

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ
РОБОТИЗОВАНИМ ПРОТЕЗОМ
ROBOTIC PROSTHESIS CONTROL SOFTWARE

Виконав: студент 4 курсу, групи КНТ-118

Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Івахненко Б.О.

(прізвище та ініціали)

Керівник Пархоменко А.В.

(прізвище та ініціали)

Рецензент Поляков М.О.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Факультет комп'ютерних наук і технологій
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Івахненко Богдана Олексійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для керування
роботизованим протезом. Robotic Prosthesis Control Software

керівник проєкту (роботи) Пархоменко Анжеліка Володимирівна, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “27” квітня 2022 року № 102

2. Строк подання студентом проєкту (роботи) 10 травня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програмного забезпечення. 3. Конструювання програмного забезпечення. 4. Керівництво програміста. 5. Керівництво оператора

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Пархоменко А.В., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “20” лютого 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка структури програми та алгоритму управління	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент

(підпис) Івахненко Б.О.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Пархоменко А.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 92 с., 35 рис., 16 табл., 2 дод., 63 джерела.

РОБОТИЗОВАНИЙ ПРОТЕЗ, АЛГОРИТМ УПРАВЛІННЯ, ARDUINO, СЕРВОПРИВІД, ГРАФІЧНИЙ ІНТЕРФЕЙС

Об'єкт дослідження – процес управління роботизованим протезом верхньої кінцівки людини.

Предмет дослідження – методи та засоби управління роботизованим протезом верхньої кінцівки.

Мета роботи – розробка програмного забезпечення для підвищення ефективності управління роботизованим протезом верхньої кінцівки людини.

Матеріали, методи та технічні засоби: плата Arduino Nano, плата розширення ArduinoSensor Shield, серводвигуни SG90, інтерфейси UART/USART, мова програмування Processing.

Результати. З використанням мови програмування Processing створено програмне забезпечення для управління роботизованим протезом на основі взаємодії Arduino та Processing.

Висновки. Графічний інтерфейс у вигляді Windows-застосунку для керування роботизованим протезом забезпечує інтуїтивно-зрозумілі функції управління та дозволяє підвищити його ефективність.

Галузь використання – при реабілітації людей з втраченими верхніми кінцівками в медичних закладах, а також в якості фізичного прототипу в процесі дослідної експлуатації біомедичних пристроїв.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 92 pages, 35 figures, 16 tables, 2 appendixes, 63 sources.

ROBOTIC PROSTHESIS, CONTROL ALGORITHM, ARDUINO, SERVO DRIVE, GRAPHIC INTERFACE

The object of studying is the process of controlling a robotic prosthesis of the upper limb of a person.

The subject of research are methods and tools for controlling the robotic prosthesis of the upper limb.

The purpose of the work is to develop software to improve the efficiency of controlling a robotic prosthesis of human upper limb.

Materials, methods and technical means: Arduino Nano board, ArduinoSensor Shield expansion board, SG90 servo motors, UART/USART interfaces, Processing programming language.

Results. Using the Processing programming language, a robotic prosthesis control software was created based on the interaction between Arduino and Processing.

Conclusions. A graphical interface in the form of a Windows-based application for robotic prosthesis control provides intuitive control functions and improves control efficiency.

The areas of usage are the rehabilitation of people with lost upper limbs in medical institutions, as well as as a physical prototype in the process of trial operation of biomedical devices.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Дослідження особливостей розробки протезів верхніх кінцівок.....	10
1.2 Огляд біонічних протезів рук.....	14
1.3 Огляд особливостей реалізації біосенсорної системи.....	20
1.4 Аналіз конструктивних та функціональних особливостей роботизованого протезу, створеного на кафедрі ПЗ НУЗП.....	36
1.5 Висновки за розділом 1.....	42
2 Проєктування програмного забезпечення.....	44
2.1 Аналіз вимог.....	44
2.2 Огляд та вибір апаратного інтерфейсу передачі даних.....	45
2.3 Вибір середовища програмної розробки.....	49
2.4 Висновки за розділом 2.....	52
3 Конструювання програмного забезпечення.....	53
3.1 Узагальнений алгоритм управління РП та структура програми...	53
3.2 Опис реалізації програми для плати Arduino.....	54
3.3 Опис програмної реалізації графічного інтерфейсу.....	58
3.4 Висновки за розділом 3.....	63
4 Керівництво програміста.....	64
4.1 Призначення й умови застосування програми.....	64
4.2 Характеристики програми.....	64
4.3 Звернення до програми.....	65
4.4 Вхідні та вихідні дані.....	65
4.5 Повідомлення програмісту.....	65
5 Керівництво оператора.....	66
5.1 Призначення програми.....	66
5.2 Умови виконання програми.....	66

5.3 Робота з програмою.....	66
5.4 Повідомлення оператору.....	67
Висновки.....	68
Перелік джерел посилання.....	69
Додаток А Текст програми.....	75
Додаток Б Слайди презентації.....	86

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ANOVA	– ANalysis of VAriance;
FSR	– Force Sensitive Resistors;
GUI	– Graphical User Interface;
I ² C	– Inter Integrated Circuits;
IDE	– Integrated Development Environment;
SEM	– Standard Errors of Mean;
SPI	– Serial Peripheral Interface;
STRAS	– Single Access and Transluminal Robotic Assistant for Surgeons;
UART	– Universal Asynchronous Receiver-Transmitter;
USART	– Universal Synchronous/Asynchronous Receiver/Transmitter;
USB	– Universal Serial Bus;
МК	– мікроконтролер;
ПЗ	– програмне забезпечення;
РП	– роботизований протез;
РС	– роботизована система.

ВСТУП

Навіть за часів мирного життя велика кількість людей в Україні потребувала протезно-ортопедичної допомоги. Наприклад, за даними профільних підприємств, в 2018 році було близько 95 тисяч пацієнтів, з яких для більше ніж 17 тисяч людей було необхідним протезування кінцівок, зокрема для 15% пацієнтів - протезування верхніх кінцівок [1].

На сьогоднішній день у світі сфера протезування розвивається значними темпами. Розробники сучасних реабілітаційних пристроїв використовують останні досягнення робототехніки, найкращі конструктивні та сервісні матеріали. Сучасний протез – це не просто механічна заміна втраченого органу, це унікальна частина тіла, яка забезпечує йому певні можливості. На даний час найбільш поширеними є функціональні (активні) протези, які на відміну від косметичних, що виконують виключно декоративну функцію, дозволяють їх власникам виконувати різноманітні дії [2].

Біонічні (біоелектричні або міотонічні) протези є найбільш сучасними та багатофункціональними заміниками верхніх кінцівок людини. Управління біонічним протезом здійснюється за рахунок сигналів, що виникають при скороченні м'язів, а далі сприймаються міодатчиками та передаються у мікропроцесор. Комп'ютерні алгоритми перетворюють інформацію в команди та в результаті протез виконує певні рухи або захоплення [2].

Проектування та виробництво біонічних протезів поки що залишається дорогим процесом та потребує подальших наукових досліджень. Тому, актуальною задачею є розробка більш дешевих та доступних пацієнтам роботизованих протезів кінцівок, що надають базовий набір рухів та захоплень, а також можливість простого інтуїтивного тягового управління.

Метою роботи є розробка програмного забезпечення для підвищення ефективності управління роботизованим протезом верхньої кінцівки людини.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Дослідження особливостей розробки протезів верхніх кінцівок

Біоніка - це прикладний напрямок науки про застосування в технічних пристроях та системах принципів організації, якостей, функцій та структур живої природи, тобто дослідження форм живого в природі та створення їх промислових аналогів. Біоніка тісно пов'язана з низкою інших наук — біологією, фізикою, хімією, кібернетикою, електронікою, технологіями передачі даних, тощо [3].

В рамках біоніки проводяться дослідження за багатьма напрямками, зокрема це розробка біонічних імплантатів та протезів. Призначення біонічних протезів – відтворення функцій втрачених кінцівок та інших органів [3].

Процес протезування спрямований саме на заміну втрачених або нефункціонуючих частин тіла людини штучними замінниками – протезами. В загальному випадку існуючі різновиди протезів верхніх кінцівок людини можна поділити на дві категорії – пасивні (косметичні) та активні (робочі або функціональні). В залежності від ступеня ампутації розрізняють протези плеча, передпліччя, кисті, пальців та інші [4].

На сьогоднішній день перевагу віддають розробці та застосуванню саме активних функціональних протезів руки, що повертають повноцінне життя пацієнту. Але для цього протез потрібно не лише механічно приєднати до тіла пацієнта, але й забезпечити управління ним.

Як показали проведені дослідження, можна виділити декілька типів управління протезами (рис. 1.1):

- тягове управління – базується на застосуванні механічних тяг для передачі впливів на рухомі частини протезу [4];
- управління на основі електроміографії (ЕМГ) - базується на зчитуванні біоелектричних потенціалів, що виникають при скороченні м'язів на частині руки, що лишилася [4];

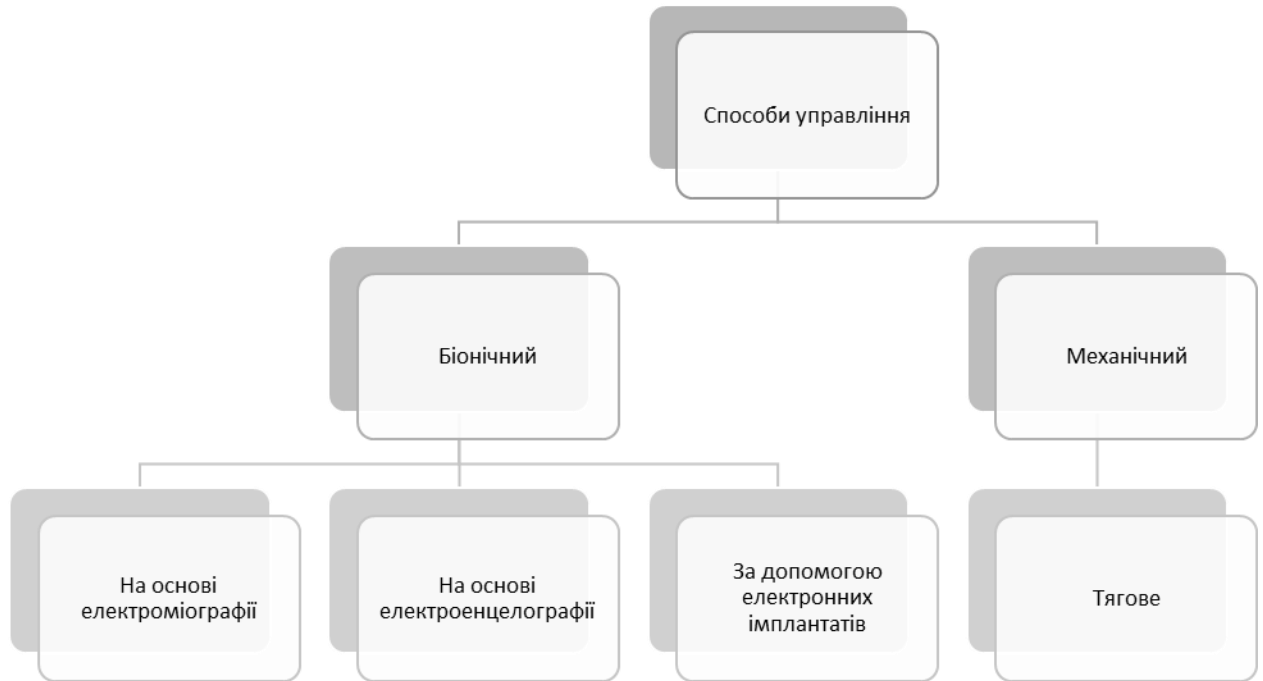


Рисунок 1.1 – Класифікація способів управління протезом

- управління на основі електроенцефалографії (ЕЕГ) – базується на отриманні моторних сигналів шляхом зчитування електричних потенціалів із мозку за допомогою датчиків, що розташовані на поверхні шкіри голови, та надають дані, які декодуються комп'ютером і перетворюються в керуючі команди для біонічного протезу [4];
- управління за допомогою електронних електродів, імплантованих у кору головного мозку, які здійснюють реєстрацію активності коркових нейронів [4].

Для того, щоб підвищити рівень корисного сигналу та позбавитися від шумів, зазвичай для зчитування електричних потенціалів із м'язів застосовують не одну пару електродів, а максимальну їх кількість. Це також дає можливість реалізувати інтелектуальні алгоритми «самонавчання», з метою розробки бази відповідностей сигналів та наступних рухів. Таким чином, кожній комбінації сигналів, що отримано з різних електродів,

відповідатиме визначений рух, який буде виконувати система при отриманні цього набору імпульсів [4].

Перші протези на основі тягового управління (рис. 1.2) забезпечували механічні процеси згинання-розгинання, захоплення на основі обмеженого набору впливів самої людини. Пацієнт мав вручну керувати протезом кінцівки, щоб отримати зворотню реакцію. Навчання користуванню таким протезом потребувало багато часу та зусиль, а результат не завжди був задовільним, особливо щодо забезпечення дрібної моторики [2], [4].

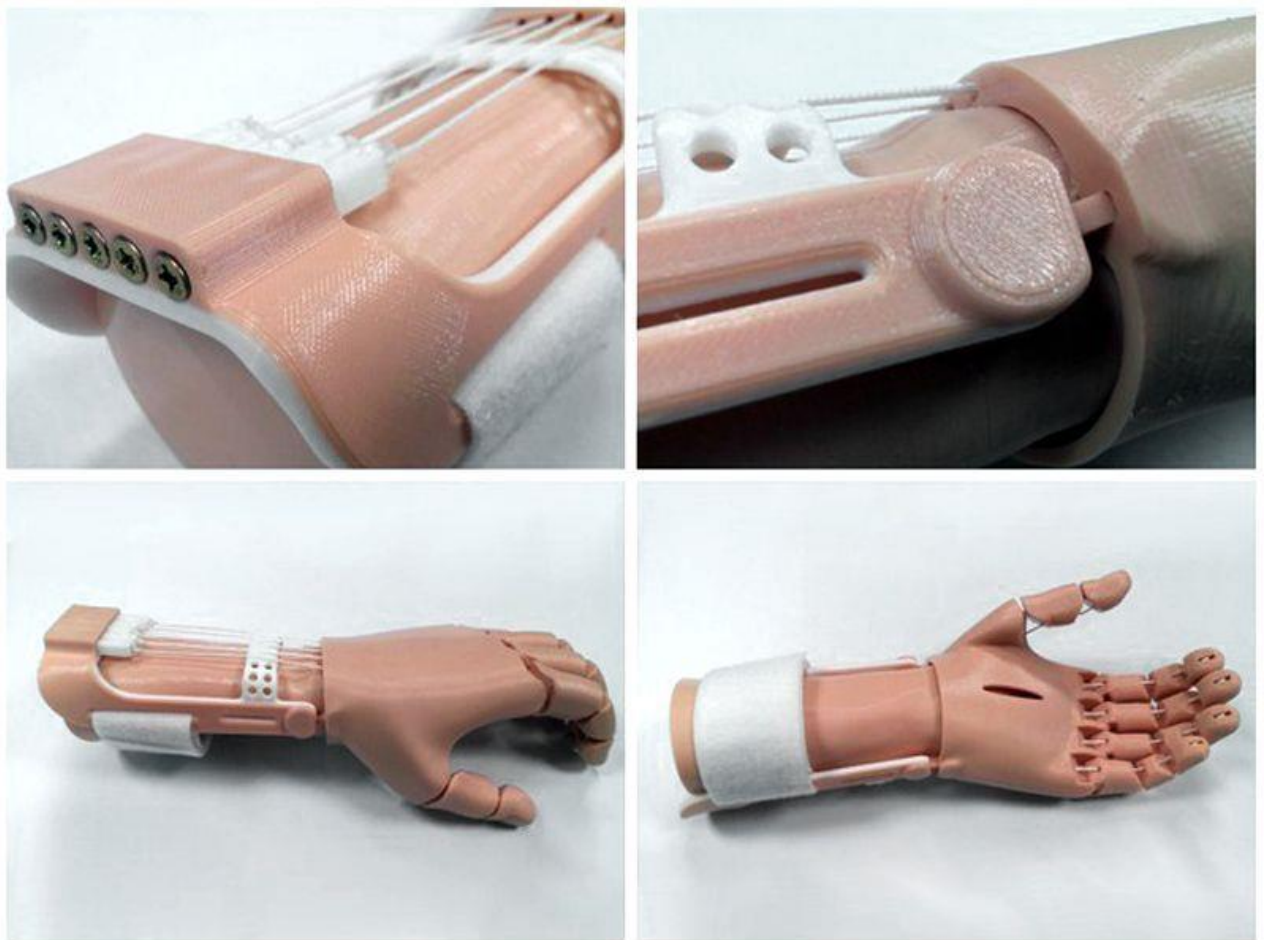


Рисунок 1.2 – Тяговий механічний протез [2]

Сучасні біонічні протези на основі електроміографії - це електронні прилади, якими людина може керувати за допомогою власної нервової системи за рахунок так званої цільової реіннервації м'язів. Оскільки після ампутації кінцівки в тілі залишаються рухові нерви, які її контролювали, то

їх можна хірургічним шляхом приєднати до невеликої ділянки якогось великого м'яза, наприклад, до великого грудного м'яза, у випадку ампутації руки. Цей процес називається реіннервацією [5].

Коли людина подумає, що треба поворушити пальцем, мозок надішле сигнал частині грудного м'яза, до якого приєднали нерв, який був раніше спрямований до пальців. Сигнал фіксується електродами, які надсилають імпульс по дротах у процесор усередині роботизованого протезу (РП) руки. Далі електроміографія дозволяє реєструвати різницю електричних потенціалів, що виникають під час роботи м'яза, фіксує рух реіннервованої частини грудного м'яза, після чого сигнал передається до потрібної частини протеза, і ця частина рухається [4] - [5].

