$

i τ h 0 = τ v 0 = τ h n = τ v n = 0 .

Вектор обертових моментів, прикладених до всіх ланок τ C ∈ R 6 n дорівнює

$$\boldsymbol{\tau}_C = \begin{bmatrix} \mathbf{0}_{1\times 3} & \boldsymbol{\tau}_{C_1}^{\mathrm{T}} & \mathbf{0}_{1\times 3} & \boldsymbol{\tau}_{C_2}^{\mathrm{T}} & \cdots & \mathbf{0}_{1\times 3} & \boldsymbol{\tau}_{C_n}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2.16}$$

Кожне карданне з’єднання 2 DOF приводилося в дію двома двигунами постійного струму. Опорні кути з’єднання БПРЗ надсилалися з ПК через CAN-

шину на мікроконтролери з частотою 10 Гц. Положення центру середньої ланки відстежується за допомогою системи захоплення руху Vicon MX з 4 камерами (MX3) разом із Matlab Simulink.

2.4 Динамічне моделювання модульних БПРЗ

Функції $x(s)$, $y(s)$, $z(s)$ є диференційованими функціями, а s є параметром кривої, який зазвичай вибирається як довжина дуги кривої. За визначенням, довжину дуги кривої від початкової точки s 0 до довільної точки

s можна обчислити як:

$$L(s) = \int|\alpha\,'(\sigma)|\,d\sigma, \qquad (2.17),$$

де $|\alpha\,'$

$$(s)| = \sqrt{\left(\frac{dx}{ds}\right)^2 + \left(\frac{dy}{ds}\right)^2 + \left(\frac{dz}{ds}\right)^2} \qquad (2.18),$$

Вибір параметра кривої s як дуги довжина:

$$L(s) = s - s\,0 \qquad (2.19),$$

в результаті чого:

$$|\alpha\,'(s)| = 1 \qquad (2.20),$$

Використовуючи цей метод параметризації, положення кожної точки на тілі змії відносно початку рамки, прикріпленої до хвоста змії, визначеної в декартовій координаті, можна отримати таким чином:

$$x{\rightarrow}(s,\,t) = \int u{\rightarrow}{\rightarrow}(\sigma,\,t)\,d\sigma \qquad (2.21),$$

де t позначає час, $u{\rightarrow}{\rightarrow}(\sigma,\,t) = \partial\alpha\,(\sigma,t)$ і $s \in I = (0,\,1)$, нормалізована довжина $\partial\sigma$ змійка, яка не змінюватиметься з часом, оскільки модульні змійки вважаються нерозширюваними.

Незважаючи на те, що структура моделювання на основі кривих здебільшого застосовується для гіпернадлишкових та/або континуальних роботів, її також можна адаптувати для моделювання модульних БПРЗ. Для модульних БПРЗ із скінченною кількістю модулів ці методи стають обчислювально складними та теоретично громіздкими, не приносячи суттєвих переваг. Тому для моделювання модульних БПРЗ більш доцільною є

сегментована модель.

Висновки до розділу 2

Дослідження в галузі роботів-змій тривають уже багато років, однак і досі існує чимало невирішених питань, пов’язаних із їхнім моделюванням та системами керування. Перш ніж такі роботи зможуть ефективно та гнучко пересуватися у тісних або складних умовах, необхідно подолати низку технічних викликів. Для роботів-змій, що функціонують у обмежених просторах, визначальне значення мають модульна побудова та відповідні алгоритми управління.

Математичний опис мобільного робота зазвичай охоплює дві основні складові: кінематику та динаміку. Кінематична модель дає змогу визначити поточні координати робота — x, y — а також його орієнтацію θ, яка характеризує напрямок руху відносно вибраної системи координат. Вхідними параметрами для цієї частини моделі є кутові швидкості ωL та ωR, що формуються на основі динамічної моделі. Виходячи з рівнянь динамічної частини, можна отримати повний опис стану мобільного робота.

Математична модель рухомого робота, яка включає кінематичну і динамічну складові. Кінематична модель рухомого робота використовується для визначення координат теперішнього положення x, y та кута θ, що відображає обертання рухомого робота відносно вибраної системи координат. Вхідними параметрами в кінематичну модель є кутові швидкості ω_L, ω_R, які створюються динамічною моделлю. З рівнянь стану динамічної моделі рухомого робота можна вивести опис стану

Крім того, для підвищення результативності хвильового пересування педалі БПРЗ запропоновано удосконалену стратегію регулювання жорсткості. У межах цього підходу сигнал про положення передається вздовж усього тіла робота, що дає йому змогу долати численні перешкоди, не перевертаючись на бік.

Математична модель рухомого робота містить кінематичну та динамічну складові. Кінематична модель слугує для визначення поточних координат положення x , y та кута θ , що відображає орієнтацію робота відносно обраної системи координат. Вхідними даними для кінематичної моделі є кутові швидкості ω L , ω R , які формує динамічна модель. Опис стану може бути отриманий з рівнянь стану динамічної моделі рухомого робота.

Результати моделювання свідчать про доцільність моделі, що базується на законі сухого тертя Кулона, адже БПРЗ (біоробот-прохідник завдяки зсуву) рухається вперед. БПРЗ складається з трьох основних частин: нерухомої "голови", що містить інфрачервоні датчики, багатоланкового рухового механізму корпусу та пасивного "хвоста", приєднаного до крайньої секції. Робот спроєктовано як послідовний механізм, а всі його частини виконані за модульним принципом. Компоненти робота змодельовані у середовищі SolidWorks та виготовлені за допомогою технології 3D-друку.

Системи керування біоподібним роботом відіграють ключову роль у роботі робота та у якісному досягненні запланованого результату, а також мають бути пристосовані до реальних умов експлуатації. Основними елементами такої системи є датчик, контролер і модуль керування. Ступінь автономності роботів може бути різним. Частину роботів оператори контролюють дистанційно. Інші здатні функціонувати без участі людини. На даний момент Індустрія 4.0 вносить власні зміни в методи застосування різноманітних систем і технологій, зокрема в галузі робототехніки. РНРК повинен еволюціонувати разом із впровадженням нових інструментів, серед яких автономні робототехнічні системи, симуляція цифрових двійників, Інтернет речей, кібербезпека та хмарні обчислення. Це додатково підвищує ефективність проведення НРК.

РОЗДІЛ 3
ПРОЄКТУВАННЯ КОНСТРУКЦІЇ ТА СИСТЕМИ КЕРУВАННЯ РОБОТА

3.1 Створення моделі біоподібного робота

Технологія 3D-друку за останній час забезпечила надшвидкий розвиток, особливо у сфері розробок та досліджень. Адитивне виробництво дозволяє швидко верифікувати моделі та з мінімальними витратами перевіряти їхню працездатність. У робототехніці 3D-друк використовується для контролю правильності роботи механізмів, з'єднань, допусків і посадок. Однак слід враховувати обмеження адитивних технологій: хоча деякі елементи можна виготовити швидко та дешево, їхня механічна міцність значно нижча, ніж у металевих або композитних конструкцій.

Механічна структура БПРЗ містить в собі модулі голови, тулуба та хвоста. Будова елементів наведена на Рис.3.5 та в Додатках В–Д.

Система управління включає три основні компоненти: головну систему управління, підлеглі системи управління та систему моніторингу.

У головному модулі БПРЗ інтегрована система, яка відповідає за автоматичне виявлення зовнішнього середовища та керує підпорядкованими системами управління через відповідні команди.

Системи управління, розподілені по тулубу та хвосту, забезпечують локальну регуляцію та контроль ходи робота.

Система моніторингу реалізована через додаток на мобільному пристрої, що виконує функції спостереження та відображення вибіркової інформації.

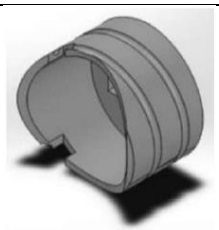
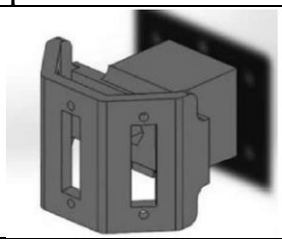
	
а) компонент двигунат	б) компонент зовнішньої оболонки
	
в) з’єднання голови/хвоста та шарніра	г) частини голови та хвоста
	
д) пакет датчиків	

Рисунок 3.1 – Основні компоненти корпусу

Живильна мережа складається з підсистеми для зарядки літієвої батареї та вузла стабілізації напруги, який забезпечує усі енергетичні потреби змієподібного робота.

Кожен шарнір обладнано власним силовим блоком

Для доведення до досконалості топології голови, корпусу та хвоста зчленованої конструкції змієподібної машини задіяно програмне забезпечення Solidworks.

Моделювання шарнірних вузлів проводиться з метою гарантування необхідного простору для розміщення літієвих акумуляторів, друкованих плат управління та рухових механізмів.

Центральний блок управління містить у собі модуль відеокамери, дальномірний лазерний пристрій, сенсор температури та вологості, датчик атмосферного тиску, а також тепловізійний сенсор. Камерний та тепловізійний

прилади збирають відомості про довкілля та візуальну інформацію, що імітує людське сприйняття, і виводять їх у вигляді зображень.

Підлеглі системи контролю взаємодіють із головним центром для здійснення нагляду та регулювання рухів робота шляхом бездротового обміну даними. Вони формують імпульсно-широтні сигнали, які подаються на відповідні приводи.

Система моніторингу оснащена модулями бездротового зв'язку WiFi та Bluetooth для комунікації з головним хабом, який відповідає за керування та нагляд за роботом.

Реалізація системи моніторингу виконана на базі платформи Android. Інтерфейс управління поділено на дві основні секції: перша займається прийомом відеопотоку з камери у режимі реального часу, друга — тепловізійним спостереженням, контролем рухів та зчитуванням показників сенсорів. Контроль за переміщенням охоплює п'ять різних режимів локомоції: змієподібний рух, повзання, бокове зміщення, перекидання та підйом по поверхні.

Для робота передбачено три варіанти стратегій керування:

1. режим, базований на типах ходи;
2. режим, заснований на серпентинних (змієподібних) функціях;
3. режим, реалізований на основі ЦПГ (Центрального Патерн-Генератора).

Робот може використовувати різні типи виконавчих механізмів, серед яких пневматичні приводи, електричні двигуни, сервоприводи чи колеса з приводом. У цьому дослідженні пропонується використання радіокерованих серводвигунів (наприклад, серводвигуна Dynamixel AX-12) (Рис.3.6).

Цей тип серводвигуна має вбудований мікроконтролер, що дозволяє послідовне підключення через інтерфейси TTL чи RS-485 зі швидкістю 1–3 Мбіт/с. Він забезпечує налаштовувану потужність обертання та крутний момент, а також високоточне регулювання положення, навантаження, рівня напруги, швидкості та температури, що значно спрощує розробку системи управління.

Сервоприводи є загальновживаними, вирізняються компактністю, високою продуктивністю та економічністю у споживанні енергії. Ці якості роблять їх

ідеальними для управління дистанційно чи радіокерованими апаратами, зокрема робототехнікою та авіамоделюванням. Також вони знаходять застосування у промислових процесах, сфері робототехніки, конвеєрному виробництві, фармацевтиці та підприємствах громадського харчування.

Рисунок 3.2– Серводвигун Dynamixel AX-12

Технічні характеристики: Вага: 52,9 г Розміри: 32 мм x 50 мм x 40 мм Передаточне число: 32:1 Робоча напруга: 9-12 В (рекомендована 11,1 В) Момент утримання: 0,2 Н*м (12 В) Струм утримання: 1,4 А

(12 D) Швидкість без навантаження: 470 об/хв (12 В) Мінімальний кут повороту: 0,29 градуса x 1,024 Кут повороту: в режимі актуатора – до 300 градусів, у режимі двигуна – без обмежень Управління: цифрове Протокол: напівдуплексний послідовний асинхронний (8N1) ID: 0...253 (за замовчуванням ID = 1) Швидкість інтерфейсу: 7843 бод/с... 1Мбод/с (за замовчуванням 1 Мб/с) Зворотній зв'язок: положення, температура, навантаження, напруга тощо. Матеріал: інженерний пластик Датчик положення: потенціометр

Управління сервоприводами здійснюється шляхом подачі електричного сигналу змінної тривалості, відомого як широтно-імпульсна модуляція (ШІМ), по контрольному дроту. Існують обмеження щодо найменшого та найбільшого імпульсу, а також частоти його повторення. Як правило, сервомотор спроможний обертатися лише на 90 градусів у чи то в один, чи то в інший бік, що сукупно дає діапазон у 180 градусів. Початковою, "нейтральною" точкою вважається таке

положення двигуна, з якого він може однаково зміститися як за, так і проти циферблатної стрілки. Саме ШІМ, який надходить на двигун, встановлює позицію його вала, і тривалість цього сигналу диктує, куди саме повернеться ротор. Сервопривід очікує отримати такий керуючий імпульс кожні 20 мілісекунд (мс). Тривалість цього імпульсу й визначає ступінь повороту двигуна. Наприклад, імпульс тривалістю 1,5 мс змусить механізм зайняти позицію на 90 градусів. Сигнал, коротший за 1,5 мс, змістить його до 0 градусів (проти руху стрілки), тоді як будь-який сигнал тривалістю понад 1,5 мс призведе до обертання сервоприводу до 180 градусів (за рухом стрілки).

Для живлення системи використовується блок, що видає 11,1 В. Для фіксації наявності перешкод до головного переднього блоку підключені інфрачервоні сенсори типу Sharp GP2Y0A21 (див. Мал. 3.7). Робочий діапазон цих датчиків становить від 10 до 80 сантиметрів.

Рисунок 3.3 – Датчик дальності інфрачервоний Sharp GP2Y0A21

Для блоку управління встановлено 32-розрядний мікропроцесор ARM Cortex M4 . Він має 256 Кб оперативної пам'яті та 2 Мб флеш-пам'яті (Рис. 3.8)

Рисунок 3.4 – Мікропроцесор ARM Cortex-M4 Мікропроцесор ARM (Advance RISC Machine) це один з ліцензованих процесорних ядер.

Створення першого чипа ARM відбулося ще у 1978 році в Кембриджському університеті, а вже у 1985 році група Acorn Group випустила перший процесор на архітектурі ARM RISC. Ці процесори знайшли своє застосування, переважно, у портативних гаджетах, таких як цифрові фотоапарати, стільникові телефони, мережеві хаби для дому, системи бездротового зв'язку та інші вбудовані рішення, завдяки їхнім перевагам, зокрема низькому енергоспоживанню та оптимальній продуктивності.

32-розрядне ядро Arm® Cortex-M4 стало першим у сімействі Cortex-M, що отримало спеціалізовані блоки для цифрової обробки сигналів (DSP), включно із вбудованим модулем для операцій з рухомою комою (FPU). Воно націлене на застосунки цифрової обробки сигналів, де потрібні ефективні та зручні засоби для керування та маніпуляції сигналами, наприклад, у сферах IoT, управління електродвигунами, контролю живлення, вбудованих аудіосистем, промислової та домашньої автоматизації, а також у медичному обладнанні та фітнес-трекерах.

Мікроконтролери на базі Cortex-M4 використовують інтегровані апаратні прискорювачі DSP для виконання математичних розрахунків, необхідних для обробки сигналів. Такий апаратний прискорювач здатний опрацьовувати будь-який аналоговий сигнал, чи то вихідний сигнал мікрофона, дані зворотного зв'язку від датчика в системі управління двигуном, чи ж інформація, отримана від сенсорів у комплексних системах.

Завдяки DSP, для виконання алгоритмів керуючого контуру потрібно менше тактів процесора, що позитивно впливає на швидкість роботи та енергоефективність програми. Фактично, коли алгоритми обробляються з використанням форматів даних Q1.15 або Float32, мікроконтролери під керуванням Cortex-M4 демонструють значно вищу продуктивність порівняно з тими, що працюють на Cortex-M3.

Керування серводвигунами здійснюється за допомогою ШІМ-сигналів. Контрольний блок реалізує послідовні та синхронні рухи відповідно до алгоритму управління, а також аналізує дані, зібрані з датчиків через входи АЦП.

Електродвигуни та сервоприводи мають відповідати низці технічних вимог, зокрема щодо габаритів, крутного моменту, робочих обертів тощо. На основі цих зовнішніх вимог проєктуються рухомі вузли компонентів.

Рисунок 3.5 – Деталізований вигляд сегмента - з’єднання частин БПРЗ

Головна і хвостова частини кріпляться до шарнірних з'єднань своїми частинами тіла. З'єднання двигунів проходять через кабельні канали, розташовані в зовнішніх оболонках

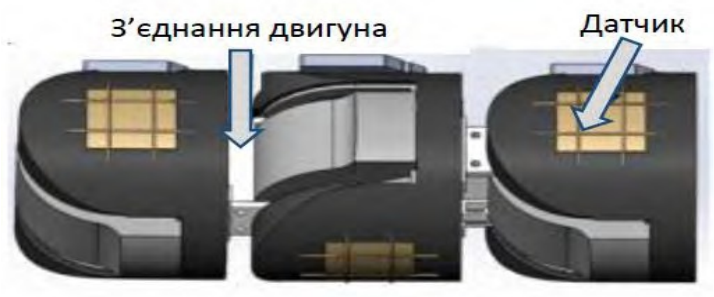


Рисунок 3.6 – З’єднання кількох сегментів тіла БПРЗ

Конструктивні елементи двигуна розроблені таким чином, щоб забезпечити його монтаж як сервоприводу, так і у вигляді звичайного валу двигуна.Комплекти сенсорів закріплені на носовій та хвостовій секціях. Ці секції розраховані на утримання трьох інфрачервоних датчиків, розміщених по вертикалі. Кожна ланка з’єднується із наступною, формуючи таким чином ланцюгову механічну систему. Орієнтація переднього сенсора задається по осі абсцис, з відхиленнями +/- 45 градусів ліворуч та праворуч відносно нього. Варто зазначити, що блок сенсорів монтується виключно на передньому вузлі. Блок управління та інша електроніка розміщені у центральній частині на фіксованій голівці. Габарити внутрішньої голівки розраховані з урахуванням габаритів цих компонентів.

Таблиця 3.1 – Робота сервоприводів у вертикальній осі

Кут	Радіус повороту [м]	Адгезія
10°	r = 1,78	добра
20°	r = 0,89	добра
30°	r = 0,6	добра

Належне електронічне проєктування відіграє вирішальну роль у взаємодії біоміметичного PVDF-сенсора. Оскільки для п'єзоелектричної плівки на низьких частотах використовується схема заміщення, що складається з послідовного конденсатора та джерела напруги, для узгодження з цим датчиком потрібне застосування високоомного вхідного каскаду.

Слід уникати використання довгих з'єднувальних дротів від датчика до інтерфейсу, бо вони спричиняють ефекти ємності витоку.

Адитивне виробництво спрощує створення прототипів. Проте процес проєктування мусить брати до уваги обмеження цього методу. Для певних компонентів, включно з бамперами, проставками та кожухами, 3D-друк є чудовим рішенням. Інші ж елементи, як-от шестерні, вали та колеса, продемонстрували гірші результати, особливо після тривалого експлуатування.

3.2 Розробка системи управління біоморфного робота

Розробка системи управління біоморфного робота

Концептуальна основа проєкту

Система управління біоморфним роботом-змією створювалась як комплексне рішення для дослідження адаптивної локомоції модульних роботів. Основна ідея полягає в розробці інтерактивного інструменту, який поєднує візуалізацію кінематики робота з інтеграцією спеціалізованих програмних модулів для оптимізації руху, машинного навчання та симуляції фізичних процесів.

Розробка почалась із визначення базової структури React-компонента. На цьому етапі було імпортовано необхідні хуки React та бібліотеку іконок lucide-react, яка забезпечує візуальне представлення різних функцій системи.

```
import React, { useState, useEffect, useRef } from 'react';
import { Camera, Thermometer, Droplets, Gauge, Sun, Activity, Video,
    Eye, Settings, Play, Pause, RotateCcw, Save, Upload, Zap, AlertTriangle,
    CheckCircle, Radio, Wifi, Battery, Cpu, HardDrive, Clock, Download, FolderOpen,
    BarChart3, Layers, Move, ArrowUp, ArrowDown, ArrowLeft, ArrowRight, Workflow, Brain,
    Code, GitBranch, Network, Database, TrendingUp, Target } from 'lucide-react';

```

Рисунок 3.7 – Архітектурний фундамент

Хуки useState та useEffect формують реактивну основу додатку, дозволяючи компоненту динамічно реагувати на зміни стану. useRef використовується для безпосереднього доступу до canvas-елемента та керування анімаційним циклом без зайвих перерендерів.

Центральним елементом системи є масив модулів, що представляють сегменти робота-змії. Кожен модуль моделюється як об'єкт з фізичними та кінематичнимивластивостями.

```
const [modules, setModules] = useState(
    Array(12).fill(0).map((_, i) => ({
        id: i,
        angle: 0,
        targetAngle: 0,
        velocity: 0,
        torque: 0,
        active: true,
        name: i === 0 ? 'Голова' : i === 11 ? 'Хвіст' : `Модуль ${i}`,
        x: 100 + i * 50,
        y: 200,
        temperature: 22 + Math.random() * 5,
        friction: 0.1 + Math.random() * 0.05,
        mass: 0.5 + Math.random() * 0.2
    }))
);
```

Рисунок 3.8 – Моделювання модульної структури робота

Такий підхід дозволяє кожному сегменту мати власні фізичні характеристики. Параметр angle визначає поточний кут повороту модуля, targetAngle задає цільову позицію для плавної анімації, а velocity та torque моделюють динамічні властивості руху. Температура, коефіцієнт тертя та маса додають реалістичності симуляції, враховуючи теплові та механічні обмеження реальних актуаторів.

```
const [sensors, setSensors] = useState({
    pressure: 1013.25,
    humidity: 45,
    temperature: 22,
    light: 750,
    distance: 150,
    accelerationX: 0,
    accelerationY: 0,
    accelerationZ: 9.81,
    gyroX: 0,
    gyroY: 0,
    gyroZ: 0
});
```

Рисунок 3.9 – Сенсорна підсистема

Для імітації реального робота була розроблена сенсорна підсистема, яка симулює різноманітні датчики.