В свою чергу, цільова сенсорна реіннервація дозволяє людині за допомогою протезу відчувати дотик, тепло або тиск. В цьому випадку також працює чутливий нерв, який лишився та був перешитий до ділянки шкіри на грудях пацієнта, а сенсори на протезі передають сигнал від дотику до цієї ділянки шкіри [5]. Схему цільової м'язової реіннервації наведено на рисунку 1.3 [4].

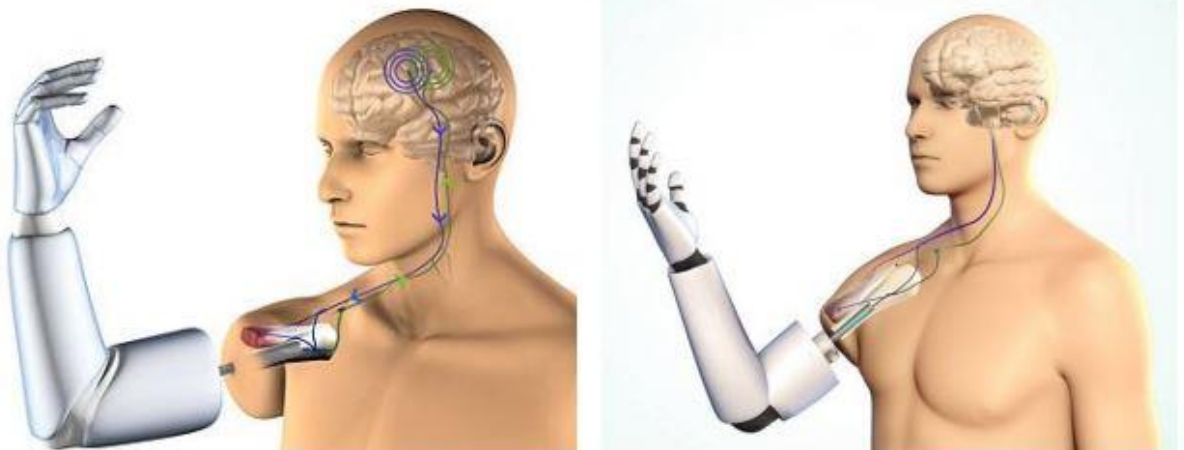


Рисунок 1.3 – Схема цільової м'язової реіннервації [4]

Для того, щоб адаптувати протез під особливості конкретної людини, а також спростити життя пацієнта, потрібно використовувати:

- достатню кількість електродів для отримання більшої вибірки сигналів;
- відповідні алгоритми машинного навчання для аналізу частоти реєстрації різних комбінацій сигналів;
- спеціальне програмне забезпечення (ПЗ) для правильного розпізнавання команд мозку [4].

Зазвичай при реалізації такого підходу вимагаються значні ресурси комп'ютера, оскільки процесору доводиться аналізувати безліч сигналів, тому цю проблему потрібно вирішувати шляхом удосконалення відповідних алгоритмів [4].

Таким чином, на сьогоднішній день існує два основних напрямки розвитку біонічних протезів. Перший – це надання їм чутливості, тобто організація зворотнього зв'язку, який дозволить власнику пристрою отримувати інформацію про якість об'єкта, якого він торкається. Другий – вживлення всіх елементів, включаючи каркас та датчик. Навіть найсучасніші протези необхідно знімати на час сну або прийняття ванни. Після того, як розробники домоглися відповідності протезів оригінальним кінцівкам за виглядом і функціями, залишилося зробити їх постійною частиною людського організму, що не вимагає додаткового догляду [2].

1.2 Огляд біонічних протезів рук

Компанія Touch Bionics пропонує цілу лінійку біонічних протезів рук. Базовою є модель i-Limb Pro, в якій не забезпечується з'єднання протеза з нервами, а лише програмно задано певний набір рухів. На рисунку 1.4, а наведено модель i-Limb Ultra, яка надає набір стандартних жестів і захватів, змогу рухати пальцями окремо, а також регулювати силу стискання, що потрібно в певних ситуаціях. На рисунку 1.4 б, в, г наведено також інші моделі i-Limb: Hand, Quantum, Revolution [4].



а - i-Limb Ultra; б - Hand, в - Quantum, г - Revolution.

Рисунок 1.4 – Моделі біонічних протезів серії i-Limb [4]

Різні модифікації дозволяють людям, що використовують протез, вибирати функціональність біонічної руки, а також тип ПЗ. Біонічний протез руки підтримує керування за допомогою програми для смартфона. Користувач може рухати рукою-протезом, напружуючи певні м'язи, і контролювати рухи пальців, дозволяючи взяти якийсь предмет. Біопротез має високу чутливість, і дозволяє взяти пляшку з водою або аркуш паперу або ж чистити картоплю. Протез має штучну шкіру, що реалістично виглядає [6].

В новій моделі протеза i-LIMB Pulse (рис. 1.5) з'явилися дві інновації: "пульсуюча технологія" та динамічне налаштування роботи біопротезу [6].

Сутність "пульсуючої технології" (pulsing technology) i-LIMB Pulse у тому, щоб подати додаткове зусилля на пальці після того, як людина обхопила якийсь предмет. Це дозволяє добре утримувати взяті предмети та допускає більш тонке маніпулювання ними, наприклад, при зав'язуванні шнурків, відкриванні замка ключем [6].



Рисунок 1.5 – Модель біонічної руки i-Limb Pulse [4]

Для реалістичності протезу руки, розробники почали випускати її у двох розмірах – чоловічому та жіночому та у кількох варіантах оформлення шкіри біоруки. Алюмінієвий каркас зменшує вагу протезу та дає можливість отримувати навантаження на руку до 90 кг [6].

Сутність налаштування роботи біопротезу полягає в тому, що розробники в новій моделі i-LIMB Pulse дали можливість людині самостійно налаштовувати руку-протез через Bluetooth та ПЗ BioSim. Зокрема, налаштовувати зусилля та поріг спрацьовування, вибирати функції та типові захоплення, включати або відключати "пульсуючий ефект". Загалом тепер людина може точно підлаштовувати руку-протез під дії та фізіологічні особливості свого організму [6].

Розробники також не забули і свої потреби – ПЗ дозволяє в реальному часі спостерігати за міоелектричними імпульсами протезу людини, а також збирати статистику роботи протезу. А це у свою чергу дозволить розробникам удосконалювати протез та ПЗ [6].

i-limb ultra revolution – це перший біонічний протез руки, що вийшов у серійне виробництво. Протез здатний відтворювати 24 основних дії кінцівки, включаючи поворот кисті та захоплення пальцями будь-якого предмета відповідно до його габаритів [7].

Міоелектричний протез руки BeBionic (RSLSteeper, Великобританія) здатний обертатися на 135 градусів і згинатися до 35 градусів (рис. 1.6). Як і більшість сучасних протезів, можливості BeBionic максимально наближаються до природних рухів руки людини. Управління здійснюється мікропроцесором, що перетворює в команди для електродвигуна сигнали від датчиків, що розпізнають рухи м'язів кукси, що збереглися. Ще один плюс BeBionic - спеціальне ПЗ, що дозволяє здійснювати різні види захватів предметів та регулювати рівень стиснення пальців [8].

BeBionic Small важить 390 грамів, точно повторює моторику звичайної руки та оснащений 14 сенсорами захоплення для отримання максимально природних рухів. Шарнірний механізм керується крихітними електродвигунами, а на кінчиках пальців є спеціальні повітряні подушечки, щоб акуратно брати дрібні предмети [9] - [10].

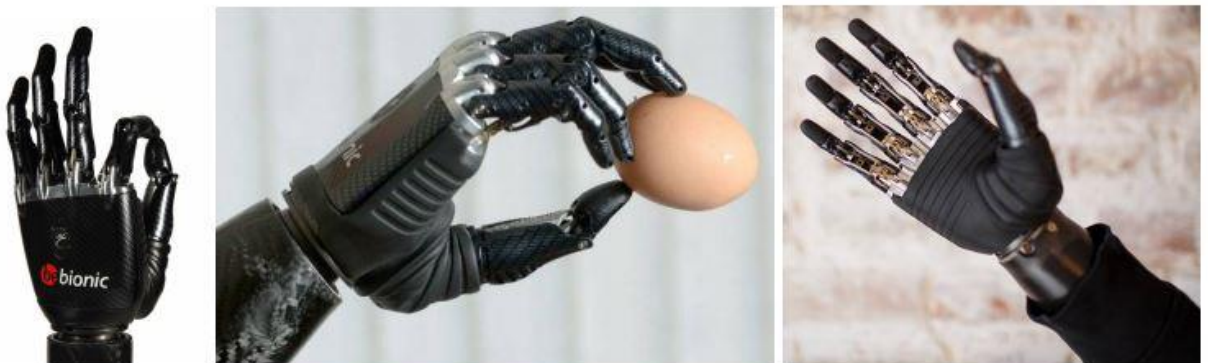


Рисунок 1.6 – Міоелектрична біонічна рука BeBionic 3 [4]

Кисть BeBionic (рис.1.7) – це протез, принцип роботи якого базується на використанні нервових імпульсів пацієнта, що обробляються електронікою та ПЗ, що нараховує 14 підпрограм, з яких 9 є базовими, а решта налаштовуються індивідуально під пацієнта. Мозок людини надсилає імпульс на м'язи. На м'язах стоїть датчик, який зчитує інформацію та передає на процесор, що знаходиться в кисті. При цьому користувач може виконувати не тільки прості завдання, але й досить складні, що вимагають великої точності рухів: утримувати в руці пластиковий стаканчик з водою або навіть застібати гудзики [11].



Рисунок 1.7 – Кисть-протез Bebionic [4]

Робо-рука BionicSoftHand від німецької компанії Festo цікава своєю пластичністю та рухається майже як природна рука. BionicSoftHand виконана з м'яких та гнучких матеріалів, фаланги пальців виконані з тривимірної текстильної тканини з тактильними сенсорами та гнучкими провідниками, які можуть згинатися за допомогою пневматичних модулів. Рухами BionicSoftHand управляє штучний інтелект, здатний навчатися методом спроб і помилок, а після адаптації до завдання швидко та автономно виконувати необхідні дії з предметами, за аналогією як це робить людина [12].

Протез SmartHand, розроблений групою вчених із шести країн (Швеція, Данія, Італія, Ісландія, Ірландія, Ізраїль) використовує для керування нервові закінчення в культі ампутованої руки. Особливістю протезу є те, що він здатний імітувати не тільки рухи руки людини, а й відтворювати відчуття від дотику до об'єкта. Все це реалізується за допомогою чотирьох електродвигунів та 40 датчиків, які активуються при дотику штучними пальцями до об'єкта. Фактично це біонічний протез зі зворотним зв'язком [8].

Вчені з Федеральної політехнічної школи у Лозанні, Швейцарія, також оголосили про створення протезу, який поверне людині дотик. За допомогою електродів кінцівка підключається до двох головних нервів руки – серединного та ліктьового [13].

Попередній варіант протезу використовував дві сенсорні зони, а в останній версії будуть використовуватися тактильні сигнали від долоні, зап'ястя, всіх кінчиків пальців, тобто потрібно використовувати декілька сенсорів (рис. 1.8) [13].



Рисунок 1.8 – Біонічний протез зі зворотним зв'язком [13]

Біонічна рука-протез Handiii [15] компанії «Exiii» (Японія) використовує смартфон як пристрій для обробки даних, які надходять з відповідних датчиків. Сенсори зчитують електричні сигнали з частини руки людини, що лишилася, та інтерпретують їх. Протез дає людині можливість повертати руку, рухати пальцями й утримувати в долоні предмети (рис. 1.9) [4].



Рисунок 1.9 – Біонічний протез руки Handiii японської компанії «Exiii» [4]

Таким чином, як показали проведені дослідження, існує декілька визначальних напрямків подальшого удосконалення системи управління біонічних протезів для отримання пацієнтами максимально природніх рухів та відчуттів. Зокрема, це використання більшої кількості сенсорів для забезпечення зворотнього зв'язку з пацієнтом, удосконалення ПЗ на основі використання алгоритмів штучного інтелекту.

1.3 Огляд особливостей реалізації біосенсорної системи

Сучасні бездротові біосенсорні системи [14] - [15]. можуть перетворювати біологічну реакцію в електричні вимірювання за допомогою електронних схем, а також дозволяють передавати виявлену інформацію дистанційно без використання кабелів або дротів на платформу збору даних [16]. Ці системи забезпечують у режимі реального часу зручний, безперервний моніторинг та аналіз сигналів, включаючи хімічні сигнали, такі як газ і біомолекули, теплові сигнали, такі як лихоманка та гіпотермія, електрофізіологічні сигнали, такі як мозкові хвилі та серцева діяльність і фізичні сигнали, такі як тиск, рух, а також дані про індивідуальні сили зчеплення [17].

Індивідуальні розподіли сили зчеплення, що необхідні для виконання функціональних завдань, мають функціональні надмірності і залежать від типу завдання [18]. Наприклад, коли потрібно маніпулювати циліндричним об'єктом, середній палець відіграватиме важливу роль у створенні великої сили, особливо коли завдання полягає в піднятті важкого предмета. У захоплення сферичних об'єктів внесок у загальну силу захоплення безіменного та маленького пальців є також важливий. Крім того, індивідуальні сили пальців відрізняються за статтю [19] і віком [20], що вказує на зміни в обробці завдань контролю дрібної моторики зі збільшенням віку, імовірно, що викликані труднощами людей старшого віку виконувати необхідну кількість силових дій швидко.

Специфічні характеристики апаратних і програмних компонентів роботизованої системи (РС) детально описані в публікаціях [17], [21] - [25]. Роботизована хірургічна система [26] є прототипом, і на даний момент доступні дані від одного висококваліфікованого експерта (загальна кількість доступних даних близько 116 тис.), одного користувача середньої підготовки (153 тис.) і, для порівняння, одного початківця користувача (171 тис.).

Існують певні проблеми, пов'язані з розбіжностями в даних у конкретному випадку розгортання високоточної сили захоплення людини. Це стосується конкретних статистичних інструментів і показників. Дисперсія даних у біосенсорних мережах, наприклад, мережа датчиків сили захоплення в руці людини (рис. 1.10) безпосередньо пов'язана зі специфічними функціональними відмінностями між місцями розміщення датчиків, які потрібно повністю врахувати, як і врахувати інтерпретацію даних у світлі обмежень конкретного завдання, з одного боку, і біомеханічних обмежень, безпосередньо пов'язаних з рухами рук і пальців під час розгортання сили захоплення, з іншого. Це проливає кардинально інше світло на аналіз дисперсії в порівнянні з випадками, коли дисперсію в даних датчика необхідно усунути або мінімізувати [27].

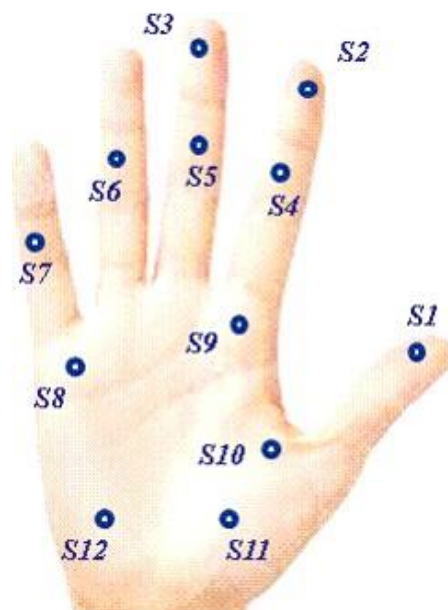


Рисунок 1.10 – Розташування сенсорів на долоні [24]

Потрібен метод, що дозволить виявляти окремі просторово-часові профілі в тисячах даних про змінну силу зчеплення, зосереджуючи увагу на певних місцях пальців. Це може допомогти зрозуміти специфічні відмінності в розгортанні зусиль захоплення між висококваліфікованим оператором, стажером і новачком, і як ці анатомічно специфічні та відповідні завданням сили захоплення розвиваються з часом і рівнем підготовки.

Автори [26], [28] описують експеримент, що проводився на прикладі використання ендоскопічної системи STRAS (Single Access and Transluminal Robotic Assistant for Surgeons), з метою оптимальної допомоги хірургам у мініінвазивних процедурах. Вона призначена для дворучного втручання, а на дистальній стороні роботизованого ендоскопа прикріплена ендоскопічна камера, яка знімає користувачів, які виконують завдання. Під час маніпуляції з системою ендоскопа STRAS, як показано на рисунку 1.11, користувач може тримати дві ручки джойстика для переміщення та керування інструментами ендоскопа в кожній руці, дивлячись на екран монітора, який відображає вигляд камери.



Рисунок 1.11 - Маніпуляції з ендоскопом STRAS [26]

Була розроблена пара спеціальних сенсорних рукавичок для бездротових датчиків, які можна носити [21] - [22], з використанням вбудованих чутливих до сили резисторів (Force Sensitive Resistors, FSR). Кожен з невеликих (діаметром 5–10 мм) FSR резисторів був припаяний до підтягуючих резисторів 10КОм, щоб створити дільник напруги. Напруга, зчитувана аналоговим входом Arduino, визначається рівнянням (1.1):

$$V_{out} = R_{PD} V_{3.3} / (R_{PD} + R_{FSR}), \quad (1.1)$$

де R_{PD} - опір підтягуючого резистора;

R_{FSR} - опір FSR резистора;

$V_{3.3}$ – напруга живлення 3,3 В.