Ці дані відображають стан навколишнього середовища та внутрішні параметри робота. Акселерометр та гіроскоп дозволяють відстежувати орієнтацію та динаміку руху, що критично важливо для стабільної локомоції. Датчик відстані використовується для уникнення перешкод, а температурні сенсори моніторять теплове навантаження на актуатори.

Інтеграція спеціалізованих модулів

Система включає декілька спеціалізованих підсистем, кожна з яких відповідає за певний аспект управління роботом.

ARC-OPT (Adaptive Recursive Constraint OPTimization) реалізує адаптивну оптимізацію траєкторій руху з урахуванням фізичних обмежень.

```
82    const [arcoptState, setArcoptState] = useState({
83        optimizing: false,
84        currentIteration: 0,
85        bestObjective: Infinity,
86        convergenceHistory: [],
87        constraints: { maxAngle: 45, maxTorque: 10, energyLimit: 100 }
88    });
```

Рисунок 3.10 – Модуль ARC-OPT

Цей модуль працює ітеративно, поступово покращуючи траєкторію руху робота. Обмеження на максимальні кути та крутний момент гарантують, що оптимізовані рухи залишаються фізично реалізовними. Історія збіжності дозволяє аналізувати ефективність оптимізаційного процесу.

BOLeRo (Behavior Optimization and LEarning for RObotics) впроваджує алгоритми машинного навчання для розробки адаптивних стратегій поведінки.

```
91      const [boleroState, setBoleroState] = useState({
92          training: false,
93          currentEpoch: 0,
94          totalReward: 0,
95          rewardHistory: [],
96          explorationRate: 0.3,
97          policyUpdates: 0
98      });
```

Рисунок 3.11 – Модуль BOLeRo

Система навчання базується на методах навчання з підкріпленням. Параметр explorationRate контролює баланс між дослідженням нових стратегій та експлуатацією вже знайдених рішень. З часом цей параметр зменшується, що відповідає поступовому переходу від експериментування до використання оптимальної політики.

HyRoDyn (Hybrid Robotics Dynamics) обчислює гібридну динаміку системи, що поєднує дискретні події контакту з безперервною динамікою руху.

```
101     const [hyrodynState, setHyrodynState] = useState({
102         mode: 'continuous',
103         contactForces: [],
104         energyConsumption: 0,
105         stabilityMetric: 85,
106         complianceLevel: 0.5
107     });
```

Рисунок 3.12 – Модуль HyRoDyn

Розрахунок контактних сил критично важливий для моделювання взаємодії робота з поверхнею. Метрика стабільності оцінює наскільки збалансована конфігурація робота, а рівень податливості визначає м'якість контакту при зіткненнях.

Серцем візуалізації є функція обчислення позицій модулів на основі їхніх кутів повороту.

```
399    const calculateModulePositions = (moduleAngles, baseX, baseY) => {
400        const positions = [];
401        let currentX = baseX;
402        let currentY = baseY;
403        let currentRotation = 0;
404
405        positions.push({ x: currentX, y: currentY, rotation: currentRotation });
406
407        for (let i = 1; i < moduleAngles.length; i++) {
408            currentRotation += moduleAngles[i].angle * (Math.PI / 180);
409            currentX += Math.cos(currentRotation) * SEGMENT_LENGTH;
410            currentY += Math.sin(currentRotation) * SEGMENT_LENGTH;
411            positions.push({ x: currentX, y: currentY, rotation: currentRotation });
412        }
413
414        return positions;
415    };
```

Рисунок 3.13 – Функція розрахунку кінематики

Дана функція реалізує розв’язання прямої кінематичної задачі для послідовного маніпулятора з жорстко з’єднаними ланками. Положення кожного наступного модуля визначається відносно попереднього з урахуванням відповідного кута повороту, що задає орієнтацію ланки в просторі. На основі тригонометричних залежностей виконується перетворення узагальнених кутових координат у декартові координати, що дозволяє обчислити просторове положення кожної ланки та кінцевого ефектора на площині.

Процедура оптимізації реалізована у вигляді ітеративного чисельного процесу, який виконується в окремому часовому або обчислювальному інтервалі. На кожній ітерації здійснюється корекція параметрів моделі з метою мінімізації заданої цільової функції, що може відображати похибку позиціонування, енергетичні витрати або інші критерії якості руху маніпулятора. Такий підхід забезпечує підвищення точності та стабільності роботи системи керування, а також дозволяє адаптувати алгоритм до змінних умов експлуатації.

```
    const runArcoptOptimization = () => {
        setArcoptState(prev => ({ ...prev, optimizing: true, currentIteration: 0 }));
        addAlert('info', 'ARC-OPT: Початок адаптивної оптимізації');

        const optimizationInterval = setInterval(() => {
            setArcoptState(prev => {
                if (prev.currentIteration >= 50) {
                    clearInterval(optimizationInterval);
                    addAlert('success', `ARC-OPT: Оптимізація завершена. Objective: ${prev.bestObjective.toFixed(4)}`);
                    return { ...prev, optimizing: false };
                }

                // Simulate optimization iteration
                const iteration = prev.currentIteration + 1;
                const objective = Math.exp(-iteration * 0.05) * Math.random() * 10;
                const newBest = Math.min(prev.bestObjective, objective);

                // Apply optimized angles to modules
                if (iteration % 10 === 0) {
                    setModules(prevMods => prevMods.map((m, i) => {
                        if (i === 0) return m;
                        const optimizedAngle = Math.sin(i * 0.3 + iteration * 0.1) * 25;
                        return { ...m, targetAngle: Math.round(optimizedAngle) };
                    }));
                }

                return {
                    ...prev,
                    currentIteration: iteration,
                    bestObjective: newBest,
                    convergenceHistory: [...prev.convergenceHistory.slice(-30), objective]
                };
            });

            setSystemStatus(prev => ({
                ...prev,
                arcopt: {
                    ...prev.arcopt,
                    iterations: prev.arcopt.iterations + 1,
                    convergence: Math.random() * 100,
                    objective: Math.random() * 10
                }
            }));
        }, 200);
    };
```

Рисунок 3.14– Алгоритм оптимізації ARC-OPT

Алгоритм імітує градієнтний спуск з експоненційним зменшенням цільової функції. Кожні 10 ітерацій оновлюються кути модулів згідно з оптимізованою траєкторією. Використання синусоїдальних функцій генерує плавні хвилеподібні рухи, характерні для змієподібної локомоції.

Навчання політики поведінки відбувається через симуляцію епізодів взаємодії з середовищем.

```
213     const startBoleroLearning = () => {
214         if (boleroState.training) return;
215
216         setBoleroState(prev => ({ ...prev, training: true, currentEpoch: 0 }));
217         addAlert('info', 'BOLeRo: Початок навчання політики поведінки');
218
219         const learningInterval = setInterval(() => {
220             setBoleroState(prev => {
221                 if (prev.currentEpoch >= 100) {
222                     clearInterval(learningInterval);
223                     addAlert('success', `BOLeRo: Навчання завершено. Reward: ${prev.totalReward.toFixed(2)}`)
224                     return { ...prev, training: false };
225                 }
226
227                 const epoch = prev.currentEpoch + 1;
228                 const reward = 10 - Math.abs(Math.sin(epoch * 0.1)) * 5 + Math.random() * 2;
229                 const totalReward = prev.totalReward + reward;
230
231                 // Update exploration rate
232                 const explorationRate = Math.max(0.05, 0.3 * Math.exp(-epoch * 0.02));
233
234                 // Apply learned behavior
235                 if (epoch % 5 === 0 && robotMode === 'autonomous') {
236                     setModules(prevMods => prevMods.map((m, i) => {
237                         if (i === 0) return m;
238                         const learnedAngle = Math.sin(i * 0.4 + epoch * 0.05) * 30 * (1 - explorationRate);
239                         return { ...m, targetAngle: Math.round(learnedAngle) };
240                     }));
241                 }
242
243                 return {
244                     ...prev,
245                     currentEpoch: epoch,
246                     totalReward: totalReward,
247                     rewardHistory: [...prev.rewardHistory.slice(-50), reward],
248                     explorationRate: explorationRate,
249                     policyUpdates: prev.policyUpdates + 1
250                 };
251             });
252
253             setSystemStatus(prev => ({
254                 ...prev,
255                 bolero: {
256                     ...prev.bolero,
257                     status: 'training',
258                     epoch: prev.bolero.epoch + 1
```

Рисунок 3.15 – Процедура навчання BOLeRo

Функція винагороди моделюється як синусоїдальна функція з додатковим шумом, що імітує складність реального середовища. Експоненційне зменшення швидкості дослідження забезпечує перехід від випадкових дій до цілеспрямованої поведінки в міру навчання.

Модуль HyRoDyn розраховує енергетичні характеристики та стабільність системи.

```
267     const calculateHybridDynamics = () => {
268         const totalEnergy = modules.reduce((sum, m) => {
269             const kineticEnergy = 0.5 * m.mass * Math.pow(m.velocity, 2);
270             const potentialEnergy = m.mass * 9.81 * (m.y / 100);
271             return sum + kineticEnergy + potentialEnergy;
272         }, 0);
273 
274         const stabilityScore = modules.reduce((sum, m) => {
275             const angleStability = 1 - Math.abs(m.angle) / 45;
276             const velocityStability = 1 - Math.abs(m.velocity) / 10;
277             return sum + (angleStability * velocityStability);
278         }, 0) / modules.length * 100;
279 
280         setHyrodynState(prev => ({
281             ...prev,
282             energyConsumption: totalEnergy,
283             stabilityMetric: stabilityScore,
284             contactForces: modules.map((m, i) => ({
285                 module: i,
286                 normal: m.torque * 0.5,
287                 friction: m.friction * m.torque
288             }))
289         }));
290 
291         setSystemStatus(prev => ({
292             ...prev,
293             hyrodyn: {
294                 ...prev.hyrodyn,
295                 dynamics: totalEnergy.toFixed(2),
296                 stability: Math.round(stabilityScore),
297                 energy: totalEnergy.toFixed(2),
298                 compliance: prev.hyrodyn.compliance
299             }
300         }));
301     };
```

Рисунок 3.16 – Обчислення гібридної динаміки

Розрахунок повної енергії включає кінетичну енергію руху та потенціальну енергію положення. Показник стабільності оцінює як кутову

конфігурацію, так і динамічний стан кожного модуля, забезпечуючи комплексну оцінку збалансованості системи.

Відображення робота реалізоване через Canvas API з використанням ефекту useEffect для перерисовки при зміні стану.

```
589         useEffect(() => {
590             const canvas = canvasRef.current;
591             if (!canvas) return;
592 
593             const ctx = canvas.getContext('2d');
594             const rect = canvas.getBoundingClientRect();
595             canvas.width = rect.width;
596             canvas.height = rect.height;
597 
598             ctx.clearRect(0, 0, canvas.width, canvas.height);
599 
600             if (showGrid) {
601                 ctx.strokeStyle = '#1e293b';
602                 ctx.lineWidth = 1;
603                 for (let x = 0; x < canvas.width; x += 20) {
604                     ctx.beginPath();
605                     ctx.moveTo(x, 0);
606                     ctx.lineTo(x, canvas.height);
607                     ctx.stroke();
608                 }
609                 for (let y = 0; y < canvas.height; y += 20) {
610                     ctx.beginPath();
611                     ctx.moveTo(0, y);
612                     ctx.lineTo(canvas.width, y);
613                     ctx.stroke();
614                 }
615             }
616 
617             const positions = calculateModulePositions(modules, robotPosition.x, robotPosition.y);
618 
619             if (showTrajectory && trajectoryHistory.length > 1) {
620                 ctx.strokeStyle = '#8b5cf6';
621                 ctx.lineWidth = 2;
622                 ctx.setLineDash([5, 5]);
623                 ctx.beginPath();
624                 ctx.moveTo(trajectoryHistory[0].x, trajectoryHistory[0].y);
625                 trajectoryHistory.forEach(point => {
626                     ctx.lineTo(point.x, point.y);
627                 });
628                 ctx.stroke();
629                 ctx.setLineDash([]);
630             }
631 
632             ctx.strokeStyle = '#3b82f6';
633             ctx.lineWidth = 8;
634             ctx.lineCap = 'round';
635             ctx.lineJoin = 'round';
```

Рисунок 3.17 – Система візуалізації на Canvas

Градієнтне забарвлення від червоного через синій до зеленого візуально підкреслює напрямок від голови до хвоста. Тіні додають глибини та об'ємності зображенню. Сітка координат полегшує просторову орієнтацію та оцінку переміщень.

Плавна анімація переходів між конфігураціями досягається через покроковий рух до цільових кутів.

```
505     useEffect(() => {
506         const animate = () => {
507             setModules(prev => {
508                 let hasChanges = false;
509                 const newModules = prev.map(m => {
510                     const diff = m.targetAngle - m.angle;
511                     if (Math.abs(diff) > 0.5) {
512                         hasChanges = true;
513                         const step = Math.sign(diff) * Math.min(Math.abs(diff), animationSpeed * 2);
514                         return { ...m, angle: Math.round(m.angle + step) };
515                     }
516                     return { ...m, angle: m.targetAngle };
517                 });
518 
519                 if (!hasChanges) setIsAnimating(false);
520                 return newModules;
521             });
522 
523             animationRef.current = requestAnimationFrame(animate);
524         };
525 
526         animationRef.current = requestAnimationFrame(animate);
527         return () => {
528             if (animationRef.current) cancelAnimationFrame(animationRef.current);
529         };
530     }, [animationSpeed]);
```

Рисунок 3.18 – Система анімації руху

```
436     const applyPresetMovement = (pattern) => {
437         if (isAnimating) return;
438 
439         const targetAngles = [...modules];
440 
441         switch(pattern) {
442             case 'sine':
443                 targetAngles.forEach((m, i) => {
444                     if (i > 0) m.targetAngle = Math.round(Math.sin(i * 0.5) * 30);
445                 });
446                 break;
447             case 's-curve':
448                 targetAngles.forEach((m, i) => {
449                     if (i > 0) m.targetAngle = i < 6 ? 20 : -20;
450                 });
451                 break;
452             case 'spiral':
453                 targetAngles.forEach((m, i) => {
454                     if (i > 0) m.targetAngle = Math.round(Math.min(45, i * 3.5));
455                 });
456                 break;
457             case 'wave':
458                 targetAngles.forEach((m, i) => {
459                     if (i > 0) m.targetAngle = Math.round(Math.sin(i * 0.8) * 25);
460                 });
461                 break;
462             case 'coil':
463                 targetAngles.forEach((m, i) => {
464                     if (i > 0) m.targetAngle = 30;
465                 });
466                 break;
467             default:
468                 break;
469         }
470 
471         setModules(targetAngles);
472         setIsAnimating(true);
```

Рисунок 3.19 –застосування різних патерн руху

Синусоїдальний шаблон генерує класичний хвилеподібний рух змії. Спіральний патерн поступово збільшує кути, створюючи закручену форму. Хвильовий рух використовує іншу частоту синусоїди для створення альтернативного стилю локомоції.

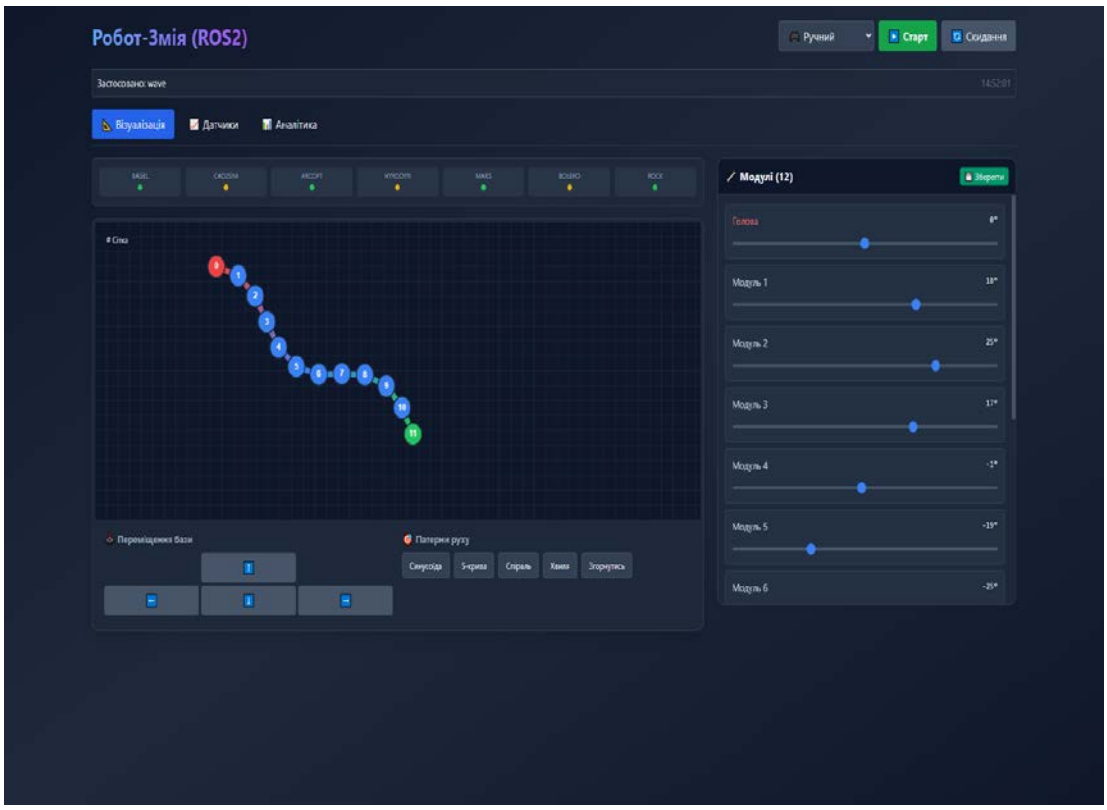


Рисунок 3.20 - Відтворення на візуалізації хвилеподібного руху.

Реалістичність системи підвищується через динамічне оновлення показань сенсорів.

```
696     useEffect(() => {
697         if (isRunning) {
698             const interval = setInterval(() => {
699                 setSensors(prev => {
700                     const newSensors = {
701                         pressure: prev.pressure + (Math.random() - 0.5) * 0.5,
702                         humidity: Math.max(0, Math.min(100, prev.humidity + (Math.random() - 0.5) * 2)),
703                         temperature: prev.temperature + (Math.random() - 0.5) * 0.3,
704                         light: Math.max(0, prev.light + (Math.random() - 0.5) * 20),
705                         distance: Math.max(10, Math.min(300, prev.distance + (Math.random() - 0.5) * 10)),
706                         accelerationX: (Math.random() - 0.5) * 2,
707                         accelerationY: (Math.random() - 0.5) * 2,
708                         accelerationZ: 9.81 + (Math.random() - 0.5) * 0.5,
709                         gyroX: (Math.random() - 0.5) * 10,
710                         gyroY: (Math.random() - 0.5) * 10,
711                         gyroZ: (Math.random() - 0.5) * 10
712                     };
713 
714                     setSensorHistory(prev => ({
715                         temperature: [...prev.temperature.slice(1), newSensors.temperature],
716                         distance: [...prev.distance.slice(1), newSensors.distance],
717                         light: [...prev.light.slice(1), newSensors.light],
718                         pressure: [...prev.pressure.slice(1), newSensors.pressure],
719                         humidity: [...prev.humidity.slice(1), newSensors.humidity]
720                     }));
721 
722                     return newSensors;
723                 });
```

Рисунок 3.21 – Симуляція сенсорних даних

Додавання випадкового шуму до кожного сенсора імітує реальні умови роботи, де завжди присутні флуктуації та похибки вимірювань. Обмеження на мінімальні та максимальні значення гарантують фізично правдоподібні дані.

Для збереження результатів експериментів реалізована функція експорту повного стану системи.

```
562     const exportData = () => {
563         const data = {
564             modules: modules.map(m => ({
565                 id: m.id,
566                 angle: m.angle,
567                 name: m.name
568             })),
569             sensors,
570             systemStatus,
571             arcopt: arcoptState,
572             bolero: boleroState,
573             hyrodyn: hyrodynState,
574             mars: marsState,
575             mmlf: mmlfState,
576             rock: rockState,
577             timestamp: new Date().toISOString()
578         };
579 
580         const blob = new Blob([JSON.stringify(data, null, 2)], { type: 'application/json' });
581         const url = URL.createObjectURL(blob);
582         const a = document.createElement('a');
583         a.href = url;
584         a.download = `snake-robot-${Date.now()}.json`;
585         a.click();
586         addAlert('success', 'Дані експортовано');
587     };
```

Рисунок 3.22 – Система експорту даних

Структурований JSON-формат забезпечує читабельність та можливість подальшого аналізу збережених даних. Часова мітка дозволяє ідентифікувати різні сесії експериментів.