Опір FSR резистора може змінюватися від 250 Ом при впливі 20 Ньютон приблизно до 10 МОм, коли сила не застосовується взагалі. Напруга монотонно змінюється від 0 до 3,22 В, як функція прикладеної сили, яка вважається рівномірною на поверхні датчика. У описаних дослідженнях прикладені сили не перевищували 10 Н; напруги змінювалися в межах від 0 до 1500 мВ. Відношення між силою та напругою є майже лінійним у цьому діапазоні. Було забезпечено, що всі датчики мають подібні калібрувальні криві. Ці FSR резистори спочатку були приклеєні всередині рукавички, а потім закріплені, пришивши круглий шматок тканини навколо. Для кожної рукавички, яка показана на рисунку 1.12, використовується дванадцять анатомічно відповідних FSR резисторів для вимірювання сили зчеплення, прикладеної до певних місць на пальці та на долоні, як було показано вище на рисунку 1.10.

Програмне забезпечення для отримання даних про силу зчеплення включає дві частини: одна працює на платі Arduino Micro, що вбудована в рукавичку, а інша працює на комп'ютері для збору даних.



Рисунок 1.12 - Поверхня сенсорної рукавички [21] - [22]

На рисунку 1.13 показано цей процес збору даних на прикладі датчика 10 (S10). Плата Мікро живиться від Li-Po акумулятора, забезпечує регульовану напругу, отримує аналоговий вихід напруги від кожного датчика FSR, який об'єднується з мітками часу та ідентифікацією датчика. Потім цей пакет даних надсилається на комп'ютер по бездротовому зв'язку через Bluetooth з частотою 50 Гц, декодується програмним забезпеченням комп'ютера та зберігається у файлі Excel.

Для дослідження сили зчеплення було розроблено завдання з використанням платформи STRAS і сенсорної рукавички. Експеримент полягав в маніпулюванні ендоскопічною системою STRAS рукою, на яку одягнуто сенсорну рукавичку. Завдання точного захоплення, що повністю керується зображенням [17], [21] - [25], виконувалося висококваліфікованим експертом, середньо підготовленим користувачем і повним новачком. Експерт є лівшою, а навчений користувач і новачок — правшою, як показано в таблиці 1.1. Лівий і правий інтерфейси ідентичні, і це завдання вибору і скидання було реалізовано кожною рукою протягом десяти послідовних сеансів.

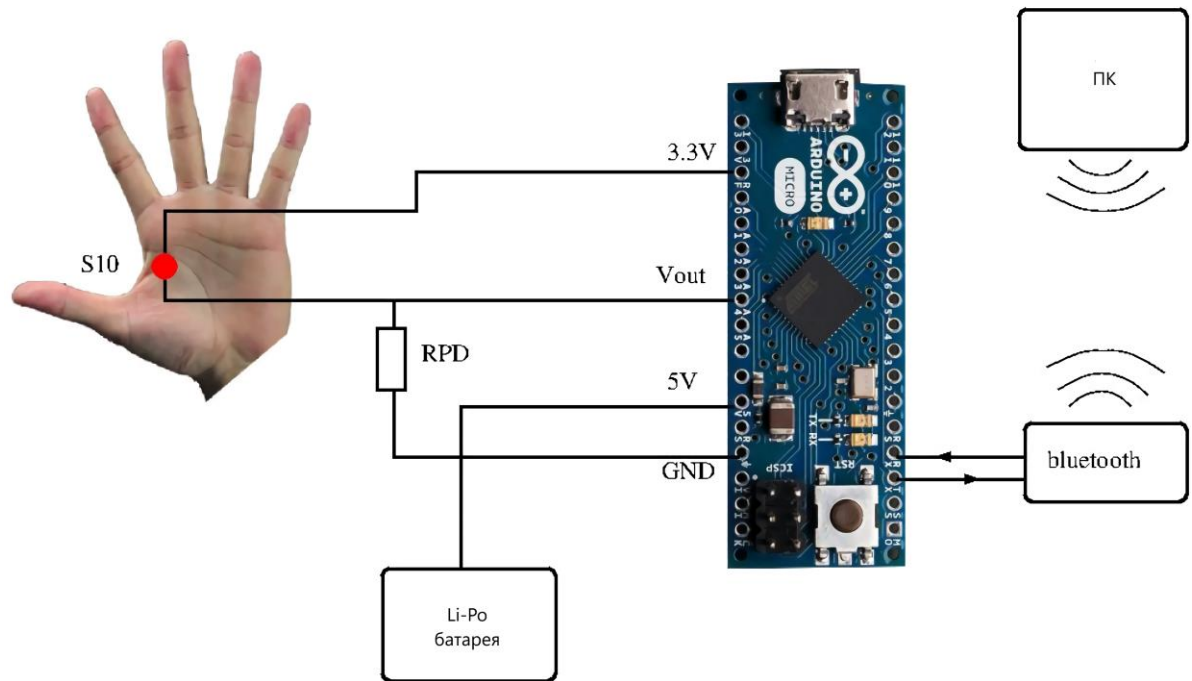


Рисунок 1.13 – Схема системи збору даних [23]

Таблиця 1.1 – Інформація про учасників експерименту

Користувач	Рука, що домінує
Висококваліфікований експерт	Ліва
Користувач середнього рівня підготовки	Права
Повний новачок	Права

Специфіка завдання точного захоплення в даному випадку тут, з одного боку, може бути описана тим, що ступені свободи для рухів кисті та пальців надзвичайно обмежені. Щоб керувати циліндричними ручками, щоб направляти хірургічні інструменти, а також відкривати та закривати захвати на кінцях інструменту під час хірургічного завдання, можливі лише бокові та рухи вперед/назад циліндричних ручок для маніпулювання рухами інструменту у всіх напрямки в тривимірному робочому просторі. Це кардинально відрізняється від захоплення циліндричних предметів, щоб піднімати та/або переміщати їх прямо й вільно у всіх можливих напрямках. З іншого боку, будь-яке хірургічне завдання, що виконується за допомогою системи, включаючи задачу симулятора, яка використовується для навчання,

накладає додаткові обмеження щодо конкретних кроків завдання. Вони повинні виконуватися один за одним у точному порядку з найменшими зусиллями, найбільшою точністю та якомога швидше [29] - [34].

Експеримент складався з чотирьох важливих кроків. На рисунку 1.14 показано знімки цих чотирьох кроків із відеопослідовності, знятої ендоскопічною камерою. В таблиці 1.2 показано, що для виконання цієї чотириетапної задачі точного захоплення користувач спочатку активує та переміщує дистальний інструмент до об'єкта, ефективно маніпулюючи ручками. Під час цього кроку потрібне переміщення по глибині в площині 2D зображення. Коли інструмент прибуде до місця розташування об'єкта, захвати відкриються щоб міцно захопити та підняти об'єкт. На третьому кроці користувач повинен перемістити інструмент з об'єктом до цільового положення, під час якого потрібне лише бічне переміщення. На останньому кроці користувачеві необхідно відкрити захвати та скинути предмет [29] - [34].



Рисунок 1.14 – Знімки чотирьох послідовних кроків експерименту [22]

Таблиця 1.2 – Опис етапів завдання

Етап	Опис
1	Активуйте та перемістіть інструмент до розташування об'єкта
2	Відкрийте та закрийте захвати, щоб захопити та підняти предмет
3	Перемістіть інструмент з об'єктом у цільове місце
4	Відкрийте захвати, щоб опустити предмет у коробку

Оскільки завдання тут повністю керується зображеннями, перший крок завдання тут є найскладнішим, особливо для новачків, оскільки він

вимагає переміщення підказок інструментів вперед по глибині вздовж невидимої осі z в площині двовимірного зображення. Навіть експерти все ще мають проблеми з адаптацією до цієї проблеми, яка може бути досить складною залежно від типу камери та системи зображення, доступної хірургу на певній платформі. Тут, у нашому випадку, для виконання кроків завдання за оптимальний час виконання завдання та з максимальною точністю (тобто без корекції траєкторії інструменту, без інцидентів) потрібно не тільки знати проекцію 2D-зображення тривимірного робочого простору завдання, але й вміло маніпулювати відповідним завданням лівою чи правою ручкою РС [17], [21] - [25].

Завдання вибору було реалізовано кожною рукою трьох користувачів протягом десяти послідовних сеансів, що означає, що одне й те саме завдання повторювалося шістдесят разів, загалом було записано 60 результатів часу разом при цьому 440, 412 сигналів сили захоплення було зібрано з дванадцяти датчиків на домінуючих і недомінуючих руках трьох користувачів у десяти послідовних сеансах виконання завдань, що відповідає загалом 36 701 сигналам сили захоплення на датчик, як показано в таблиці 1.3. Різні рівні досвіду послідовно відображаються параметрами завдання продуктивності, такими як середня тривалість сеансу завдання або кількість інцидентів із завданням у вигляді падінь об'єкта, промахів та коригування траєкторії інструментів під час виконання окремих сеансів завдання.

Таблиця 1.3 – Кількість сигналів сили захоплення для кожного датчика

Користувач	Домінуюча рука	Недомінуюча рука
Висококваліфікований експерт	4442	5244
Користувач середнього рівня підготовки	5974	6764
Повний новачок	7780	6479

Лівий і правий системні інтерфейси ідентичні, і одне і те ж завдання виконується будь-якою рукою. Індивідуальні сили зчеплення централізовано керуються в мозку людини і спрямовані на оптимізацію рухової роботи та контролю [34] - [38].

Існує три рівні коефіцієнта користувача та два рівні фактора зручності. Для того, щоб отримати доступ до впливу цих двох факторів на час виконання завдання, було проведено двосторонній аналіз дисперсій (ANalysis Of VAriance, ANOVA). Як показано в таблиці 1.4, існує значна різниця між трьома користувачами, тоді як між домінуючими та недомінуючими руками немає суттєвої різниці. Більше того, останній рядок ілюструє значну взаємодію між користувачем і факторами зручності, що означає тут, що час виконання різних користувачів залежить від того, якою рукою вони користуються для виконання завдання точності.

Таблиця 1.4 – Результати ANOVA аналізу даних часу як функції користувача та зручності

Джерело варіації	Ступінь свободи	F	P
Користувач	2	15,65	< 0,001
Зручність	1	0,09	Несуттєве
Зручність/Користувач	2	4,13	< 0,05

Результати часу для різних сеансів наведені в термінах середніх та їхніх Standard Errors of Mean (SEM) у таблиці 1.5. На основі таблиці 1.5 видно, що експерт і домінуюча рука середнього користувача мають кращу продуктивність на основі результатів часу, з меншими середніми і варіаціями. Більше того, початківець користувач повільніший і менш стабільний зі своєю домінуючою рукою, що вказує на те, що в такому надзвичайно обмеженому середовищі точних завдань початківець користувач більше нерішуче маніпулював своєю домінуючою рукою.

Таблиця 1.5 – Середнє значення та SEM часу виконання завдання за сеансами

Користувач	Зручність	Середнє значення	Помилка SEM
Висококваліфікований експерт	Домінант	8,88	0,36
	Не домінант	10,49	0,49
Користувач середнього рівня підготовки	Домінант	11,95	0,49
	Не домінант	13,53	0,66
Повний новачок	Домінант	15,56	1,60
	Не домінант	12,99	0,75

Локальні сили зчеплення, розгорнуті в кожному місці датчика, змінюються залежно від того, наскільки вправним став користувач у точному виконанні завдання симулятора з роботою. Дані про силу зчеплення були зібрані від трьох користувачів із чітко різними рівнями досвіду виконання завдань, при чому даних про силу набагато більше, ніж про час. Загальні зусилля, прикладені до кожного датчика протягом десяти сеансів кожною рукою, розраховуються на основі вихідних необроблених даних і наводяться в таблиці 1.6. Згідно з цією таблицею, для різних користувачів і рук застосовувалися різні силові стратегії. Новачок застосував значно більшу загальну силу в порівнянні з двома іншими користувачами, особливо з домінуючою рукою. Крім того, деякі датчики не використовувалися під час певних маневрів. Наприклад, експерт не активував S1, S8 і S11 своєю домінуючою рукою, а S2 і S3 не використовував його не домінуючою рукою.

У випадку силових результатів є три рівні коефіцієнта користувача, два рівні коефіцієнта ручності та дванадцять рівнів фактора датчика. Тресторонній ANOVA дозволяє отримати доступ до впливу цих трьох факторів на результати сили.

Таблиця 1.6 – Загальна сила за сеансами для кожного датчика

Датчик	Експерт D	Експерт N	Середній D	Середній N	Новачок D	Новачок N
1	0	3,40	0	0,00	0	0
2	6,23	0	9.16	58.67	193.23	447.29
3	10.96	0	0	0	5328.26	0
4	9.03	46.90	37.50	0.60	6.07	0
5	437.13	1811.11	2901.79	283.75	5946.81	1926.19
6	2009.06	1895.47	3327.50	3520.78	3915.37	6910.31
7	2607.76	115.71	60.63	3638.08	664.06	3420.98
8	0	487.50	1064.15	38.27	5022.85	1512.46
9	2.50	534.02	786.74	0	8838.14	0.66
10	2106.27	900.44	489.66	499.17	5062.52	3246.43
11	0	3966.70	0	0	6842.42	0
12	5.15	2242.13	1593.11	0	6585.59	2.71
13	7194.10	12,003.39	10,270.23	8039.33	48,405.31	17,467.03
Усього	10.96	0	0	0	5328.26	0

Як показано в таблиці 1.7, існує надзвичайно значна різниця у силі, що застосовується між трьома користувачами, між двома руками та між дванадцятьма датчиками, і існує надзвичайно значна взаємодія між кожними двома факторами. Відповідні засоби та SEM наведені в таблиці 1.8.

Таблиця 1.7 – Результати тристороннього аналізу даних про силу як функцію користувача, зручності та датчика

Джерело варіації	Ступінь свободи	F	P
Користувач	2	107.72	<0.001
Зручність	1	38.27	<0.001
Датчик	11	41.88	<0.001
Користувач/Зручність	2	47.83	<0.001
Користувач/Датчик	22	7,41	<0.001
Зручність/Датчик	11	9,85	<0.001

Таблиця 1.8 – Середнє значення та помилка для всіх датчиків

Фактор	Рівень	Середнє значення	Помилка SEM
Користувач	Експерт	161,96	15,85
	Середній	120,29	13,10
	Новачок	370,55	25,49
Зручність	Домінантний	263,74	17,93
	Недомінантний	171,46	14,48
Датчик	S1	0,12	0,07
	S2	17,6	3,6
	S3	115,7	34,08
	S4	3,05	0,54
	S5	337,63	32,69
	S6	575,04	30,21
	S7	294,61	33,76
	S8	193,40	31,19
	S9	223,82	54,57
	S10	320,96	34,51
	S11	267,99	56,54
	S12	261,28	44,41

Крім того, середні сили в різних категоріях також наведені на рисунку 1.15.

Дані сили зчеплення від трьох конкретних датчиків, а саме S5, S6 і S7, які показані на рисунку 1.16 та описані в таблиці 1.9, потребують детального аналізу, оскільки вони є функціонально репрезентативними згідно з дослідженнями [37] - [38].

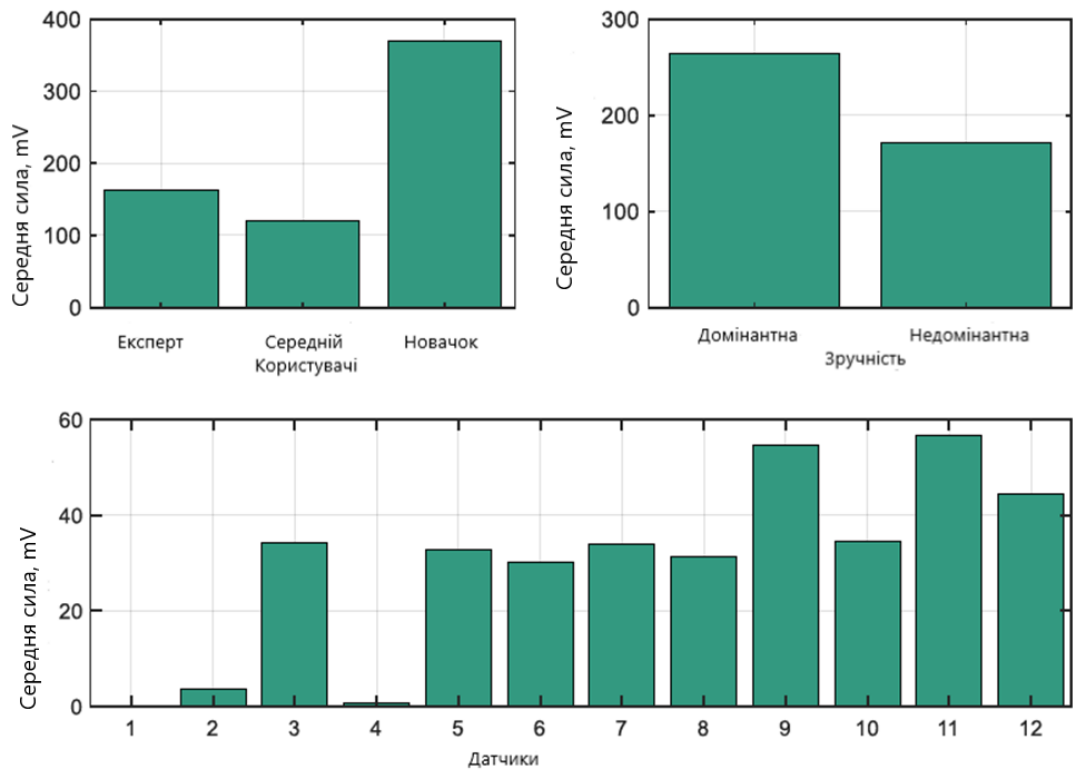


Рисунок 1.15 - Результуючі сили для трьох користувачів з домінуючими та недомінуючими руками для десяти сеансів [37] - [38]

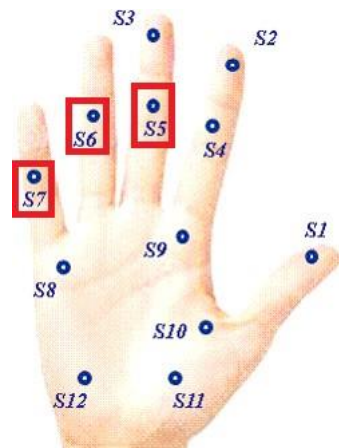


Рисунок 1.16 – Розташування трьох основних датчиків [24]

Таблиця 1.9 – Опис трьох основних датчиків

Датчик	Палець	Роль в контролі сили захоплення
S5	Середній	Розгортання сили зчеплення
S6	Безіменний	Немає значущої ролі в контролі сили захоплення
S7	Мізинець	Точний контроль сили захоплення

Датчик S5, який розташовується на середній фаланзі середнього пальця, здебільшого сприяє розгортанню великої сили, наприклад, підйому важких предметів, але марний для виконання точних завдань. Навпаки, S7 знаходиться на середній фаланзі мізинця і критично важливий для маніпуляції з контролем сили захоплення, що вивчається тут. Крім того, останній датчик S6, розташований на середній фаланзі безіменного пальця, є одним з найменш важливих у контролі сили захоплення при виконанні різноманітних завдань.

Виходячи з попереднього аналізу в цьому підрозділі, найбільш чітка різниця спостерігається між домінуючими руками експерта та новачка. Відповідно, двосторонній аналіз ANOVA був проведений на основі необроблених даних сили зчеплення цих двох рук під час їх першого та останнього сеансів виконання завдань, що показано в таблиці 1.10.

Статистичне порівняння показує значну взаємодію між дворівневим користувачем і факторами сеансу для всіх трьох датчиків, що тут розглядаються. Щоб забезпечити більш пряме порівняння, окремі просторово-часові профілі сили зчеплення були нанесені на графік з точки зору середньої пікової амплітуди ворога кожні сто сигналів для датчиків S5, S6 і S7 на рисунку 1.17 разом з відносною тривалістю кожного з чотирьох кроків завдання, використовуючи кольорові лінії.

Одна з основних відмінностей між кваліфікованими або досвідченими операторами та початківцями стосується пропорційної великої сили зчеплення, що розгортається середнім пальцем.

Оператори-початківці використовують занадто багато непотрібних, нерелевантних завдань великої сили захоплення, тоді як експерт навчився вміло мінімізувати їх [25]. З іншого боку, зусилля точного захоплення, які в основному розгортаються мізинцем і є особливо важливими при хірургічних завданнях, зазвичай недостатньо розгортаються новачками [25]. Безіменний палець не відіграє великої ролі в контролі сили зчеплення, і можна очікувати,

що відмінності між профілями сили захоплення безіменного пальця новачків і експертів будуть мінімальними.

Таблиця 1.10 – Двосторонній ANOVA на трьох датчиках на домінантній руці експерта та новачка для першого та останнього сеансу

Датчик	Сесія	Експерт	Новачок	Взаємодія
S5	Перша	240.37 /4.56	790.00 /3.02	F(1,3120) = 169.39; $p < 0.001$
	Остання	48.32 /0.3	691.72 /2.19	
S6	Перша	575.63 /4.51	504.12 /2.42	F(1,3120) = 394.24; $p < 0.001$
	Остання	473.98 /5.17	540.30 /2.23	
S7	Перша	594.02 /3.41	110.82 /0.75	F(1,3120) = 260.72; $p < 0.001$
	Остання	608.51 /2.38	72.90 /0.61	

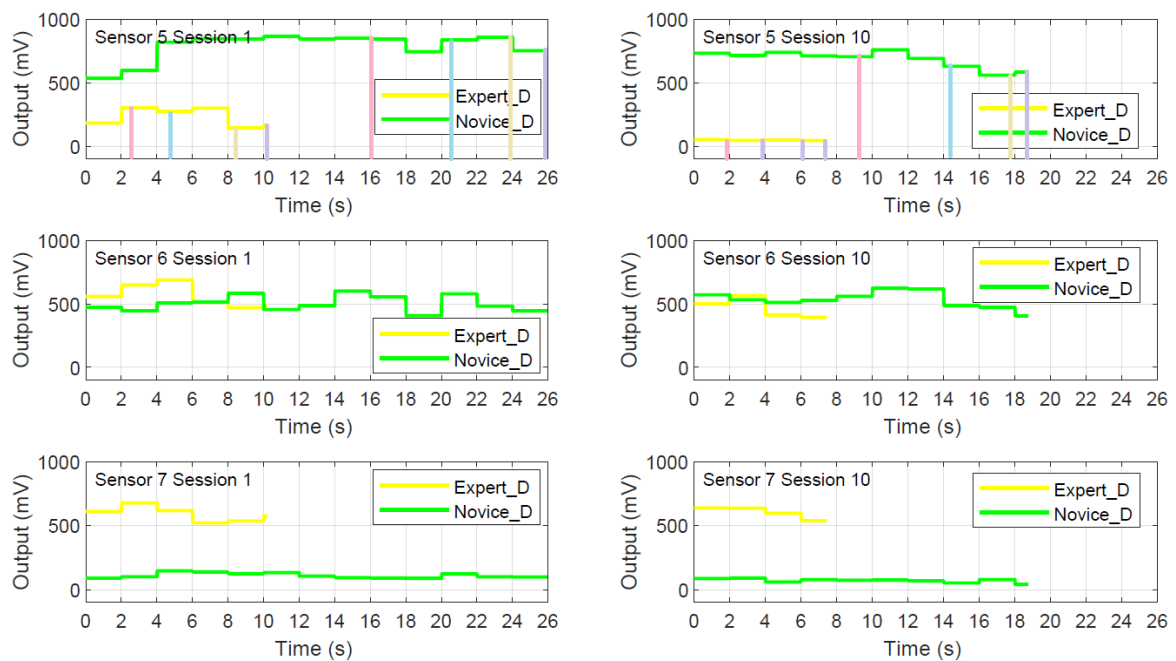


Рисунок 1.17 – Середній пік трьох датчиків на домінантній руці експерта та новачка за перший і останній сеанс [25]

Аналіз просторово-часового профілю показує, що новачкові для виконання завдання потрібно вдвічі більше часу, ніж експерту, але в кінці він отримує 30% приріст часу, що вказує на значний часовий тренувальний ефект. Цей ефект здебільшого стосується першого критичного кроку на рисунку 1.15 завдання «підбирай і кидай» і стає зрозумілим лише у світлі наданого тут конкретного аналізу. Функціональна інтерпретація цього ефекту пов'язана зі специфікою переміщення інструменту від тіла в периперсональному просторі хірурга, необхідного для виконання цього першого кроку завдання. У конкретній системі завдання користувача, що керується зображенням, зусилля зчеплення та рухи рук обмежені обмеженими ступенями свободи РС, з одного боку, і перцептивним відновленням фізично відсутньої інформації про глибину [25], тут уздовж віртуальної вісь z у площині двовимірного зображення необхідна. Така складність призводить до збільшення часу виконання завдання та неточності переміщення інструментів, як показано в роботах [17], [21] - [25].

Таким чином, у цьому підрозділі наведено просторово-часовий статистичний аналіз даних про силу зчеплення, зібраних від трьох різних людей, які носять бездротові носимі сенсорні рукавички під час маніпуляції на платформі ендоскопа, щоб виконати точно чотириетапне завдання підбору й скидання. Згідно з результатами, різні користувачі застосовують значну різну стратегію сили зчеплення залежно від рівня їхньої підготовки та досвіду виконання завдань. Моніторинг сили зчеплення можна запускати в режимі реального часу під час виконання завдання, що може генерувати корисну інформацію для досліджень щодо прийняття клінічних рішень. Наприклад, це може допомогти запобігти ризикам під час роботи [39], де надмірна сила захоплення може призвести до пошкодження тканин. У майбутньому, доцільно провести подальше вивчення ефекту тренування та розвитку силового шаблону, що буде корисним студентам-хірургам, які навчаються [40], розробникам екзоскелетів [41], проєктувальникам реабілітаційних роботів [42] та розробникам тактильних Інтернет-технологій

[43] - [45]. Крім того, потрібно створювати нову систему рукавичок з більш досконалими датчиками зчеплення [46], які менші та гнучкіші. Також, існує очевидна потреба в додаткових дослідженнях динамічних вимірювань сили зчеплення, які враховують комплекс кисті та зап'ястя при вимірюванні динамічної сили [47] - [48]. Комплекс кисті та зап'ястя особливо важливий для лапароскопічних та ендоскопічних хірургічних завдань, оскільки кут рухів зап'ястя безпосередньо впливає на силу зчеплення кисті та пальців у залежності від діаметра інструмента, який захоплюється і яким маніпулюють. Тоді сили зчеплення також залежать від розміру руки [49]. Подальше розуміння цих функціональних взаємозв'язків має бути корисним для розробки ручок, які вимагають захоплення в певних напрямках, щоб зменшити необхідні зусилля та мінімізувати втому та навантаження хірургів.

1.4 Аналіз конструктивних та функціональних особливостей роботизованого протезу, створеного на кафедрі ПЗ НУЗП

РП верхньої кінцівки людини (рис. 1.18) об'єднує механічну, електронну та програмну частини [50] - [51].

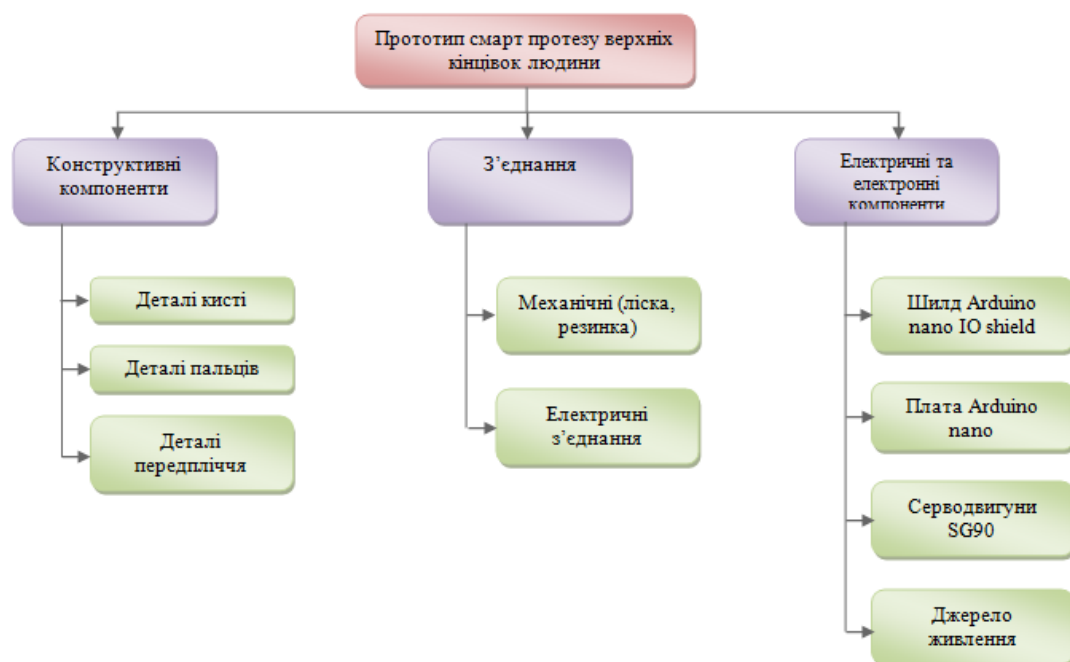


Рисунок 1.18 – Структура РП

Механічну частину РП розроблено в середовищі PTC Creo (рис. 1.19) та роздруковано з використанням технологій 3D-принтіngu. Для підготовки до тривимірного друку використовувалося ПЗ Simplify3D (рис. 1.20) та 3D-принтер Leapfrog CreatrHS (рис. 1.21) [50] - [51].

Електронна частина прототипу включає наступні складові:

- апаратно-програмна платформа Arduino Nano;
- шилд (плата розширення) Arduino Nano Shield V3.0;
- серводвигуни Tower Pro 9g SG90.

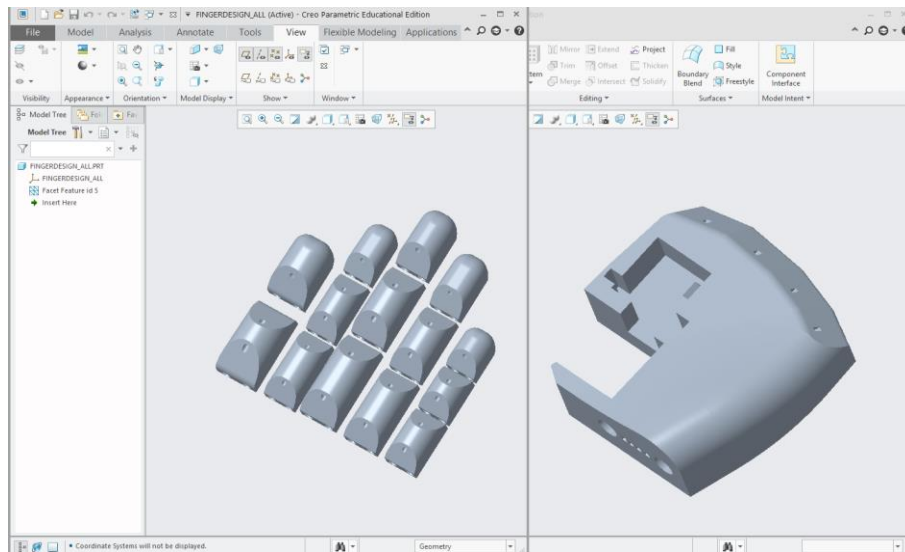


Рисунок 1.19 – Проектування механічної частини в середовищі PTC Creo [51]

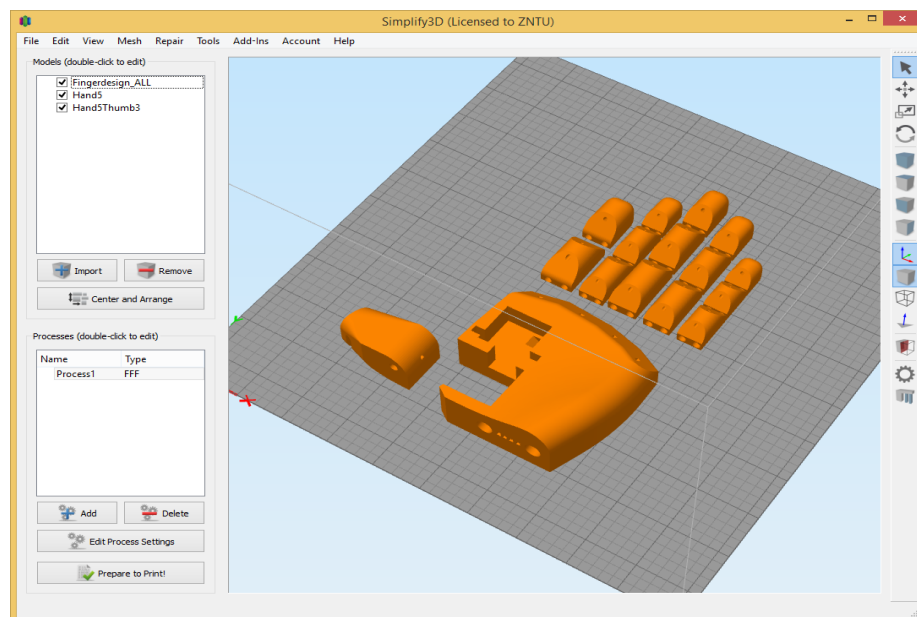


Рисунок 1.20 – Підготовка до друку механічної частини РП в середовищі Simplify3D [51]

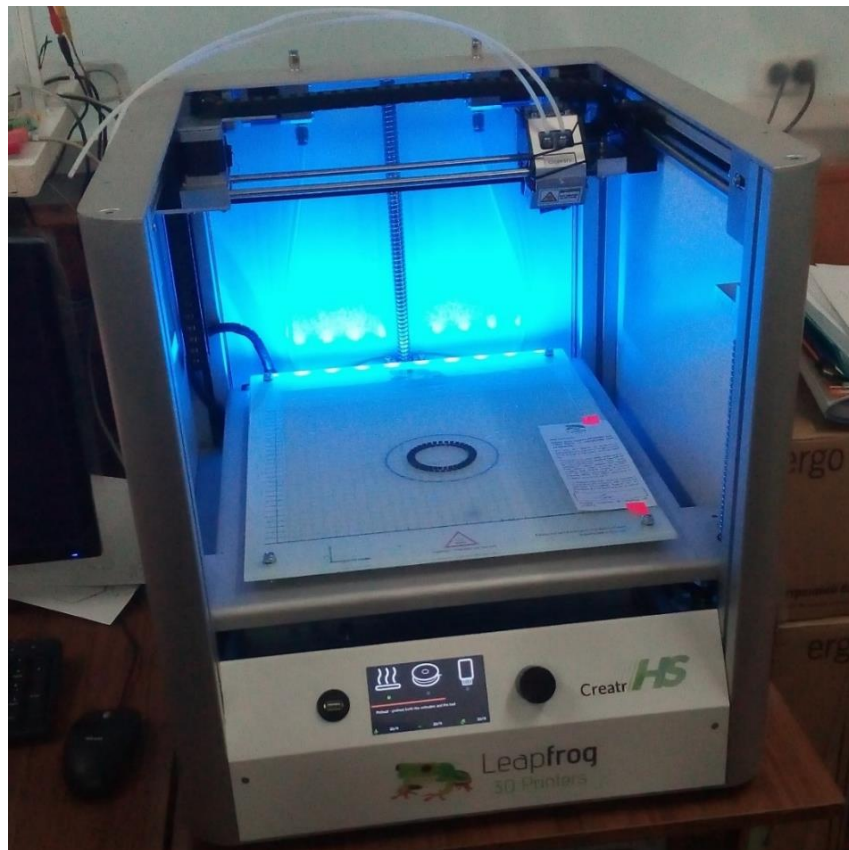


Рисунок 1.21 – Leapfrog Creatr HS [50]

Для того, щоб привести в дію пальці РП використовуються серводвигуни (рис. 1.22).