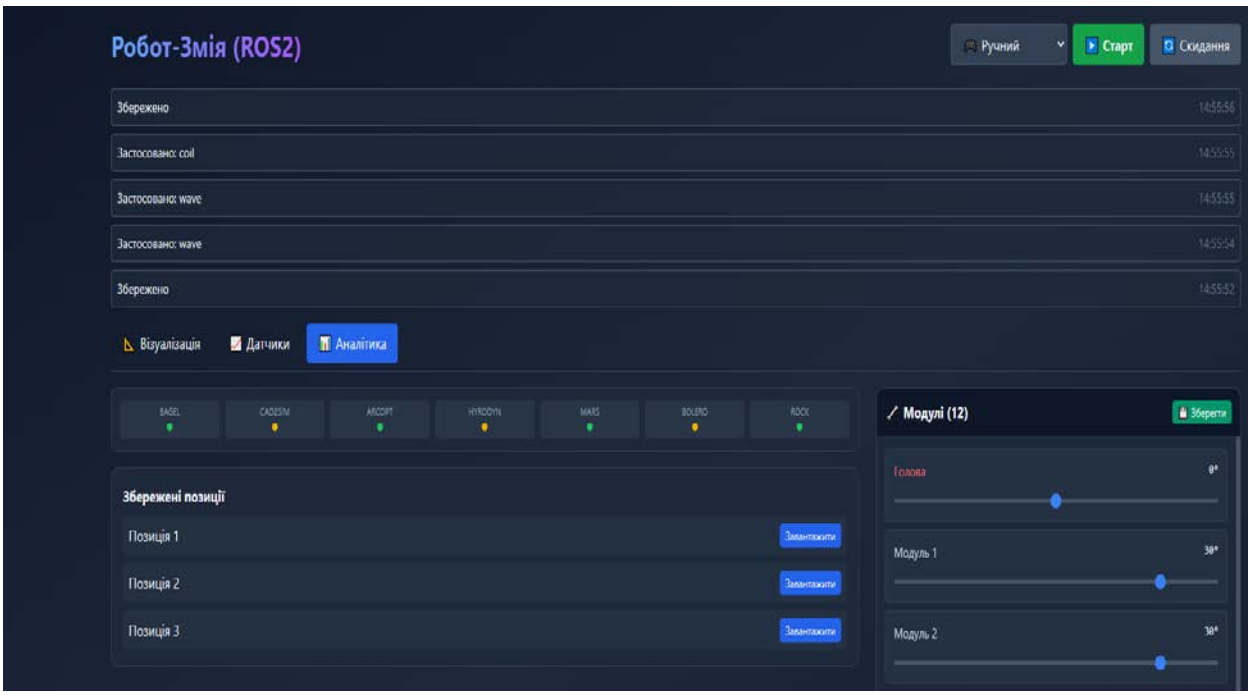


Рисунок 3.23 - Збереження позицій кожного модуля та відтворення в аналітиці

Висновки до розділу 3

Дослідження змієподібних роботів активно розвиваються протягом десятиліть, проте залишається значна кількість викликів, пов'язаних із їхнім моделюванням та керуванням, перш ніж ці роботи зможуть адаптивно переміщатися складним рельєфом. Модульні структури та алгоритми керування є двома ключовими аспектами для змієподібних роботів, що функціонують в обмеженому просторі.

Проте, адаптивне керування БПРЗ у режимі реального часу на нерівних поверхнях вимагає впровадження нових підходів для подолання недостатньої чутливості, високої нелінійності та невизначеності моделей робота. Важливим завданням, яке потребує вирішення, є інтеграція сенсорної інформації в контролер БПРЗ для забезпечення ефективного переміщення. Необхідною є розробка, виробництво та валідація спеціалізованого сенсорного механізму для БПРЗ для збору даних та ефективного руху по нерівних поверхнях.

У розділі представлено розробку механічної структури та системи управління біоморфного робота-змії. Описано конструкцію сегментів, шарнірів, силових модулів і сенсорних блоків, а також використання адитивних технологій

для виготовлення корпусних елементів. Наведено принципи роботи сервоприводів Dynamixel AX-12 та характеристику електронної системи на базі мікроконтролера ARM Cortex-M4.

Створена система представляє собою інтегроване середовище для дослідження біоморфної робототехніки. Поєднання інтерактивної візуалізації, реалістичного моделювання фізики та інтеграції алгоритмів машинного навчання створює потужний інструмент для експериментів з адаптивною локомоцією модульних роботів. Модульна архітектура коду дозволяє легко розширювати функціональність та адаптувати систему під конкретні дослідницькі задачі.

ВИСНОВКИ

1.З'ясовано, що впровадження технологій Нерозруйнівного контролю (НРК) є скрутним без автоматичного розпізнавання складних елементів, особливо без високоточного автоматизованого неруйнівного визначення компонентів з криволінійною поверхнею, що характеризується змінною кривизною, товщиною та складною формою контуру. Роботизація НРК (РНРК) розширює можливості інспектування, забезпечує доступ до важкодоступних зон та підвищує загальну ефективність процесу; хоча початкові вкладення у роботизоване устаткування можуть бути суттєвими, це призводить до значної економії коштів у перспективі. Одним із напрямів розвитку РНРК стало залучення біологічних принципів та структур для створення інженерних рішень та технологічних методик, відомих як біоніка, біоміметика або біомімікрія.

2.Пропонується розглядати біоподібного робота як автоматизований пристрій чи систему, що використовується для виконання різноманітних завдань та операцій, спроєктовану шляхом імітації будови чи функціоналу біологічно сформованої речовини чи матеріалу. При проєктуванні біоміметичних роботів ключовим аспектом є не буквальне копіювання природних структур і матеріалів, а усвідомлення принципів проєктування та фізико-хімічних механізмів, які зумовлюють оптимізовану організацію структури та геометрії біологічних систем, а також їхній зв'язок з багатогранними функціями.

3.Визначено різновиди роботизованих систем, які застосовуються для цілей НРК, та їхні специфічні характеристики, зокрема: дистанційно керовані апарати; безпілотні літальні апарати (БПЛА) або дрони; системи для повзання та вертикального переміщення; крабоподібні системи; магнітні гусеничні системи; автономні підводні апарати; роботизовані маніпулятори; колаборативні роботи; моделі на базі штучного інтелекту.

4.Наголошено, що конфігурація біоподібного робота неминуче залежатиме від його типу та функцій, які він має виконувати. Для забезпечення належного функціонування робота та якісного досягнення запланованих результатів

критично важливим є система управління, яка повинна бути адаптована до реального середовища. До головних вимог до системи управління належать такі параметри: здатність інтерпретувати ключові характеристики оточуючого простору (наприклад, висоту перешкод та відстань між роботом і ними); можливість моніторингу стабільності робота у режимі реального часу та підтримання його стійкого стану.

5.Індустрія 4.0 вносить корективи у підходи до застосування різноманітних систем та технологій, включно з робототехнікою. Роботизований НРК має розвиватися паралельно з розробкою новітніх інструментів, таких як автономна робототехніка, створення віртуальних двійників, Інтернет речей, кібербезпека та хмарні обчислення.

6.Визначено базові концепції конструювання біоподібного робота на прикладі Біоподібного Робота-Змії (БПРЗ), обраного для дослідження, з огляду на такі критерії: функціональна гнучкість (можливість роботи як на суші, так і у воді); висока прохідність; модульність конструкції; надійний захист від проникнення пилу та рідин (герметичність); стійкість до механічних впливів. Це універсальні механізми, що складаються з великої кількості ланок та зчленувань, які дозволяють здійснювати гнучкі переміщення, імітуючи рухи справжньої змії, і на відміну від колісних, гусеничних або ходових механізмів, забезпечують високий ступінь стабільності, при цьому БПРЗ, як правило, за своєю суттю є більш міцним, реалізуючи різноманітні способи локомоції.

7.Математична модель БПРЗ, безумовно, прямо залежить від його конструктивних особливостей. Для класифікації різних конфігурацій БПРЗ враховуються основні властивості, такі як тип з'єднань; кількість ступенів свободи; наявність або відсутність пасивних коліщаток. Математична модель мобільного робота складається з кінематичної та динамічної частин. Кінематична модель служить для визначення координат у поточному положенні x, y та кута θ, що відображає орієнтацію мобільного робота відносно обраної системи координат. Вхідними даними для кінематичної моделі є кутові швидкості ω_L, ω_R, що генеруються динамічною моделлю. З рівнянь стану

динамічної моделі мобільного робота може бути виведено його описовий стан.

8.Механічна структура БПРЗ складається з трьох основних модулів: голови, тулуба та хвоста. Система управління, у свою чергу, складається з трьох головних компонентів: центральної системи управління, підпорядкованих систем управління та системи моніторингу. У голову БПРЗ інтегрований вузол, який переважно виконує функцію автоматичного визначення зовнішнього середовища і керує веденою системою управління за допомогою відповідних команд. Системи управління розташовані вздовж тулуба та хвоста, забезпечуючи специфічне регулювання та контроль локомоції. Спільні модулі проєктуються так, щоб задовольнити просторові вимоги для розміщення акумуляторних блоків, друкованих плат керування та двигунів.

9.Розроблено програмну архітектуру системи управління, що включає моделювання сегментів, сенсорну підсистему, візуалізацію кінематики та анімацію руху. Інтегровано три спеціалізовані модулі: ARC-OPT для оптимізації траєкторій, BOLeRo для навчання поведінки та HyRoDyn для розрахунку гібридної динаміки. Розроблена система поєднує в собі можливість візуалізувати процеси в реальному часі, реалістично моделювати фізичні взаємодії та інтегрувати алгоритми штучного інтелекту. Це робить її цінним інструментом для експериментів з адаптивним рухом модульних роботів. Завдяки гнучкій структурі програмного забезпечення, систему легко доповнювати новими функціями та пристосовувати до специфічних дослідницьких цілей.

Реалізовано бібліотеку шаблонних рухів, систему оновлення сенсорних даних та механізм експорту результатів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Менеджмент стартап-проєктів [Електронний ресурс]: навчальний наочний посібник для студентів спеціальностей 051 «Економіка», 073 «Менеджмент», 075 «Маркетинг» / О. А. Гавриш, К. О. Бояринова, М. О. Кравченко, К. О. Копішинська ; КПІ ім. Ігоря Сікорського. – Київ : КПІ ім. Ігоря Сікорського, 2021. 435 с.

2. Основні напрями удосконалення системи управління біоподібними роботами в неруйнівному контролі в умовах Індустрії 4.0» // Куранда А.В., Киричук Ю.В. – Збірник праць XIX Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених «Ефективність та автоматизація інженерних рішень у приладобудуванні», 20 – 21 грудня 2023 р. К.: ПБФ, КПІ ім. Ігоря Сікорського. 2023. 480 с. (С. 185 – 188)

3. Розроблення стартап-проекту [Електронний ресурс] : Методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / За заг. ред. О.А. Гавриша. Київ : НТУУ «КПІ», 2016. 28 с.

4. A Novel Modular Biomimetic LiveWorking Robot for Power Distribution Line // Jianbin Luo, Peng Guo and Yueke Lin. Machines 2022, 10(3), 195; https://doi.org/10.3390/machines10030195

5. Analysis of the Snake Robot Kinematics with Virtual Reality Visualisation // Anna Sibilska-Mroziewicz, Ayesha Hameed, Jakub Możaryn, Andrzej Ordys, Krzysztof Sibilski Sensors 2023, 23(6), 3262; https://doi.org/10.3390/s23063262

6. ARC-OPT - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https:// github.com/ARC-OPT

7. Avdelidis, N.P.; Tsourdos, A.; Lafiosca, P.; Plaster, R.; Plaster, A.; Droznika, M. Defects Recognition Algorithm Development from Visual UAV Inspections. Sensors 2022, 22.