Рисунок 1.22 – Закріплення сервоприводів на РП [51]

Кожному пальцю на руці відповідає свій серводвигун, що певним чином під'єднаний до різних пінів на платі. На рисунку 1.23 пронумеровано

кожний з серводвигунів, а в таблиці 1.11 розписано виводи плати Arduino Nano, що їм відповідають.

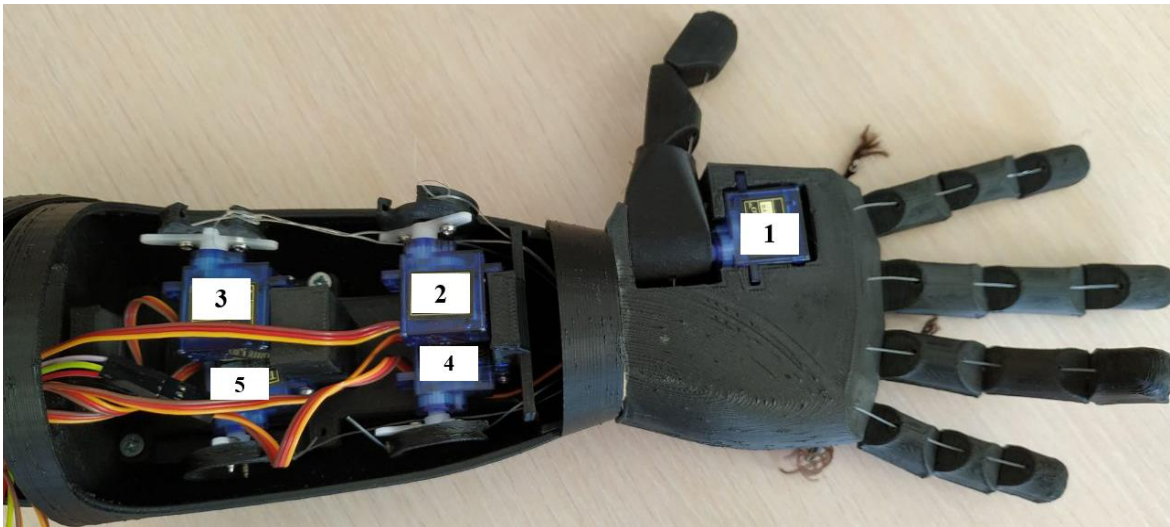


Рисунок 1.23 – Нумерація серводвигунів

Таблиця 1.11 – Відповідність серводвигунів пінам на платі Arduino Nano

Номер серводвигуна	Палець	Номер піна
1	Великий	pin 8
2	Перша фаланга великого пальця	pin 11
3	Вказівний та середній	pin 9
4	Безіменний	pin 10
5	Мізинець	pin 12

На рисунку 1.24 показано узагальнену схему підключення серводвигунів до Arduino Nano.

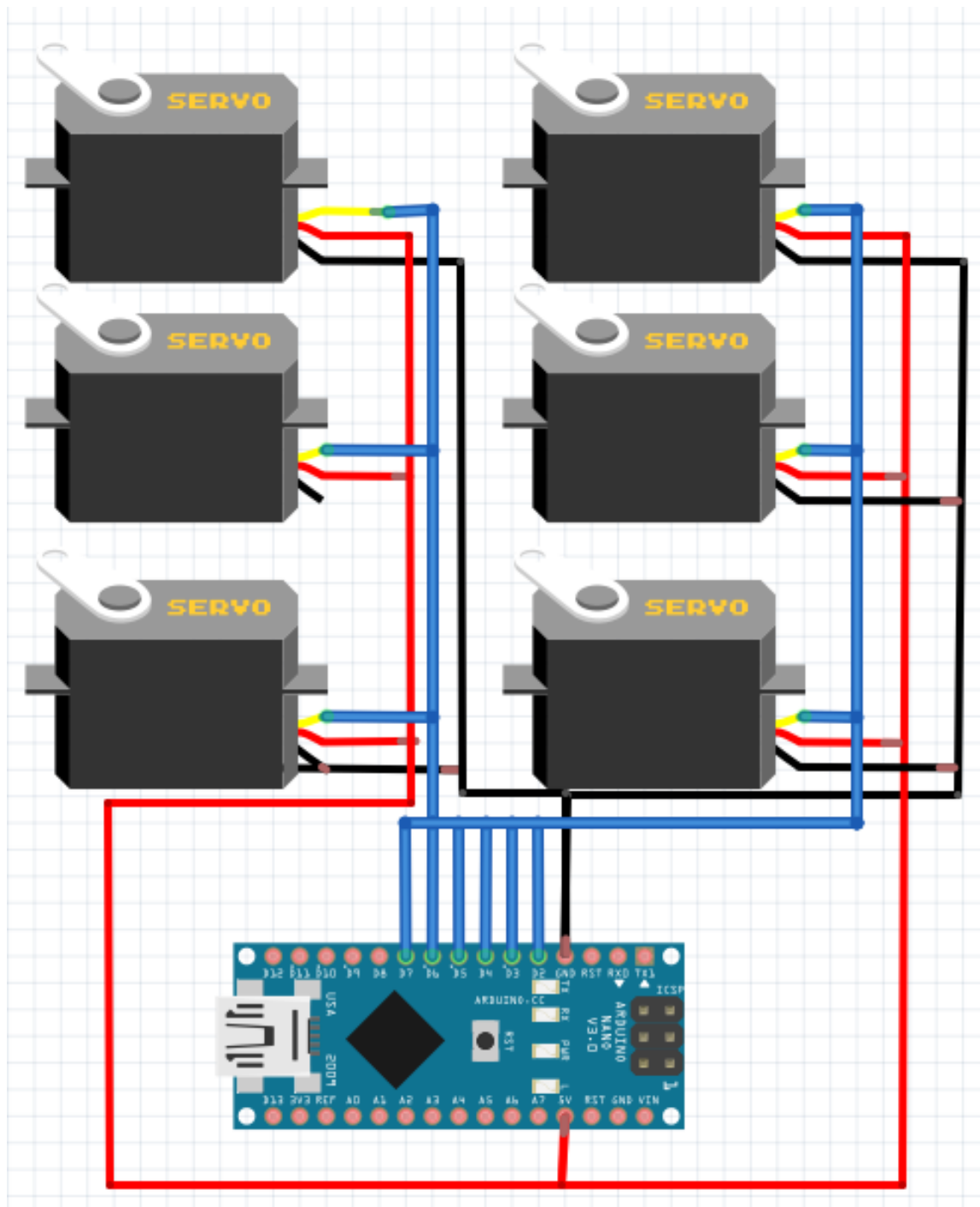


Рисунок 1.24 – Узагальнена схема підключення

Кожен серводвигун має три контакти, які пофарбовані в різні кольори. Функціональне призначення їх наступне:

- земля - коричневий колір;
- живлення +5 В - червоний колір;
- сигнальний - жовтий колір.

Сигнальний дріт підключається до цифрового піну, два інших – до землі та живлення відповідно. Плату Arduino Nano встановлено на плату розширення Arduino Nano Shield V3.0 (рис. 1.25), до якої підключені серводвигуни (рис.1.26).

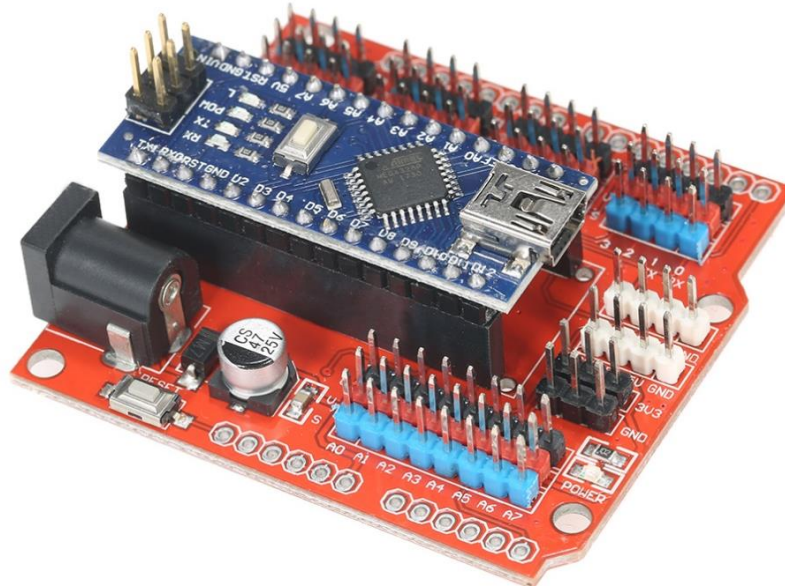


Рисунок 1.25 – Встановлення Arduino Nano на плату розширення [50]

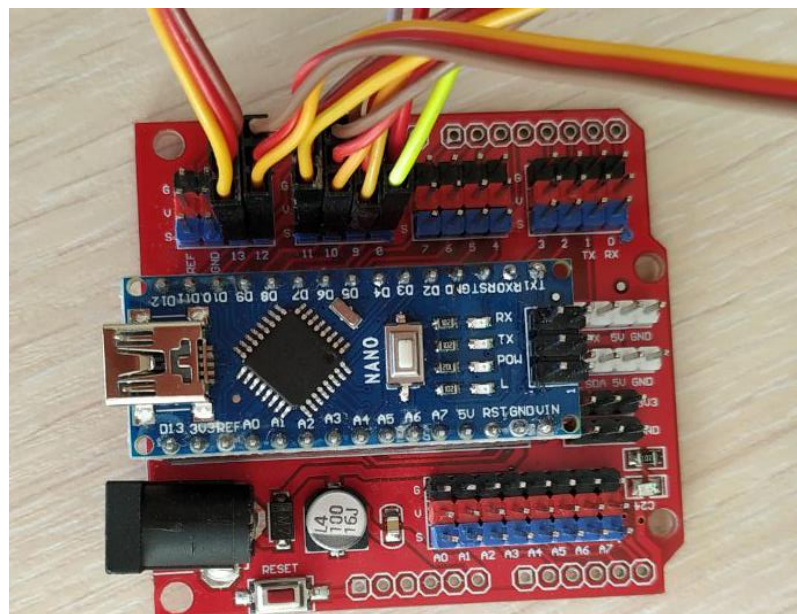


Рисунок 1.26 – Підключення серводвигунів до плати Arduino Nano Shield

Бібліотека Servo дозволяє здійснювати програмне управління серводвигунами. Синтаксис є наступним `# include "Servo.h"`. Для роботи з серводвигунами вказується змінна типу Servo. Управління здійснюється наступними функціями:

- `attach ()` – приєднує змінну до конкретного виводу. Синтаксис наступний: `servo.attach (pin)`, де `pin` – номер виводу, до якого приєднують серводвигун;
- `write ()` – повертає команду серводвигуну прийняти деяке значення параметра. Синтаксис наступний: `servo.write (angle)`, де `angle` – кут, на який повинен повернутися серводвигун.

Якщо необхідно зігнути одночасно тільки конкретний палець або декілька пальців, наприклад, мізинець та безіменний, то потрібно внести зміни в `void loop ()`, тобто вказати значення відповідних виводів в цій функції.

Таким чином, розроблене раніше в середовищі Arduino IDE програмне забезпечення прошивається в мікроконтролер (МК) та реалізує послідовне згинання/розгинання п'яти пальців, а для інших варіантів рухів та захоплень необхідно вносити зміни до коду програми, що є неприйнятним для звичайного користувача.

1.5 Висновки за розділом 1

В цьому розділі було розглянуто принципи розробки та функціонування, а також особливості управління існуючими протезами верхніх кінцівок людини. Було показано доцільність подальшого розвитку методів управління роботизованими протезами, які є більш дешевою альтернативою біонічним.

Проаналізувавши особливості існуючого на кафедрі програмних засобів РП, було вирішено удосконалити програмну частину та створити графічний інтерфейс для ефективної взаємодії користувача з РП.

Метою роботи є розробка програмного забезпечення для підвищення ефективності управління роботизованим протезом верхньої кінцівки людини.

Завдання роботи:

- виконати аналіз вимог до розроблюваного ПЗ для управління РП;
- дослідити апаратні інтерфейси для передачі даних;
- обрати програмне середовище розробки;
- розробити алгоритм управління РП;
- виконати програмну реалізацію;
- розробити програмну документацію.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз вимог

Розроблюване ПЗ призначене для управління РП шляхом взаємодії користувача з визначеними зонами на графічному інтерфейсі для забезпечення базових рухів та захоплень.

Можна виділити дві основні групи користувачів розроблюваного ПЗ:

- користувач, що планує використовувати РП в процесі навчання з метою ознайомлення з принципами управління РП та виконання певних рухів та захоплень;
- програміст, який займається розробками в галузі протезування та може використовувати даний РП в якості прототипу для досліджень алгоритмів управління.

На основі аналізу виявлених груп користувачів можна сформулювати наступні вимоги до розроблюваного ПЗ.

Функціональні вимоги:

- взаємодія з платою Arduino через апаратний інтерфейс передачі даних;
- формування керуючих впливів на частини РП з метою отримання базових рухів або захоплень;
- розпізнавання та виконання команд користувача;
- можливість вказати точку на певній ділянці долоні для точкового управління.

Нефункціональні вимоги:

- ПЗ має працювати на різних платах Arduino;
- графічний інтерфейс має бути наочним і зрозумілим;
- певні точки взаємодії мають підсвічуватися для інтерпретації отриманої команди;
- мають надаватися зрозумілі для користувача назви для ділянок для покращення результатів очікуваних подій.

Процес взаємодії користувача з розробленим ПЗ представлений у вигляді діаграми прецедентів, що наведена на рисунку 2.1.

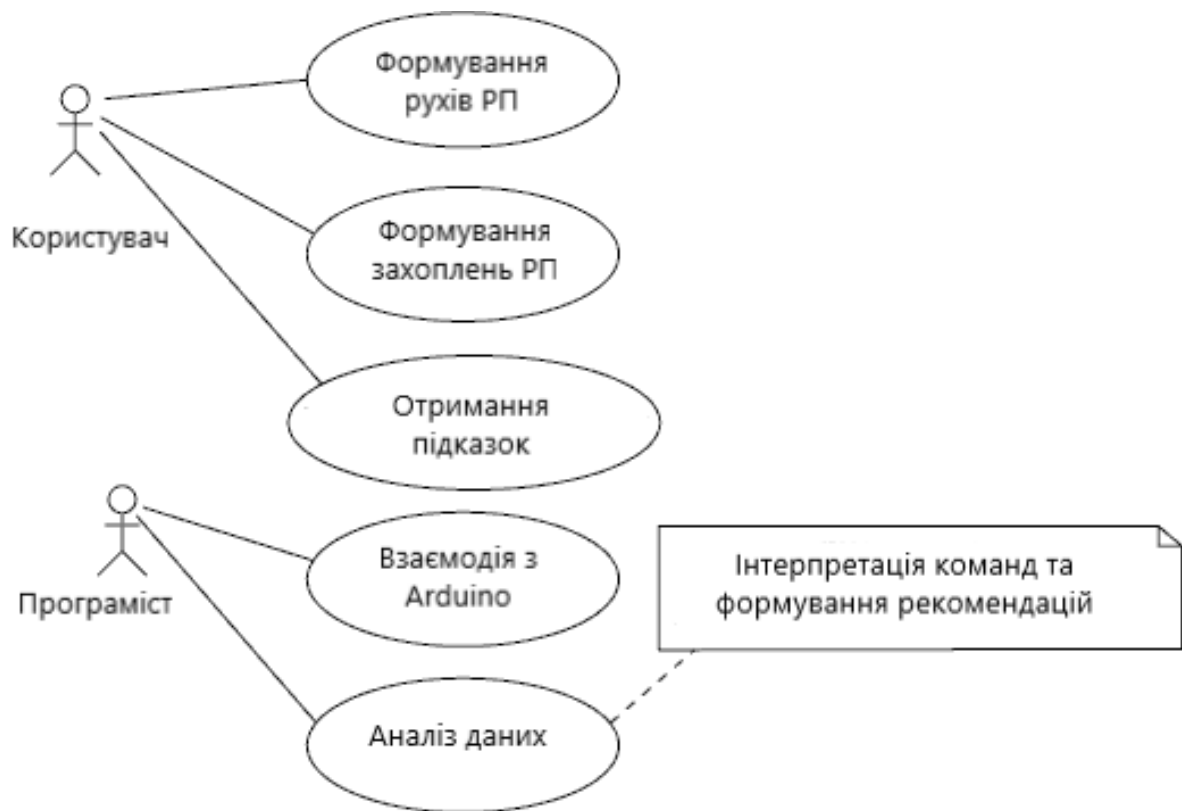


Рисунок 2.1 – Діаграма прецедентів

2.2 Огляд та вибір апаратного інтерфейсу передачі даних

Даними з МК можуть обмінюватися датчики, виконавчі пристрої, індикатори, дисплеї та інші пристрої. Причому зв'язок може бути як двосторонній так і односторонній, коли, наприклад, датчик температури передає інформацію на МК і нічого не чекає від нього у відповідь [52].