8. Bagel - Softwaretools: Robotics Innovation Center - DFKI GmbH (dfki-bremen.de) Режим доступу: https://robotik.dfki-

bremen.de/en/research/softwaretools/bagel Berkshire Engineering | Guardian S – Remote Visual Inspection (berk- eng.com) Режим доступу: https://www.berk-eng.com/sarcos-robotics/guardian-s/

9. Biomimetic control system for a robot hand | Download Scientific Diagram (researchgate.net) Режим доступу: https://www.researchgate.net/figure/Biomimetic-control-system-for-a-robot-hand

10. Biomimetic Functional Structures – LAMBDA Режим доступу: https://coefs.charlotte.edu/ejoyee/research/biomimetic-functional-structures/

11. Biomimetic hexapod robot; Leg structure parameters; Control... | Download Scientific Diagram Режим доступу: https://www.researchgate.net/figure/a-Biomimetic-hexapod-robot-b-Leg-structure-parameters-c-Control-diagram-d_fig2_321261856

12. Biomimetics | Free Full-Text | Research and Experiment on a Bionic Fish Based on High-Frequency Vibration Characteristics Режим доступу: https://www.mdpi.com/2313-7673/8/2/253

13. Biomimetics | Free Full-Text | Sensor Fusion-Based Teleoperation Control of Anthropomorphic Robotic Arm Режим доступу: https://www.mdpi.com/2313- 7673/8/2/169

14. BiomiMETRIC Assistance Tool: A Quantitative Performance Tool for Biomimetic Design// Philippe Terrier, Mathias Glaus,Emmanuel Raufflet. Biomimetics 2019, 4(3), 49; https://doi.org/10.3390/biomimetics4030049

15. BOLeRo - Softwaretools: Robotics Innovation Center - DFKI GmbH (dfki-bremen.de) Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/bolero

16. CAD-2-SIM - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/cad-2- sim

17. Chris Fonceca Robotics for NDT (Non-Destructive Testing) Режим доступу: https://mfe-is.com/robotics-for-ndt-non-destructive-testing/

18. Difference Between Bionics and Biomimetics Режим доступу: https://www.differencebetween.com/difference-between-bionics-and-vs- biomimetics/

19. Design and Construction of aSnake-Like Robot Implementing Rectilinear and Sidewinding Gait Motions// Jairo José Marín Arciniegas, Oscar Andrés Vivas Albán TecnoL. (Inst. Tecnol. Metrop.) 2023; 26 (56). 1-17.

20. Design and Implementation of a Snake-like Robot with Amplitude-Controlled Phase Oscillator-based Motion Control // Serkan Karaçöl, Deniz Korkmaz, Gonca Ozmen Koca. Conference: 1st International Conference on Computing and Machine Intelligence (ICMI 2021)

21. Design of a motion system for 3D printed snakebot // Krzysztof Mateja, Wawrzyniec Panfil. Technical Sciences. DOI:10.31648/ts.682

22. Digital Twins - Revolutionizing Inspection Reports | MFE Inspection Solutions– Режим доступа: https:// mfe-is.com/digital-twins-revolutionizing-inspection-reports – 02.12.2023 p.

23. Environmental Adaptive Control of a Snake-like Robot With Variable Stiffness Actuators //Dong Zhang , Hao Yuan , Zhengcai Cao , doi: 10.1109/JAS.2020.1003144

24. HyRoDyn - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/hyrodyn

25. Industry 4.0 technologies assessment: A sustainability perspective //Chunguang Bai, Patrick Dallasega, Guido Orzes, Joseph Sarkis. International Journal of Production Economics. Volume 229, November 2020, https://doi.org/10.1016/j.ijpe.2020.107776

A. J. Ijspeert, Central pattern generators for locomotion control in animals and robots: a review Neural networks, 21(4), 2008, pp.642-653.

26. MARS - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/mars

27. Mineo, C.; Javadi, Y. Robotic Non-Destructive Testing. Sensors 2022, 22, 7654. https:// doi.org/10.3390/s22197654

28. MMLF - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/mmlf

29. N. M. Nor and S. Ma, A simplified CPGs network with phase oscillator model for locomotion control of a snake-like robot Journal of Intelligent and Robotic Systems, 75(1), 2014, pp.71-86. Режим доступу: https:// link.springer.com/article/10.1007/s10846-013-9868-9

30. NDE 4.0: Digitale Transformation und ihre Auswirkungen auf die Zerstörungsfreie Prüfung Режим доступу: https:// vision.fraunhofer.de/de/technologien-anwendungen/technologien/zerstoerungsfreie

31. NDE 4.0: Progress, promise, and its role to industry 4.0 // Norbert Meyendorf , Nathan Ida, Ripudaman Singh , Johannes Vrana NDT & E International Volume 140, December 2023, 102957 https://doi.org/10.1016/j.ndteint.2023.102957

32. NDLCom - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/ndlcom

33. Parameterized and Scripted Gaits for Modular Snake Robots// Matthew Tesch, Kevin Lipkin, Isaac Brown, Ross Hatton, Aaron Peck, Justine Rembisz & Howie Choset. Advanced Robotics Volume 23, 2009 - Issue 9: Disaster Response Robotics.

34. Ranjan Vepa Engineered Biomimicry Режим доступу: https://www.sciencedirect.com/science/article/abs/pii/B9780124159952000040

35. reSPACE - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/respace

36. Robotic for Non-Destructive Inspection Режим доступу: https:// www. mfe-is.com/robotics-for-ndt-non-destructive-testing/

37. Robotic path planning for non-destructive testing – A custom MATLAB toolbox approach — ScienceDirect Режим доступу: https://www.sciencedirect.com/science/article/pii/S0736584515000666#bib2

38. Robotic Systems in NDT Applications — OnestopNDT Режим доступу: https://www.onestopndt.com/ndt-articles/robotic-systems-in-ndt

39. Rock - Softwaretools: Robotics Innovation Center - DFKI GmbH Режим доступу: https://robotik.dfki-bremen.de/en/research/softwaretools/rock

40. SenSnake: A snake robot with contact force sensing for studying locomotion in complex 3-D terrain //Divya Ramesh, Qiyuan Fu, Chen Li Conference: IEEE International Conference on Robotics and Automation (ICRA) 2022 At: Philadelphia, Pennsylvania DOI:10.1109/ICRA46639.2022.9812159

41. Sensor Fusion-Based Teleoperation Control of Anthropomorphic Robotic Arm Biomimetics 2023, 8(2), 169; https://doi.org/10.3390/biomimetics8020169

42. Snake robots // Kristin Y. Pettersen Annual Reviews in Control Volume 44, 2017, Pages 19-44. https://doi.org/10.1016/j.arcontrol.2017.09.006.

43. Snake-Like Robot with Fusion Gait for High Environmental Adaptability: Design, Modeling, and Experiment // Softwaretools: Robotics Innovation Center - DFKI GmbH

44. Steering motion control of a snake robot via a biomimetic approach // Wenjuan Ouyang, Wenyu Liang, Chenzui Li, Hui Zheng, Qinyuan Ren & Ping Li . Frontiers of Information Technology & Electronic Engineering . Volume 20, pages 32–44, (2019)

45. Svinov A, Turygin YV, Sitar J. Remote control system and its development for linear asynchronous motor. In: *13th International symposium on mechatronics, mechatronika* 2010, Slovakia. ISBN: 978-808075461-7. 2–4 June 2010, Trencianske Teplice

46. Teoh, E. K., & Wong, C. Y. (1991). An expert system for real-time control of the sir-3 robotic system. in 1991, IEEE International Sympoisum on Circuits and Systems pp. 2709 – 2712. IEEE.

47. The Fourth Industrial Revolution, by Klaus Schwab | World Economic Forum – Режим доступу: https://www.weforum.org/about/the-fourth-industrial-revolution-by-klaus-schwab – 03.12.2023 р.

48. Trajectory tracking control law of multi-joint snake-like robot based on improved snake-like curve in flow field - Dongfang Li, Zhenhua Pan, Hongbin Deng,

Teng Peng, 2019

49. Transeth A. A., Leine R. I., and Pettersen K. Y., “3-D snake robot motion: nonsmooth modeling, simulations, and experiments,” *IEEE Trans. Robotics*, vol. 24, no. 2, pp. 361–376, 2008. doi: 10.1109/TRO.2008.917003

50. Transforming Non-destructive Testing Procedures Through Robotics – Режим доступу: https://www.onestopndt.com/ndt-articles/transforming-non-destructive-testing-procedures-through-robot – 01.12.2023 р.

51. Turning and Radius Deviation Correction for a Hexapod Walking Robot Based on an Ant-Inspired Sensory Strategy // Yaguang Zhu, Tong Guo, Qiong Liu, Qianwei Zhu, Xiangmo Zhao and Bo Jin Sensors 2017, 17(12), 2710; https://doi.org/10.3390/s17122710

52. Types of Sensors in Robotics – Режим доступу: https://www.wevolver.com/article/sensors-in-robotics-the-common-types – 01.12.2023 р.

53. What Is a Robot? - ROBOTS: Your Guide to the World of Robotics Режим доступу: https://robotsguide.com/learn/what-is-a-robot

54. What is an automatic control system? - AutomationForum Режим доступу: https://automationforum.co/what-is-an-automatic-control-system/