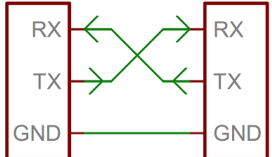
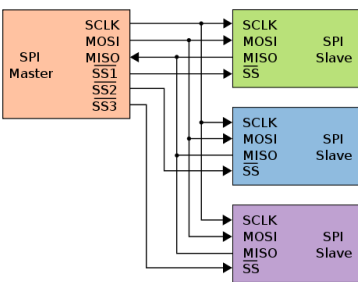
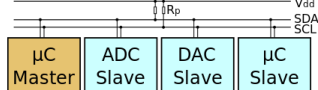
В апаратному вигляді у всіх плат Arduino присутні три інтерфейси: UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface) та I²C (Inter-Integrated Circuits), додатково деякі мають вбудований адаптер CAN-шини. У кожного з перерахованих інтерфейсів є свої переваги та недоліки, а також свої "клієнти", тобто пристрої, які використовують тільки той чи інший інтерфейс для зв'язку із зовнішнім світом [52].

Всі плати Arduino (крім Pro та Pro Mini) мають «на борту» USB-роз'єм і спеціальну мікросхему, яка перетворює інтерфейс UART на USB [53].

Середовище розробки Arduino IDE вже має у своєму складі монітор послідовного порту, який дозволяє переглядати дані, що проходять через порт, а також надіслати свої дані. Таким чином, завжди можливо передати дані з пристрою прямо до комп'ютера та вивести на екран [53]. Монітор порту - не єдиний засіб для роботи з ним, в мережі можна знайти безліч програм-моніторів, які дозволяють не тільки приймати дані, а й будувати графіки на їх основі, записувати дані в таблицю та багато іншого. Це значно спрощує налагодження програми [53].

Порівняльну характеристику трьох інтерфейсів передачі даних наведено в таблиці 2.1 [54].

Таблиця 2.1 – Порівняльна характеристика інтерфейсів передачі даних [54]

Характеристика	UART	SPI	I ² C
1	2	3	4
Схема			
Позначення контактів	TxD: Передача даних. RxD: Отримання даних	SCLK: послідовний порт MOSI: Master Out Slave In. MISO: Master In Slave Out .	SDA: Дані SCL: Послідовний порт
Швидкість передачі даних	Максимальна швидкість зв'язку може становити від 230 Кб/с до 460 КБ/с.	Зазвичай від 10 Мб/с до 20 Мб/с.	До 3,4 Мб/с, деякі варіанти можуть досягати 1 Мб/с

Продовження таблиці 2.1

1	2	3	4
Відстань	Менше 15 метрів щита. Так само, як і вище,	Призначений для комунікацій всередині пристрою	Призначений для комунікацій всередині пристрою
Тип зв'язку	Асинхронний	Синхронний	Синхронний
Апаратна складність	Невисока	Середня	Достатня
Кількість Майстрів	немає	Один	Кілька
Годинник	Кожен пристрій використовує свій внутрішній годинник.	Синхронний сигнал між ведучим і підпорядкованим.	Загальний тактовий сигнал між кількома ведучими та підпорядкованими.
Протокол	8-розрядний протокол із стартовим і стоповим бітом.	Кожна компанія використовує свій власний протокол.	Використовує початковий і стоповий біт, біт підтвердження (ACK) кожні 8 біт.
Програмна адресація	Зв'язок між двома пристроями, це не обов'язково.	SS (SS1, SS2...) для використовується для вибору пристрою, чим більше пристроїв, тим більше виходів SS нам потрібно.	Всі головні можуть спілкуватися з усіма підлеглими, ми можемо поставити 27 ведених і ввести адресу в протокол I ² C.
Переваги	Дуже простий, дозволяє швидко підключити два пристрої, зазвичай використовується, наприклад, з RS232 або RS485.	Простий; має повнодуплексний зв'язок; використовує push-pull і високу швидкість передачі даних; вимагає менше енергії, ніж I ² C	Більше одного Майстра; для зв'язку знадобиться всього два кабелі; простіше, ніж SPI, оскільки вимагає менше рядків; концепція відкритого колектора; має контроль потоку.

Продовження таблиці 2.1

1	2	3	4
Недоліки	Обмежується зв'язком між двома пристроями; швидкість має бути узгоджена на початку з'єднання, якщо вона не однакова, це дасть помилкові показання.	Чим більше підпорядкованих пристроїв, тим більше потрібно зв'язків. Щоб додати пристрій, потрібне додаткове підключення. Немає контролю потоку.	Складність схеми збільшується при збільшенні кількості ведучих та підлеглих. I2C є напівдуплексним. ПЗ може перевантажувати процесор.

I²C - послідовна шина даних для зв'язку інтегральних схем, розроблена як проста шина внутрішнього зв'язку для створення керованої електроніки. I²C використовує дві двонаправлені лінії, підтягнуті до напруги живлення і керовані через відкритий колектор або відкритий стік. Це так звані послідовна лінія даних та послідовна лінія тактування. Стандартні напруги +5 В або +3,3 В, проте допускаються й інші. Переваги шини I²C – вона залишається двопровідною, незалежно від кількості підключених до неї мікросхем [55].

Переваги шини SPI - гранична простота протоколу передачі фізично обумовлює високу надійність і швидкодію передачі. Гранична швидкодія шини SPI вимірюється десятками мегагерц і тому вона ідеальна для потокової передачі великих обсягів даних і широко використовується в високошвидкісних ЦАП/АЦП, драйверах світлодіодних дисплеїв і мікросхемах пам'яті. З іншого боку, інтерфейс SPI є основою побудови низки спеціалізованих інтерфейсів [55].

Протокол USART/UART забезпечує як синхронний, так і асинхронний послідовний зв'язок. При цьому забезпечується наявність різних швидкостей передачі, що робить його придатним для широкого застосування. Це одна з найпростіших форм послідовного зв'язку. Недоліком тут є те, що одночасно можуть бути підключені лише два пристрої.

Таким чином, рекомендується використовувати SPI, якщо є лише один провідний та кілька керованих пристроїв, оскільки SPI для цього є найшвидшим протоколом. У випадку, коли є кілька провідних пристроїв та кілька керованих пристроїв, слід віддати перевагу використанню I²C замість SPI, що дозволить зменшити кількість використовуваних дротів. Якщо потрібен протокол для послідовного зв'язку лише двох пристроїв між собою, то USART/UART виявляється найкращим рішенням, оскільки його легко застосовувати і з ним просто працювати у багатьох периферійних пристроях [56]. Тому, саме USART/UART буде обрано для використання в даній розробці.

2.3 Вибір середовища програмної розробки

Зазвичай розробники організовують "спілкування" з платою Arduino через монітор послідовного порту Arduino IDE. Але, набагато зручніше працювати з програмою, написаною будь-якою мовою програмування, що має бібліотеки для обміну по послідовному порту [57].

Мова програмування Processing дуже схожа на мову Arduino, тим не менш, мова Arduino (на основі Wiring) реалізована на C/C++, певним чином відрізняється від мови Processing, яка заснована на Java (див. табл. 2.2) [58].

Інші популярні мови програмування, для яких існують великі бібліотеки для роботи з послідовним портом – це Python, PHP, Visual Basic, C і т. п. [57]. Наприклад, бібліотека pySerial забезпечує зв'язок з Arduino шляхом інкапсуляції для послідовного порту. Вона надає доступ до параметрів послідовного порту через властивості Python та можливість його налаштування.

Processing – це мова програмування з відкритим вихідним кодом та безкоштовне для завантаження середовище доступне для GNU/Linux, Mac OS X та Windows. Processing дозволяє читати передані послідовні дані, а

також створити простий графічний інтерфейс користувача (Graphical User Interface, GUI) для відправки команд на плату Arduino.

Таблиця 2.2 – Порівняльна характеристика мовних конструкцій Processing та Arduino [58]

Мовна конструкція	Arduino	Processing
Масиви	int bar[8]; bar[0] = 1;	int[] bar = new int[8]; bar[0] = 1;
	int foo[] = { 0, 1, 2 };	int foo[] = { 0, 1, 2 }; <i>or</i> int[] foo = { 0, 1, 2 };
Цикли	int i; for (i = 0; i < 5; i++) { ... }	for (int i = 0; i < 5; i++) { ... }
Виведення	Serial.println("hello world");	println("hello world");
	int i = 5; Serial.println(i);	int i = 5; println(i);
	int i = 5; Serial.print("i = "); Serial.print(i); Serial.println();	int i = 5; println("i = " + i);

Переваги Processing:

- проста мова програмування для створення інтерактивних візуальних ефектів;
- базується на графічних можливостях мови Java та використовує її спрощений синтаксис;
- легкий та швидкий інструментарій для людей, які бажають програмувати зображення, анімацію та інтерфейси;

- створена для вивчення основ комп'ютерного програмування у візуальному контексті;
- це відкритий, абсолютно безкоштовний проєкт, а вихідний код є доступним для всіх бажаючих [59].

Середовище програмування Processing схоже із середовищем програмування Arduino IDE (див. рис. 2.2.), що надає додаткові переваги при вивченні та використанні.

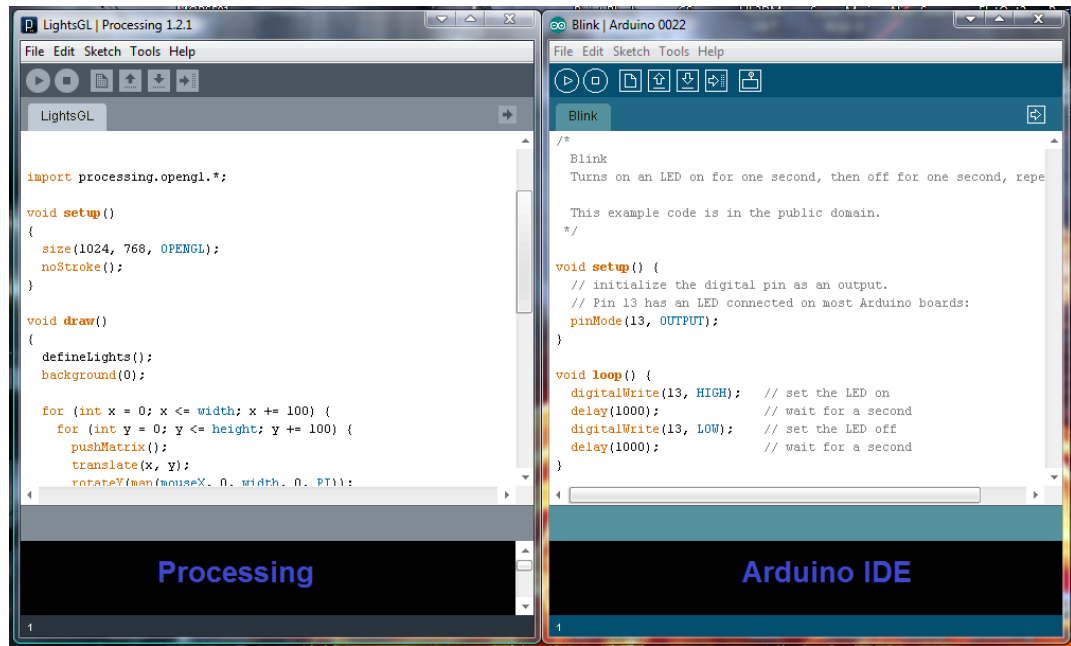


Рисунок 2.2 – Середовища розробки Processing та Arduino IDE

Порівняльну характеристику двох середовищ наведено в таблиці 2.3.

Таблиця 2.3 – Порівняльна характеристика популярних середовищ розробки для платформи Arduino

Характеристики	Середовище розробки	
	Arduino IDE	Processing
Простота	+	+
Підтримка	+	+
Мова програмування	C / Wiring	Processing, /Java
Доступність	Open-source	Open-source

2.4 Висновки за розділом 2

В цьому розділі було проведено аналіз вимог до розроблюваного ПЗ та обґрунтування вибору апаратних та програмних засобів розробки.

Розроблено діаграму прецедентів, що демонструє варіанти взаємодії користувача та програміста з розроблюваним ПЗ.

На основі порівняльного аналізу трьох інтерфейсів було вирішено, що для даної розробки буде використано апаратний інтерфейс USART/UART, оскільки його легко застосовувати і з ним просто працювати у багатьох периферійних пристроях.

Графічний інтерфейс користувача для відправки команд управління РП на плату Arduino буде створено в середовищі програмування Processing, яке надає для цього легкий та швидкий інструментарій.

3 КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Узагальнений алгоритм управління РП та структура програми

В загальному вигляді алгоритм управління РП зображено на рисунку 3.1.



Рисунок 3.1 – Схема управління РП

Управління РП має здійснюватися за допомогою графічного інтерфейсу, розробленого на основі Processing, що надасть можливість обирати дії з пальцями РП. Відповідна команда буде передаватися через Serial порт на плату Arduino, де в свою чергу формуватиметься керуючий вплив для серво та відповідне формування рухів та захоплень.

Розроблюване ПЗ складається з двох основних модулів – модуль для плати Arduino та графічний інтерфейс користувача (рис. 3.2).

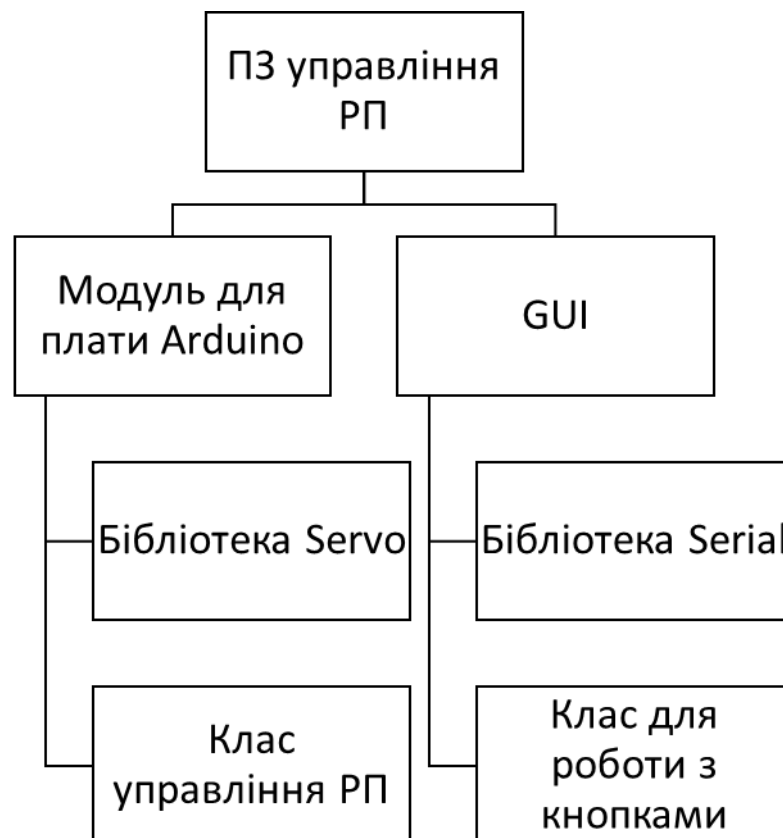


Рисунок 3.2 – Структура програми

3.2 Опис реалізації програми для плати Arduino

Для реалізації управління РП необхідне програмне забезпечення, яке «прошивається» в плату Arduino Nano [60] - [62]. Зокрема необхідно навчити плату розпізнавати команди та, в залежності від введеної інформації, виконувати рухи РП.

Для спілкування між платою Arduino і комп'ютером або іншим пристроєм в МК використовується інтерфейс USART/UART, який в поєднанні з вбудованим в UNO USB-to-UART конвертером, дозволяє встановити двоспрямований зв'язок з комп'ютером через віртуальний послідовний порт. Порт з'єднується через цифровий пін 0 (RX) і 1 (TX) при підключенні до комп'ютера через USB, тому не рекомендується використовувати піни 0 і 1 для введення / виведення.

Раніше на старих комп'ютерах були COM-порти, зараз вони створюються віртуально за допомогою мікросхеми FTDI, коли підключаємо плату до комп'ютера через USB.

Можна як посилати сигнал з комп'ютера, наприклад, з клавіатури, так і приймати сигнали з плати.

Для коректної роботи з портом слід дотримуватися двох умов: вибрати правильний COM-порт, вибрати швидкість роботи в скетчі, яка повинна збігатися зі швидкістю, обраною в моніторі порту.

Для роботи з серводвигунами, що виконують керування механічною частиною, використовується бібліотека Servo. Синтаксис є наступним `#include "Servo.h"`. Використовується 5 змінних типу Servo. Для спілкування використовується клас Serial.

Управління здійснюється наступними функціями, що представлені в таблиці 3.1.

В програмі є два основних цикла – `setup` та `loop`. Обидва цикла викликаються вбудованою функцією `main ()` з файлу `main.cpp`. При цьому функція `void setup ()` викликається один раз, а й `void loop ()` викликається в циклі `for` нескінченну кількість разів.

В циклі `setup` встановлюємо послідовне з'єднання. В циклі `loop` програма зчитує символ з буферу. За допомогою конструкції `if-else` програма перевіряє символ на відповідність та в разі співпадіння виконує блок команд, що відповідає за певний рух.

Таблиця 3.1 – Функції програми

Функція	Опис
attach(pin)	підключає змінну Servo до виходу конкретного піну, де pin - номер піна, до якого приєднують серводвигун
detach()	від'єднує змінну Servo від зазначеного виходу
write()	повертає команду серводвигуну прийняти деяке значення параметра. Синтаксис наступний: servo.write (angle), де angle - кут, на який повинен повернутися серводвигун
delay(ms)	зупиняє виконання програми на задану в параметрі кількість мілісекунд (1000 мілісекунд в 1 секундi)
Serial.begin(speed)	ініціює послідовне з'єднання і задає швидкість передачі даних в біт / с (бод)
Serial.available()	отримує кількість байт (символів) доступних для читання з послідовного інтерфейсу зв'язку. Це ті байти які вже надійшли і записані в буфер послідовного порту. Буфер може зберігати до 64 байт
Serial.read()	зчитує черговий доступний байт з буфера послідовного з'єднання
Serial.flush()	очищає буфер послідовного з'єднання

Алгоритм управління РП (рис. 3.3) складається з наступних етапів.

Блок 1. Початок.

Блок 2. Ініціалізація сервоприводів.

Блок 3. Ввімкнення сервоприводів.

Блок 4. Перевірка поточної позиції сервопривода.

Блок 5. Повернення на початкову позицію.

Блок 6. Умова: чи слід продовжити роботу?

Блок 7. Приведення сервоприводів в дію, для формування руху або захоплення.

Блок 8. Завершення роботи.

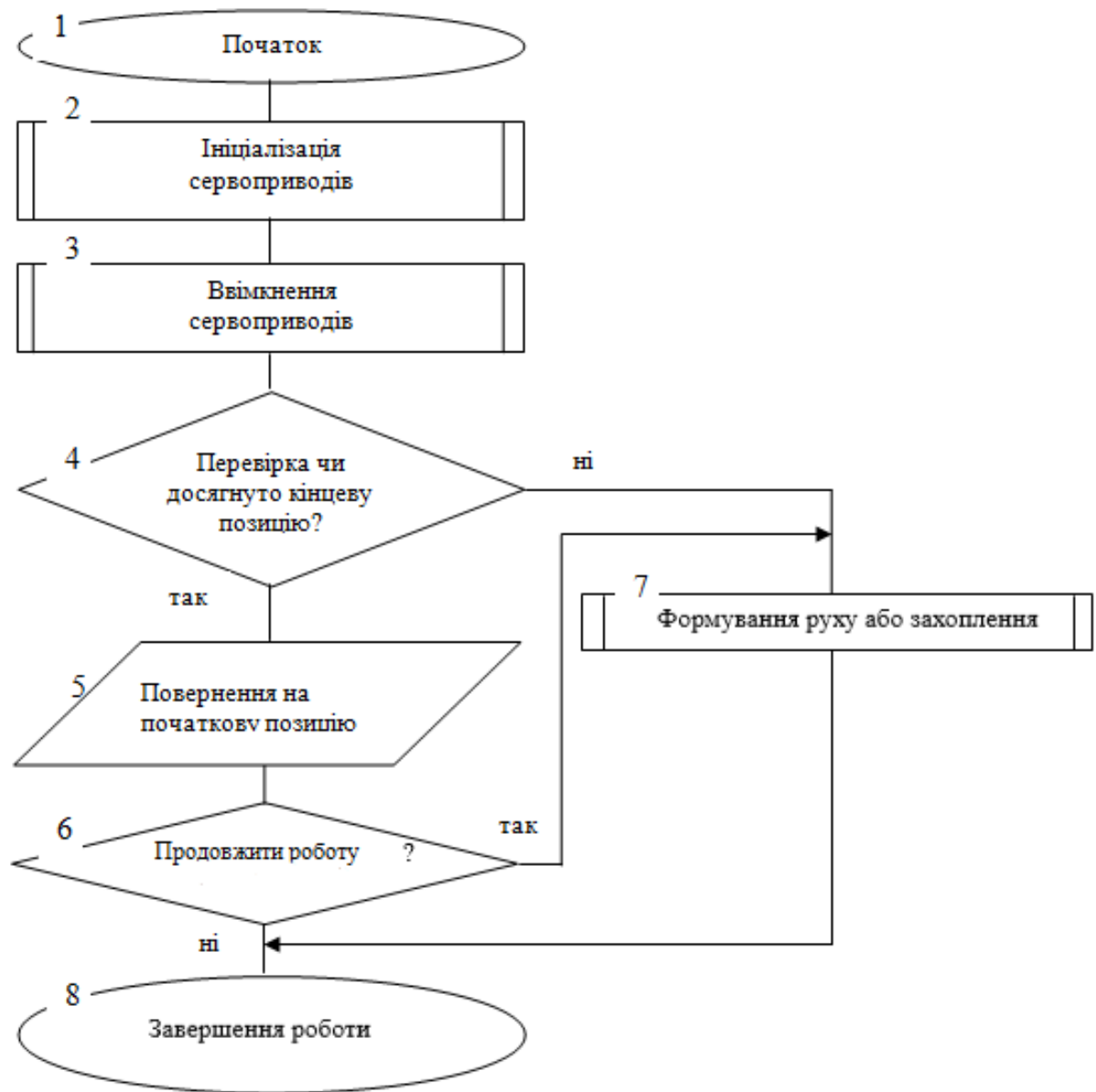


Рисунок 3.3 – Схема алгоритму програми управління РП

Таким чином, модуль для плати Arduino складається з таких складових:

- бібліотека для роботи з сервоприводами (`#include "Servo.h"`);
- клас управління РП (`class Finger`);
- змінні (`Servo servo, byte pos`);
- функції (`init(int16_t interval, byte pin); void update(); void setup(); void loop();`).

Текст програми для плати Arduino наведено в додатку А.1.

3.3 Опис програмної реалізації графічного інтерфейсу

У загальному випадку, всі елементи графічного інтерфейсу можна умовно поділити на чотири групи, що поєднують елементи за деякими загальними властивостями [63]:

- елементи управління, за допомогою яких формуються керуючі впливи, а інформація про стан елемента надсилається з керуючого пристрою до МК;
- сенсорні елементи, за допомогою яких можна передати в МК дані сенсорів керуючого пристрою;
- елементи індикації, за допомогою яких може відображатися будь-яка інформація та дані, спрямовані з МК в керуючий пристрій;
- декоративні елементи не передають будь-які дані, вони необхідні щоб зробити інтерфейс більш красивим і читаним.

Для підвищення ефективності управління РП було вирішено створити графічний інтерфейс мовою Processing, оскільки достатньо просто можливо реалізувати зв'язок програми на Processing і МК плати Arduino. Для цієї мови програмування існує бібліотека Serial, яка дозволяє приймати повідомлення, відправлені як `Serial.write ()`, а так само дозволяє відправляти повідомлення як `Serial.print ()`.

Важливо відзначити, що при подібній відправці повідомлення буде записано в буфер порту, а значить буде прочитано МК. Так що потрібно лише підключитися до потрібного Serial порту і приймати / відправляти на нього повідомлення.

Щоб забезпечити отримання даних від послідовного порту, потрібен об'єкт класу Serial. Оскільки з Arduino відправляються дані типу String, нам треба отримати рядок і в Processing.

Для реалізації графічного інтерфейсу використовується фонове зображення руки. Змінна типу `PImage` зберігає зображення всередині програми.

Використовується бібліотека `serial`. Синтаксис є наступним: `import processing.serial.*`. Типи змінних, що використовуються в програмі представлено в таблиці 3.2.

Таблиця 3.2 – Типи змінних програми

Тип змінної	Опис
<code>int</code>	для задання позицій кнопок, задання її розмірів
<code>color</code>	для задання кольорів кнопок при наведенні, натисненні, та в неактивному положенні
<code>boolean</code>	відображають чи знаходиться курсор в межах кнопки

Для кнопок створений власний клас `MyButton`. В якості вхідних параметрів виступають змінні типу `float` що визначають позицію кнопки та змінна типу `String`, що зберігає інформацію про кнопку, яка відображається при наведенні. Всередині класу описана функція `ButtonDraw`, що відповідає за відображення кнопки. В програмі створено 12 екземплярів класу `MyButton`.

Функції `setup` та `draw` в `processing` є аналогами `setup` та `loop` в `Arduino`.

В функції `setup` встановлюємо послідовне з'єднання, встановлюємо розміри програми, розмір тексту та завантажуюмо фонове зображення отримуємо доступний послідовний порт (як правило, це перший доступний порт зі списку), налаштовуємо об'єкт `Serial`, вказавши порт і швидкість передачі даних.

Функція `draw` безпервно виконує рядки коду, що містяться всередині свого блоку, до тих пір, поки програма не буде зупинена. `Draw` викликається автоматично і не повинна викликатися явно. Всі програми обробки оновлюють екран в кінці.

За допомогою конструкції `if-else` програма перевіряє активність функції `ButtonDraw` для кожної кнопки.

Текст програми модуля графічного інтерфейсу керування наведено в Додатку А.2. Зовнішній вигляд інтерфейсу керування наведено на рисунку 3.4.



Рисунок 3.4 – Інтерфейс керування протезом

Верхній ряд кнопок відповідає за згинання відповідного пальця, нижній ряд за розгинання пальців, дві нижні кнопки відповідають за

стиснення та розтиснення кулаку. У разі натиснення на кнопку, вона відправляє команду за допомогою функції `serial.wtite()`.

При наведенні курсора на кнопку вона змінює свій колір та відображається інформація про її функціональне призначення, тобто підказка для користувача, як зображено на рисунку 3.5.

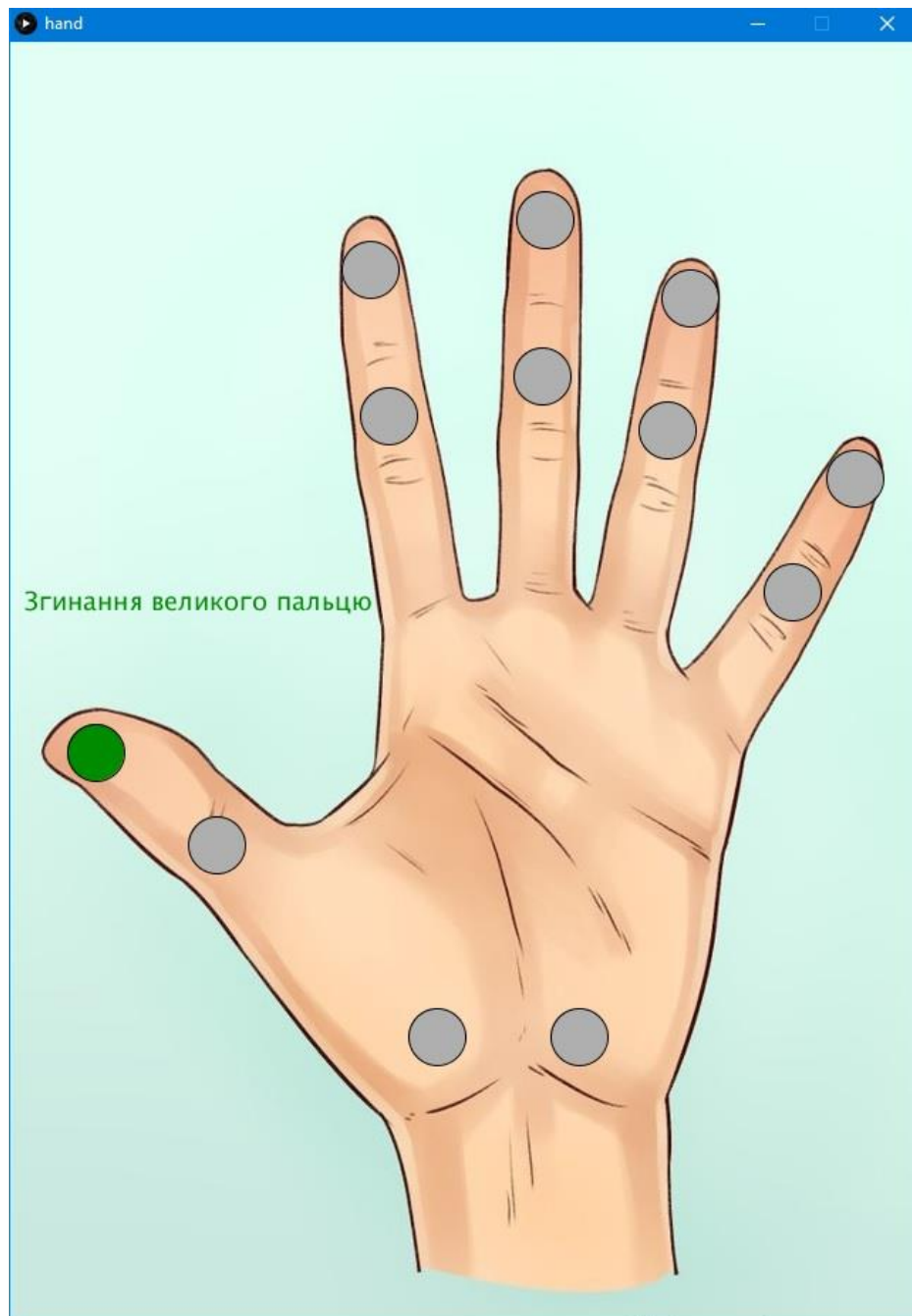


Рисунок 3.5 – Наведення на кнопку

При натисненні на кнопку через Serial порт відправляється відповідна команда, кнопка змінює свій колір та відображається інформація про команду, що виконується, як зображено на рисунку 3.6.

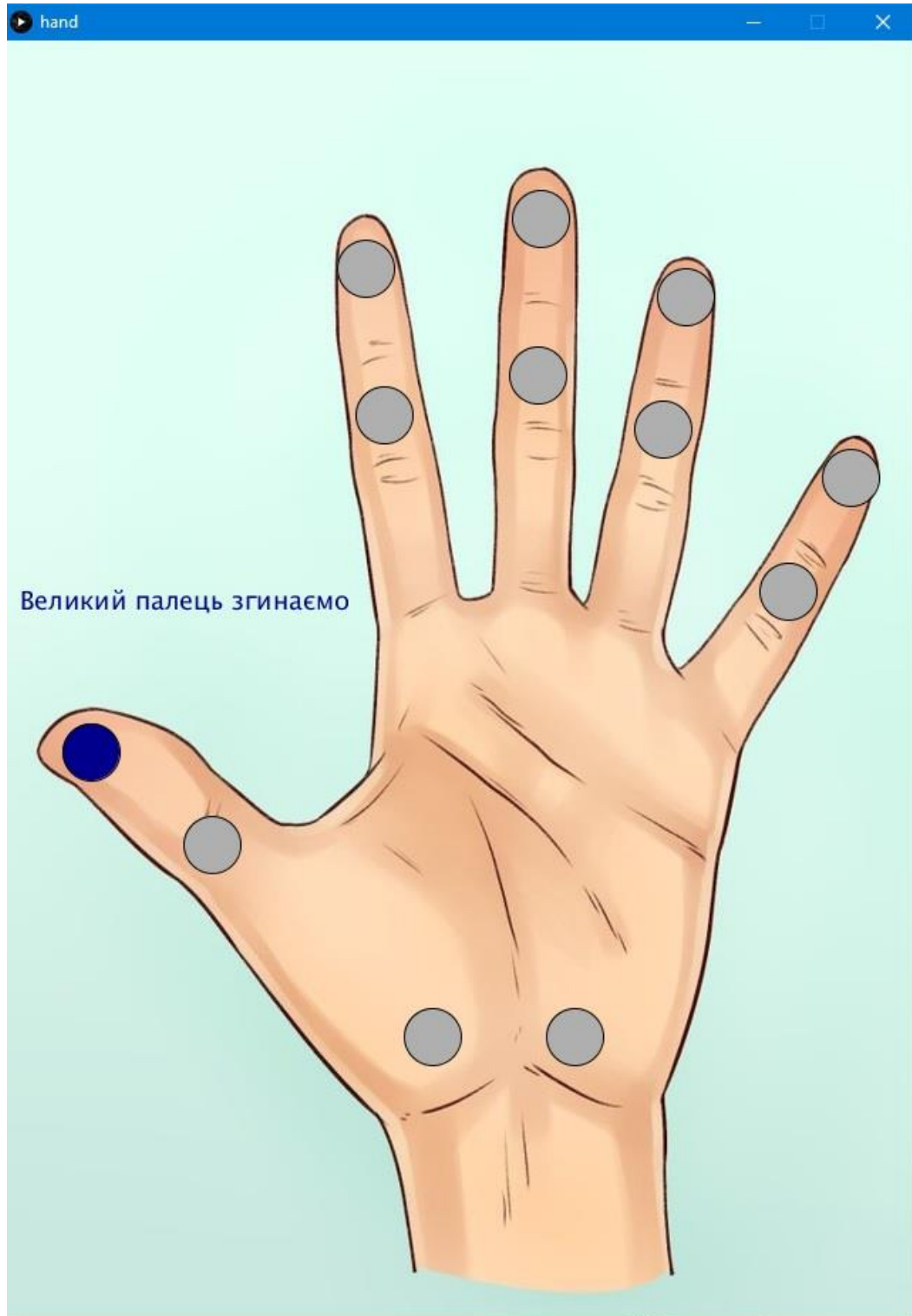


Рисунок 3.6 – Натиснення на кнопку

3.4 Висновки за розділом 3

В третьому розділі були розглянуті основні рішення щодо реалізації компонентів системи, описано загальний алгоритм роботи програми. Також було описано бібліотеки та функції, що використовувались для створення програмного забезпечення.

Інтерфейс програмного забезпечення створено засобами мови програмування Processing.

Розроблений застосунок дозволяти керувати РП за допомогою графічного інтерфейсу та виконувати базові рухи та захоплення.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення й умови застосування програми

Програма призначена для управління РП шляхом взаємодії користувача з визначеними зонами на графічному інтерфейсі для забезпечення базових рухів та захоплень.

Програма функціонує на платі Arduino Nano з використанням серводвигунів Tower Pro 9g SG90 та плати розширення Arduino Nano Shield V3.0.

Для функціонування графічного інтерфейсу керування, а також в разі його доопрацювання, необхідно мати таку мінімальну апаратну платформу: персональний комп'ютері з тактовою частотою мікропроцесора не менше 1 ГГц, з оперативною пам'яттю з мінімальним об'ємом 1 Гб. На персональному комп'ютері повинна бути встановлена операційна система Windows XP/7/10.

4.2 Характеристики програми

Програмний продукт надає наступні можливості:

- підключення до роботизованого протезу через інтерфейс USART/UART;
- розпізнавання та виконання команд управління;
- формування захоплення шляхом стискання та розтискання руки в кулак;
- формування базових рухів шляхом стискання та розтискання кожного окремого пальцю;
- індикація виконання команд управління та надання підказок користувачу.