55. Z. Bing, L. Cheng, K. Huang, M. Zhou, and A. Knoll, CPG-based control of smooth transition for body shape and locomotion speed of a snake-like robot In 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp.4146-4153. https:// dl.acm.org/doi/10.1109/ICRA.2017.7989476

```
import React, { useState, useEffect, useRef } from 'react';
import { Camera, Thermometer, Droplets, Gauge, Sun, Activity, Video, Eye, Settings, Play, Pause, RotateCcw, Save, Upload, Zap, AlertTriangle, CheckCircle, Radio, Wifi, Battery, Cpu, HardDrive, Clock, Download, FolderOpen, BarChart3, Layers, Move, ArrowUp, ArrowDown, ArrowLeft, ArrowRight, Workflow, Brain, Code, GitBranch, Network, Database, TrendingUp, Target } from 'lucide-react';

const SnakeRobotControl = () => {
  const canvasRef = useRef(null);
  const animationRef = useRef(null);
  const [isDragging, setIsDragging] = useState(false);
  const [draggedModule, setDraggedModule] = useState(null);

  const [modules, setModules] = useState(
    Array(12).fill(0).map((_, i) => ({
      id: i,
      angle: 0,
      targetAngle: 0,
      velocity: 0,
      torque: 0,
      active: true,
      name: i === 0 ? 'Голова' : i === 11 ? 'Хвіст' : `Модуль ${i}`,
      x: 100 + i * 50,
      y: 200,
      temperature: 22 + Math.random() * 5,
      friction: 0.1 + Math.random() * 0.05,
      mass: 0.5 + Math.random() * 0.2
    }))
  );

  const [robotPosition, setRobotPosition] = useState({ x: 150, y: 200 });

  const [sensors, setSensors] = useState({
    pressure: 1013.25,
    humidity: 45,
    temperature: 22,
    light: 750,
    distance: 150,
    accelerationX: 0,
```

```
    accelerationY: 0,
    accelerationZ: 9.81,
    gyroX: 0,
    gyroY: 0,
    gyroZ: 0
  });

  const [cameras, setCameras] = useState({
    normal: true,
    infrared: false
  });

  const [systemStatus, setSystemStatus] = useState({
    arcopt: { status: 'active', load: 67, iterations: 0, convergence: 0, objective: 0 },
    bolero: { status: 'standby', epoch: 0, reward: 0, exploration: 0, policy:
'contextual' },
    cad2sim: { status: 'ready', models: 3, accuracy: 98.5, meshQuality: 95,
lastSync: Date.now() },
    hyrodyn: { status: 'calculating', dynamics: 0, stability: 85, energy: 0,
compliance: 0 },
    mars: { status: 'running', simTime: 0, fps: 60, entities: 12, physics: 'bullet' },
    mmlf: { status: 'standby', learning: 0, episodes: 0, performance: 0, algorithm:
'TD3' },
    rock: { status: 'active', channels: 7, latency: 12, bandwidth: 85, packets: 0 }
  });

  const [robotMode, setRobotMode] = useState('manual');
  const [isRunning, setIsRunning] = useState(false);
  const [selectedTab, setSelectedTab] = useState('visualization');
  const [trajectoryHistory, setTrajectoryHistory] = useState([]);
  const [performance, setPerformance] = useState({
    cpuUsage: 45,
    memoryUsage: 60,
    networkLatency: 12,
    batteryLevel: 85,
    powerConsumption: 23.5
  });

  const [alerts, setAlerts] = useState([]);
  const [savedPositions, setSavedPositions] = useState([]);
  const [animationSpeed, setAnimationSpeed] = useState(1);
```

```
const [showGrid, setShowGrid] = useState(true);
const [showTrajectory, setShowTrajectory] = useState(true);
const [isAnimating, setIsAnimating] = useState(false);

// ARC-OPT optimization state
const [arcoptState, setArcoptState] = useState({
  optimizing: false,
  currentIteration: 0,
  bestObjective: Infinity,
  convergenceHistory: [],
  constraints: { maxAngle: 45, maxTorque: 10, energyLimit: 100 }
});

// BOLeRo learning state
const [boleroState, setBoleroState] = useState({
  training: false,
  currentEpoch: 0,
  totalReward: 0,
  rewardHistory: [],
  explorationRate: 0.3,
  policyUpdates: 0
});

// HyRoDyn dynamics state
const [hyrodynState, setHyrodynState] = useState({
  mode: 'continuous',
  contactForces: [],
  energyConsumption: 0,
  stabilityMetric: 85,
  complianceLevel: 0.5
});

// MARS simulation state
const [marsState, setMarsState] = useState({
  simulationTime: 0,
  realTimeRatio: 1.0,
  collisionDetections: 0,
  physicsIterations: 0,
  sceneComplexity: 'medium'
});
```

```
// MMLF learning state
const [mmlfState, setMmlfState] = useState({
  algorithm: 'TD3',
  totalEpisodes: 0,
  averageReturn: 0,
  learningCurve: [],
  explorationNoise: 0.1
});

// Rock communication state
const [rockState, setRockState] = useState({
  activeConnections: 7,
  dataRate: 0,
  totalPackets: 0,
  latencyHistory: [],
  channelHealth: Array(7).fill(100)
});

// CAD2SIM state
const [cad2simState, setCad2simState] = useState({
  loadedModels: ['snake_body', 'actuator', 'sensor_suite'],
  simAccuracy: 98.5,
  lastUpdate: Date.now(),
  geometrySync: true
});

const presetMovements = [
  { name: 'Синусоїда', pattern: 'sine' },
  { name: 'S-крива', pattern: 's-curve' },
  { name: 'Спіраль', pattern: 'spiral' },
  { name: 'Хвиля', pattern: 'wave' },
  { name: 'Згорнутись', pattern: 'coil' }
];

const [sensorHistory, setSensorHistory] = useState({
  temperature: Array(20).fill(22),
  distance: Array(20).fill(150),
  light: Array(20).fill(750),
  pressure: Array(20).fill(1013.25),
```

```
  humidity: Array(20).fill(45)
 });

 const MODULE_RADIUS = 15;
 const SEGMENT_LENGTH = 40;

 // ARC-OPT: Adaptive optimization function
 const runArcoptOptimization = () => {
  if (arcoptState.optimizing) return;

  setArcoptState(prev => ({ ...prev, optimizing: true, currentIteration: 0 }));
  addAlert('info', 'ARC-OPT: Початок адаптивної оптимізації');

  const optimizationInterval = setInterval(() => {
   setArcoptState(prev => {
    if (prev.currentIteration >= 50) {
     clearInterval(optimizationInterval);
     addAlert('success', `ARC-OPT: Оптимізація завершена. Objective:
${prev.bestObjective.toFixed(4)}`);
     return { ...prev, optimizing: false };
    }

    // Simulate optimization iteration
    const iteration = prev.currentIteration + 1;
    const objective = Math.exp(-iteration * 0.05) * Math.random() * 10;
    const newBest = Math.min(prev.bestObjective, objective);

    // Apply optimized angles to modules
    if (iteration % 10 === 0) {
     setModules(prevMods => prevMods.map((m, i) => {
      if (i === 0) return m;
      const optimizedAngle = Math.sin(i * 0.3 + iteration * 0.1) * 25;
      return { ...m, targetAngle: Math.round(optimizedAngle) };
     }));
    }

    return {
     ...prev,
     currentIteration: iteration,
     bestObjective: newBest,
```

```
        convergenceHistory: [...prev.convergenceHistory.slice(-30), objective]
       };
      });

      setSystemStatus(prev => ({
       ...prev,
       arcopt: {
        ...prev.arcopt,
        iterations: prev.arcopt.iterations + 1,
        convergence: Math.random() * 100,
        objective: Math.random() * 10
       }
      }));
    }, 200);
  };

  // BOLeRo: Behavior optimization and learning
  const startBoleroLearning = () => {
    if (boleroState.training) return;

    setBoleroState(prev => ({ ...prev, training: true, currentEpoch: 0 }));
    addAlert('info', 'BOLeRo: Початок навчання політики поведінки');

    const learningInterval = setInterval(() => {
      setBoleroState(prev => {
        if (prev.currentEpoch >= 100) {
          clearInterval(learningInterval);
          addAlert('success', `BOLeRo: Навчання завершено. Reward:
${prev.totalReward.toFixed(2)}`);
          return { ...prev, training: false };
        }

        const epoch = prev.currentEpoch + 1;
        const reward = 10 - Math.abs(Math.sin(epoch * 0.1)) * 5 + Math.random()
* 2;
        const totalReward = prev.totalReward + reward;

        // Update exploration rate
        const explorationRate = Math.max(0.05, 0.3 * Math.exp(-epoch * 0.02));
```

```
      // Apply learned behavior
      if (epoch % 5 === 0 && robotMode === 'autonomous') {
        setModules(prevMods => prevMods.map((m, i) => {
          if (i === 0) return m;
          const learnedAngle = Math.sin(i * 0.4 + epoch * 0.05) * 30 * (1 -
explorationRate);
          return { ...m, targetAngle: Math.round(learnedAngle) };
        }));
      }

      return {
        ...prev,
        currentEpoch: epoch,
        totalReward: totalReward,
        rewardHistory: [...prev.rewardHistory.slice(-50), reward],
        explorationRate: explorationRate,
        policyUpdates: prev.policyUpdates + 1
      };
    });

    setSystemStatus(prev => ({
      ...prev,
      bolero: {
        ...prev.bolero,
        status: 'training',
        epoch: prev.bolero.epoch + 1,
        reward: Math.random() * 10,
        exploration: Math.max(5, 30 - prev.bolero.epoch * 0.2)
      }
    }));
  }, 300);
 };

 // HyRoDyn: Hybrid dynamics calculation
 const calculateHybridDynamics = () => {
   const totalEnergy = modules.reduce((sum, m) => {
     const kineticEnergy = 0.5 * m.mass * Math.pow(m.velocity, 2);
     const potentialEnergy = m.mass * 9.81 * (m.y / 100);
     return sum + kineticEnergy + potentialEnergy;
   }, 0);
```

```
    const stabilityScore = modules.reduce((sum, m) => {
      const angleStability = 1 - Math.abs(m.angle) / 45;
      const velocityStability = 1 - Math.abs(m.velocity) / 10;
      return sum + (angleStability * velocityStability);
    }, 0) / modules.length * 100;

    setHyrodynState(prev => ({
      ...prev,
      energyConsumption: totalEnergy,
      stabilityMetric: stabilityScore,
      contactForces: modules.map((m, i) => ({
        module: i,
        normal: m.torque * 0.5,
        friction: m.friction * m.torque
      }))
    }));

    setSystemStatus(prev => ({
      ...prev,
      hyrodyn: {
        ...prev.hyrodyn,
        dynamics: totalEnergy.toFixed(2),
        stability: Math.round(stabilityScore),
        energy: totalEnergy.toFixed(2),
        compliance: prev.hyrodyn.compliance
      }
    }));
  };

  // MARS: Simulation management
  const updateMarsSimulation = () => {
    setMarsState(prev => ({
      ...prev,
      simulationTime: prev.simulationTime + 0.016,
      physicsIterations: prev.physicsIterations + 1,
      collisionDetections: prev.collisionDetections + Math.floor(Math.random() *
2)
    }));
```

```
  setSystemStatus(prev => ({
    ...prev,
    mars: {
      ...prev.mars,
      simTime: prev.mars.simTime + 0.016,
      fps: 58 + Math.random() * 4
    }
  }));
};

// MMLF: Machine learning framework
const updateMmlfLearning = () => {
  if (robotMode === 'autonomous' && isRunning) {
    setMmlfState(prev => {
      const episode = Math.floor(prev.simulationTime / 10);
      const returnValue = Math.random() * 100 - 50 + episode * 0.5;

      return {
        ...prev,
        totalEpisodes: episode,
        averageReturn: returnValue,
        learningCurve: [...prev.learningCurve.slice(-50), returnValue],
        explorationNoise: Math.max(0.05, 0.2 * Math.exp(-episode * 0.01))
      };
    });

    setSystemStatus(prev => ({
      ...prev,
      mmlf: {
        ...prev.mmlf,
        status: 'learning',
        learning: Math.random() * 100,
        episodes: prev.mmlf.episodes + 1,
        performance: 50 + Math.random() * 30
      }
    }));
  }
};

// Rock: Communication management
```

```
const updateRockCommunication = () => {
  setRockState(prev => {
    const newLatency = 8 + Math.random() * 8;
    const packetsPerSecond = 100 + Math.random() * 50;

    return {
      ...prev,
      dataRate: packetsPerSecond * 0.512,
      totalPackets: prev.totalPackets + packetsPerSecond,
      latencyHistory: [...prev.latencyHistory.slice(-30), newLatency],
      channelHealth: prev.channelHealth.map(h =>
        Math.max(80, Math.min(100, h + (Math.random() - 0.5) * 5))
      )
    };
  });

  setSystemStatus(prev => ({
    ...prev,
    rock: {
      ...prev.rock,
      latency: 8 + Math.random() * 8,
      bandwidth: 75 + Math.random() * 20,
      packets: prev.rock.packets + 100
    }
  }));
};

// CAD2SIM: Sync geometry
const syncCad2Sim = () => {
  setCad2simState(prev => ({
    ...prev,
    lastUpdate: Date.now(),
    simAccuracy: 97 + Math.random() * 3
  }));

  setSystemStatus(prev => ({
    ...prev,
    cad2sim: {
      ...prev.cad2sim,
      accuracy: 97 + Math.random() * 3,
```

```
      meshQuality: 93 + Math.random() * 5
    }
  }));

  addAlert('success', 'CAD2SIM: Геометрія синхронізована');
};

const calculateModulePositions = (moduleAngles, baseX, baseY) => {
  const positions = [];
  let currentX = baseX;
  let currentY = baseY;
  let currentRotation = 0;

  positions.push({ x: currentX, y: currentY, rotation: currentRotation });

  for (let i = 1; i < moduleAngles.length; i++) {
    currentRotation += moduleAngles[i].angle * (Math.PI / 180);
    currentX += Math.cos(currentRotation) * SEGMENT_LENGTH;
    currentY += Math.sin(currentRotation) * SEGMENT_LENGTH;
    positions.push({ x: currentX, y: currentY, rotation: currentRotation });
  }

  return positions;
};

const moveRobot = (direction) => {
  const step = 20;
  setRobotPosition(prev => {
    let newX = prev.x;
    let newY = prev.y;

    switch(direction) {
      case 'up': newY -= step; break;
      case 'down': newY += step; break;
      case 'left': newX -= step; break;
      case 'right': newX += step; break;
      default: break;
    }

    return { x: newX, y: newY };
```

```
  });
  addAlert('info', `Переміщення: ${direction}`);
};

const applyPresetMovement = (pattern) => {
  if (isAnimating) return;

  const targetAngles = [...modules];

  switch(pattern) {
    case 'sine':
      targetAngles.forEach((m, i) => {
        if (i > 0) m.targetAngle = Math.round(Math.sin(i * 0.5) * 30);
      });
      break;
    case 's-curve':
      targetAngles.forEach((m, i) => {
        if (i > 0) m.targetAngle = i < 6 ? 20 : -20;
      });
      break;
    case 'spiral':
      targetAngles.forEach((m, i) => {
        if (i > 0) m.targetAngle = Math.round(Math.min(45, i * 3.5));
      });
      break;
    case 'wave':
      targetAngles.forEach((m, i) => {
        if (i > 0) m.targetAngle = Math.round(Math.sin(i * 0.8) * 25);
      });
      break;
    case 'coil':
      targetAngles.forEach((m, i) => {
        if (i > 0) m.targetAngle = 30;
      });
      break;
    default:
      break;
  }

  setModules(targetAngles);
```

```
    setIsAnimating(true);
    addAlert('info', `Застосовано: ${presetMovements.find(p => p.pattern === pattern)?.name}`);
  };

  useEffect(() => {
    if (robotMode === 'autonomous' && isRunning) {
      const autonomousInterval = setInterval(() => {
        setModules(prev => prev.map((m, i) => {
          if (i === 0) return m;

          const randomChange = (Math.random() - 0.5) * 10;
          let newAngle = m.targetAngle + randomChange;
          newAngle = Math.max(-45, Math.min(45, Math.round(newAngle)));

          return { ...m, targetAngle: newAngle };
        }));

        if (Math.random() < 0.3) {
          const messages = [
            'Навчання: епоха 47/100',
            'Оптимізація траєкторії...',
            'Знайдено кращий рух',
            'Loss: 0.0245 → 0.0198',
            'Accuracy покращено на 3.2%'
          ];
          addAlert('info', messages[Math.floor(Math.random() * messages.length)]);
        }
      }, 2000);

      return () => clearInterval(autonomousInterval);
    }
  }, [robotMode, isRunning]);

  useEffect(() => {
    const animate = () => {
      setModules(prev => {
        let hasChanges = false;
        const newModules = prev.map(m => {
```

```
        const diff = m.targetAngle - m.angle;
        if (Math.abs(diff) > 0.5) {
          hasChanges = true;
          const step = Math.sign(diff) * Math.min(Math.abs(diff), animationSpeed
* 2);
          return { ...m, angle: Math.round(m.angle + step) };
        }
        return { ...m, angle: m.targetAngle };
       });

       if (!hasChanges) setIsAnimating(false);
       return newModules;
     });

     animationRef.current = requestAnimationFrame(animate);
   };

   animationRef.current = requestAnimationFrame(animate);
   return () => {
     if (animationRef.current) cancelAnimationFrame(animationRef.current);
   };
 }, [animationSpeed]);

 const saveCurrentPosition = () => {
   const position = {
     id: Date.now(),
     name: `Позиція ${savedPositions.length + 1}`,
     angles: modules.map(m => m.angle),
     timestamp: new Date().toLocaleTimeString('uk-UA')
   };
   setSavedPositions([...savedPositions, position]);
   addAlert('success', 'Позицію збережено');
 };

 const loadPosition = (position) => {
   setModules(prev => prev.map((m, i) => ({
     ...m,
     targetAngle: position.angles[i]
   })));
   setIsAnimating(true);
```

```
    addAlert('info', `Завантажено: ${position.name}`);
  };

  const addAlert = (type, message) => {
    const alert = {
      id: Date.now(),
      type,
      message,
      timestamp: new Date().toLocaleTimeString('uk-UA')
    };
    setAlerts(prev => [alert, ...prev].slice(0, 5));
  };

  const exportData = () => {
    const data = {
      modules: modules.map(m => ({
        id: m.id,
        angle: m.angle,
        name: m.name
      })),
      sensors,
      systemStatus,
      arcopt: arcoptState,
      bolero: boleroState,
      hyrodyn: hyrodynState,
      mars: marsState,
      mmlf: mmlfState,
      rock: rockState,
      timestamp: new Date().toISOString()
    };

    const blob = new Blob([JSON.stringify(data, null, 2)], { type:
'application/json' });
    const url = URL.createObjectURL(blob);
    const a = document.createElement('a');
    a.href = url;
    a.download = `snake-robot-${Date.now()}.json`;
    a.click();
    addAlert('success', 'Дані експортовано');
  };
```

```
  useEffect(() => {
    const canvas = canvasRef.current;
    if (!canvas) return;

    const ctx = canvas.getContext('2d');
    const rect = canvas.getBoundingClientRect();
    canvas.width = rect.width;
    canvas.height = rect.height;

    ctx.clearRect(0, 0, canvas.width, canvas.height);

    if (showGrid) {
      ctx.strokeStyle = '#1e293b';
      ctx.lineWidth = 1;
      for (let x = 0; x < canvas.width; x += 20) {
        ctx.beginPath();
        ctx.moveTo(x, 0);
        ctx.lineTo(x, canvas.height);
        ctx.stroke();
      }
      for (let y = 0; y < canvas.height; y += 20) {
        ctx.beginPath();
        ctx.moveTo(0, y);
        ctx.lineTo(canvas.width, y);
        ctx.stroke();
      }
    }

    const positions = calculateModulePositions(modules, robotPosition.x,
robotPosition.y);

    if (showTrajectory && trajectoryHistory.length > 1) {
      ctx.strokeStyle = '#8b5cf6';
      ctx.lineWidth = 2;
      ctx.setLineDash([5, 5]);
      ctx.beginPath();
      ctx.moveTo(trajectoryHistory[0].x, trajectoryHistory[0].y);
      trajectoryHistory.forEach(point => {
        ctx.lineTo(point.x, point.y);
```

```
    });
    ctx.stroke();
    ctx.setLineDash([]);
  }

  ctx.strokeStyle = '#3b82f6';
  ctx.lineWidth = 8;
  ctx.lineCap = 'round';
  ctx.lineJoin = 'round';

  const gradient = ctx.createLinearGradient(positions[0].x, positions[0].y,
                                            positions[positions.length-1].x,
                                            positions[positions.length-1].y);
  gradient.addColorStop(0, '#ef4444');
  gradient.addColorStop(0.5, '#3b82f6');
  gradient.addColorStop(1, '#22c55e');
  ctx.strokeStyle = gradient;

  ctx.beginPath();
  ctx.moveTo(positions[0].x, positions[0].y);
  for (let i = 1; i < positions.length; i++) {
    ctx.lineTo(positions[i].x, positions[i].y);
  }
  ctx.stroke();

  positions.forEach((pos, i) => {
    ctx.shadowColor = 'rgba(0, 0, 0, 0.3)';
    ctx.shadowBlur = 10;
    ctx.shadowOffsetX = 2;
    ctx.shadowOffsetY = 2;

    ctx.beginPath();
    ctx.arc(pos.x, pos.y, i === 0 || i === 11 ? MODULE_RADIUS + 5 :
MODULE_RADIUS, 0, Math.PI * 2);

    if (i === 0) {
      ctx.fillStyle = '#ef4444';
    } else if (i === 11) {
      ctx.fillStyle = '#22c55e';
    } else {
```

```
        ctx.fillStyle = modules[i].active ? '#3b82f6' : '#64748b';
      }
      ctx.fill();

      ctx.shadowColor = 'transparent';
      ctx.strokeStyle = isDragging && draggedModule === i ? '#fbbf24' : '#1e293b';
      ctx.lineWidth = isDragging && draggedModule === i ? 3 : 2;
      ctx.stroke();

      ctx.fillStyle = 'white';
      ctx.font = 'bold 12px sans-serif';
      ctx.textAlign = 'center';
      ctx.textBaseline = 'middle';
      ctx.fillText(i.toString(), pos.x, pos.y);

      if (modules[i].temperature > 30) {
        ctx.fillStyle = '#ef4444';
        ctx.font = '10px sans-serif';
        ctx.fillText('🔥', pos.x, pos.y - 25);
      }
    });

    setModules(prev => prev.map((m, i) => ({
      ...m,
      x: positions[i].x,
      y: positions[i].y
    })));

  }, [modules.map(m => m.angle).join(','), isDragging, draggedModule, showGrid, showTrajectory, trajectoryHistory, robotPosition]);

  useEffect(() => {
    if (isRunning) {
      const interval = setInterval(() => {
        setSensors(prev => {
          const newSensors = {
            pressure: prev.pressure + (Math.random() - 0.5) * 0.5,
            humidity: Math.max(0, Math.min(100, prev.humidity + (Math.random() - 0.5) * 2)),
```

```
          temperature: prev.temperature + (Math.random() - 0.5) * 0.3,
          light: Math.max(0, prev.light + (Math.random() - 0.5) * 20),
          distance: Math.max(10, Math.min(300, prev.distance + (Math.random()
- 0.5) * 10)),
          accelerationX: (Math.random() - 0.5) * 2,
          accelerationY: (Math.random() - 0.5) * 2,
          accelerationZ: 9.81 + (Math.random() - 0.5) * 0.5,
          gyroX: (Math.random() - 0.5) * 10,
          gyroY: (Math.random() - 0.5) * 10,
          gyroZ: (Math.random() - 0.5) * 10
        };

        setSensorHistory(prev => ({
          temperature: [...prev.temperature.slice(1), newSensors.temperature],
          distance: [...prev.distance.slice(1), newSensors.distance],
          light: [...prev.light.slice(1), newSensors.light],
          pressure: [...prev.pressure.slice(1), newSensors.pressure],
          humidity: [...prev.humidity.slice(1), newSensors.humidity]
        }));

        return newSensors;
      });

      setPerformance(prev => ({
        cpuUsage: Math.max(20, Math.min(90, prev.cpuUsage + (Math.random()
- 0.5) * 5)),
        memoryUsage: Math.max(30, Math.min(85, prev.memoryUsage +
(Math.random() - 0.5) * 3)),
        networkLatency: Math.max(5, Math.min(50, prev.networkLatency +
(Math.random() - 0.5) * 2)),
        batteryLevel: Math.max(0, prev.batteryLevel - 0.01),
        powerConsumption: Math.max(15, Math.min(35,
prev.powerConsumption + (Math.random() - 0.5) * 1))
      }));

      setModules(prev => prev.map(m => ({
        ...m,
        temperature: Math.max(20, Math.min(40, m.temperature +
(Math.random() - 0.5) * 0.5)),
        velocity: (Math.random() - 0.5) * 5,
        torque: Math.abs(m.angle) * 0.5 + Math.random() * 2
```

```
        }));

        if (modules[0]) {
setTrajectoryHistory(
```