4.3 Звернення до програми

Для роботи програми необхідно завантажити програмний код до плати розширення Arduino Nano (в разі першого запуску), підключити через USART/UART інтерфейс РП до комп'ютера та запустити графічний інтерфейс керування.

Для роботи повинні бути у наявності монітор, клавіатура, та маніпулятор «миша». Для стабільності роботи рекомендується підключити живлення до плати Arduino Nano.

4.4 Вхідні та вихідні дані

Вхідними даними до програми є сигнали та команди управління. Вихідними даними є повідомлення про виконану дію та статус РП.

4.5 Повідомлення програмісту

Для полегшення розуміння функцій програми в коді залишені коментарі. Програма може бути дороблена шляхом перекомпіляції на інші платформи, доробки дизайну або шляхом додавання нових функцій.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення програми

Програма призначена для управління РП шляхом взаємодії користувача з визначеними зонами на графічному інтерфейсі для забезпечення базових рухів та захоплень

5.2 Умови виконання програми

Програма для фізичного прототипу працює на платформі Arduino Nano з використанням серводвигунів Tower Pro 9g SG90 та плати розширення Arduino Nano Shield V3.0

Для функціонування розробленого програмного продукту, необхідно підключити РП до комп'ютера та запустити графічний інтерфейс керування.

Для роботи повинні бути у наявності монітор, клавіатура, та маніпулятор «миша». Для стабільності роботи рекомендується підключити живлення до плати Arduino Nano.

5.3 Робота з програмою

Для запуску програми необхідно підключити прототип через USART інтерфейс протез до комп'ютера та запустити на виконання файл `hand.pde`. (рис. 5.1)

Для формування керуючого впливу з метою формування руху або захоплення можливо обрати відповідні контрольні точки на графічному зображенні долоні:

- для згинання конкретного пальця – верхня кнопка на зображенні пальця;
- для розгинання конкретного пальця – нижня кнопка на зображенні пальця;

- для стискання всіх пальців – ліва кнопка на зображенні долоні;
- для стискання всіх пальців – ліва кнопка на зображенні долоні;
- для розтискання всіх пальців – права кнопка на зображенні долоні.

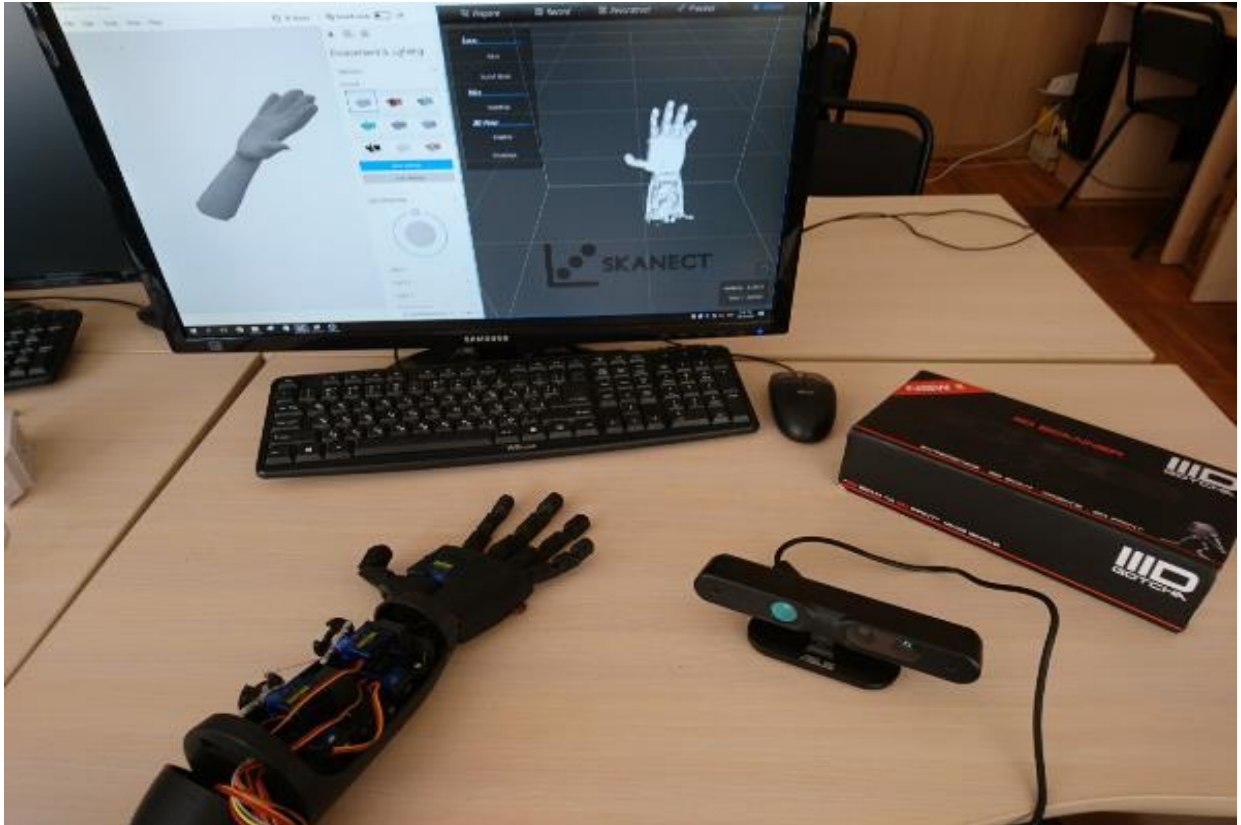


Рисунок 5.1 – Процес керування РП

5.4 Повідомлення оператору

Повідомлення про неможливість підключитися до COM-порту може виникати при неправильному підключенні або при одночасному підключенні до РП графічного інтерфейсу та інших сторонніх програм.

ВИСНОВКИ

В дипломній роботі проаналізовано принципи створення та функціонування, конструктивні й функціональні особливості сучасних протезів верхніх кінцівок людини.

Як показали проведені дослідження, на сьогоднішній день існує багато варіантів біонічних протезів, які використовуються як функціональні замітники втрачених верхніх кінцівок пацієнтів. Ці протези дозволяють не лише відновлювати природний зовнішній вигляд людини, але й відновлювати найважливіші втрачені функції. Тим не менш, залишається багато відкритих питань щодо вартості таких протезів, розробки та удосконалення алгоритмів управління.

Тому, в роботі показано, що актуальною задачею є не лише розвиток біонічних протезів, але й роботизованих протезів, які є більш дешевими аналогами, що надають пацієнтам базовий набір рухів.

Для досягнення мети роботи було удосконалено ПЗ для РП, розробленого на кафедрі програмних засобів. Створено графічний інтерфейс з використанням мови програмування Processing у вигляді Windows-застосунку.

Тестування розробленого ПЗ підтвердило його працездатність. Розроблений графічний інтерфейс дозволяє більш наочно та ефективно здійснювати управління РП та виконувати базові рухи та захоплення.

В подальшому можливо додати функціонал та удосконалити графічний інтерфейс, наприклад, шляхом реалізації тривимірної візуалізації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Недорез, Г.О. Система керування протезом верхньої кінцівки [Електрон. ресурс] / Г. О. Недорез. – Режим доступу: https://openarchive.nure.ua/bitstream/document/11575/1/2019_M_BMI_Nedorez_GO-1-51.pdf.
2. Протезы рук: виды и перспективы развития [Электрон. ресурс]. – Режим доступа: <https://vils.ru/articles/protezy-ruk-vidy-i-perspektivy-razvitiya/>.
3. Ильин, Ю. Бионические протезы: какие органы сегодня можно подменить электроникой [Электрон. ресурс] / Ю. Ильин. – Режим доступа: <https://www.computerra.ru/184048/bionic/>.
4. Попадюха, Ю. Особливості біонічних протезів верхніх кінцівок [Електрон. ресурс] / Ю. Попадюха. – Режим доступу: <https://evnuir.vnu.edu.ua/bitstream/123456789/13204/1/Yuriy%20Popadiukha.pdf>.
5. Стефанов, Ю. Как работают бионические конечности? [Электрон. ресурс] / Ю. Стефанов. – Режим доступа: <https://theoryandpractice.ru/posts/2353-kak-rabotayut-bionicheskie-konechnosti>.
6. Бионическая рука i-LIMB Pulse [Электрон. ресурс]. – Режим доступа: <http://roboting.ru/1170-bionicheskaya-ruka-i-limb-pulse.html>.
7. Бионический протез руки I-LIMB ULTRA REVOLUTION [Электрон. ресурс]. – Режим доступа: <http://www.tehnari.ru/f157/t98034/>.
8. Современные технологии протезирования: как ИТ помогает людям жить [Электрон. ресурс]. – Режим доступа: https://itc.ua/articles/covremennye_tehnologii_protezirovaniya_kak_it_pomogaet_lyudyam_zhit_48171/.
9. Британцы сделали бионический протез руки [Электрон. ресурс]. – Режим доступа: <http://ua-today.com/news-from-ukraine/60613/>.
10. Обзор рынка бионических рук на 2016 год: что можно купить в России [Электрон. ресурс]. – Режим доступа: <https://habr.com/ru/post/395115/>.
11. В Перми презентован уникальный бионический протез кисти [Электрон. ресурс]. – Режим доступа: <http://neinvalid.ru/v-permi-prezentovan->

unikalnyj-bionicheskij-protez-kisti/.

12. Бионические протезы [Электрон. ресурс]. – Режим доступа: https://www.livemd.ru/tags/bionicheskie_protezy/.

13. Как родная. Бионический протез руки с обратной связью [Электрон. ресурс]. – Режим доступа: <http://argumentua.com/stati/kak-rodnaya-bionicheskii-protez-ruki-s-obratnoi-svyazyu>.

14. Pantelopoulos, A. A survey on wearable sensor-based systems for health monitoring and prognosis / A. Pantelopoulos, N. G. Bourbakis // N.G. IEEE Trans. Syst. Man, Cybern. Part C. - 2009. –Vol. 40. – P.1–12.

15. Wearable biosensors for healthcare monitoring / [J. Kim, A. S. Campbell, B.E.F.de Ávila et.al] // J.Nat. Biotechnol. – 2019. – Vol. 37. – P. 389–406.

16. Arefin, M. Wireless biosensors for POC medical applications / M. Arefin, J. M. Redouté, M. Yuce / Medical Biosensors for Point of Care (POC) Applications; Duxford, United Kingdom: Woodhead Publishing Limited. – 2017. – P. 151–180.

17. Dresch-Langley, B. Wearable sensors for individual grip force profiling [Electronic source] / B. Dresch-Langley. – Access mode: <https://arxiv.org/abs/2011.05863>.

18. Distribution of grip force in three different functional prehension patterns / [C. Pylatiuk, A. Kargov, S. Schulz et.al] // J. Med Eng. Technol. – 2006. – Vol. 30. – P. 176–182.

19. Hand-grip strength of young men, women and highly trained female athletes / [D. Leyk, W. Gorges, D. Ridder et.al] // Eur. J. Appl. Physiol. – 2007. - Vol. 99. – P. 415–421.

20. Age-related differences in finger force control are characterized by reduced force production / [S. Vieluf, B. Godde, E. M. Reuter et.al] // Exp. Brain Res. – 2013. - Vol. 224. – P. 107–117.

21. Novice and expert haptic behaviours while using a robot controlled surgery system / [A. U. Batmaz, M. A. Falek, L. Zorn et.al] // Biomedical Engineering (BioMed): 13th IASTED International Conference, Innsbruck,

Austria, 20–21 February 2017: proceedings. – 2017. - P. 94–99.

22. Correlating grip force signals from multiple sensors highlights prehensile control strategies in a complex task-user system / [B. Dresch-Langley, F. Nageotte, P. Zannen et.al] // Bioengineering. – 2020. - Vol. 7. – P. 143.

23. Sensors for expert grip force profiling: Towards benchmarking manual control of a robotic device for surgical tool movements / [M. de Mathelin, F. Nageotte, P. Zanne et.al] // Sensors. – 2019. - Vol. 19. – P. 4575.

24. Wearable sensors for spatio-temporal grip force profiling [Electronic source] / [R. Liu, F. Nageotte, P. Zanne et. al]. – Access mode: <https://arxiv.org/abs/2101.06479>.

25. Wearable wireless biosensors for spatiotemporal grip force profiling in real time / [R. Liu, F. Nageotte, P. Zanne et. al] // Engineering Proceedings. Multidisciplinary Digital Publishing Institute: Basel, Switzerland. – 2020. – Vol. 2. – P. 45.

26. Introducing STRAS: A new flexible robotic system for minimally invasive surgery / [A. De Donno, L. Zorn, P.Zanne et. al] // Robotics and Automation: IEEE International Conference, Karlsruhe, Germany, 6–10 May 2013: proceedings. – Los Alamitos: IEEE, 2013. – P. 1213–1220.

27. Data mining techniques for wireless sensor networks: a survey / [A. Mahmood, K.Shi, S.Khatoun et. al] // Int. J. Distrib. Sens. Netw. - 2013. – Vol. 9. – P. 1-24.

28. A novel telemanipulated robotic assistant for surgical endoscopy: Preclinical application to ESD / [L. Zorn, F. Nageotte, P. Zanne et. al] // IEEE Trans. Biomed. Eng. – 2017. – Vol. 65. – P. 797–808.

29. Batmaz, A.U. Getting nowhere fast: trade-off between speed and precision in training to execute image-guided hand-tool movements / A. U. Batmaz, M.de Mathelin, B. Dresch-Langley//BMC Psychol.–2016 –Vol.4.– P.1–19.

30. Dresch, B. Local brightness mechanisms sketch out surfaces but do not fill them / B. Dresch // Psychophysical evidence in the Kanizsa square. Percept. Psychophys. – 1992. - Vol. 52. – P. 562–570.

31. Batmaz, A.U. Seeing virtual while acting real: Visual display and

strategy effects on the time and precision of eye-hand coordination / A. U. Batmaz, M. de Mathelin, B. Dresch-Langley // PLoS ONE. – 2017. – Vol. 12. – P. 1-18.

32. Batmaz, A.U. Effects of 2D and 3D image views on hand movement trajectories in the surgeon's peri-personal space in a computer controlled simulator environment / A.U. Batmaz, M. de Mathelin, B. Dresch-Langley // Cogent Med. – 2018. – Vol. 5. – P. 1–29.

33. Dresch-Langley, B. Towards expert-based speed–precision control in early simulator training for novice surgeons / B. Dresch-Langley // Information. – 2018. – Vol. 9. – P. 316.

34. Johansson, R.S. Sensory-motor coordination during grasping and manipulative actions / R. S. Johansson, K. J. Cole // Curr. Opin. Neurobiol. – 1992. – Vol. 2. – P. 815–823.

35. Zatsiorsky, V.M. Multifinger prehension: an overview / V. M. Zatsiorsky, M.L. Latash // J. Mot. Behav. – 2008. – Vol. 40. – P. 446–476.

36. Kinoshita, H. Grip posture and forces during holding cylindrical objects with circular grips / H. Kinoshita, T. Murase, T. Bandou // Ergonomics. – 1996. - Vol. 39. – P. 1163–1176.

37. Kjnoshita, H. Contributions and co-ordination of individual fingers in multiple finger prehension / H. Kjnoshita, S. Kawai, K. Ikuta // Ergonomics. – 1995. - Vol. 38. – P. 1212–1230.

38. Latash, M.L. Multi-finger prehension: control of a redundant mechanical system / M. L. Latash, V. M. Zatsiorsky// In Progress in Motor Control; Springer: Berlin/Heidelberg, Germany, 2009. – P. 597–618.

39. Multi-modal haptic feedback for grip force reduction in robotic surgery / [A. Abiri, J. Pensa, A. Tao et. al] // Sci. Rep. – 2019. – Vol. 9. – P. 1–10.

40. Force-sensing enhanced simulation environment (ForSense) for laparoscopic surgery training and assessment / [T. P. Cundy, E.Thangaraj, H.Rafii-Tari et. al] // Surgery. – 2015. – Vol. 157. – P. 723–731.

41. Dynamic model of a humanoid exoskeleton of a lower limb with hydraulic actuators / [S. Glowinski, M.Obst, S. Majdanik et. al] // Sensors 2021. – Vol. 21. – P. 1-20.

42. Interactive rehabilitation robot for hand function training / [M. Chen, S. Ho, H. F. Zhou et. al] // Rehabilitation Robotics: IEEE International Conference on, Kyoto, Japan, 23–26 June 2009: proceeding. – Los Alamitos: IEEE, 2009. – P. 777–780.
43. Internet of skills, where robotics meets AI, 5G and the Tactile Internet / [M. Dohler, T. Mahmoodi, M. A. Lema et. al] // Networks and Communications (EuCNC): European Conference, Oulu, Finland, 12–15 June 2017: proceedings. – 2017. – P. 1–5.
44. Tactile Internet: with human-in-the-loop / [F. H. Fitzek, S.C.Li, S. Speidel et. al] // Academic Press: Cambridge, MA, USA. – 2021. – 508 p.
45. Tactile internet and its applications in 5g era: a comprehensive review / [R. Gupta, S. Tanwar, S. Tyagi et. al] // Int. J. Commun. Syst. – 2019. – Vol. 32. P. – 1-49.
46. Learning the signatures of the human grasp using a scalable tactile glove / [S. Sundaram, P. Kellnhofer, Y. Li et. al] // Nature. – 2019. – Vol. 569. – P. 698–702.
47. Maximal dynamic grip force and wrist torque: the effects of gender, exertion direction, angular velocity, and wrist angle / [J. L. Morse, M.C.Jung, G. R. Bashford et. al] // Appl. Ergon. – 2006. – Vol. 37. – P. 737–742.
48. Dynamic force-sharing in multi-digit task / [C.E. Dumont, M. R. Popovic, T. Keller et. al] // Clin. Biomech. – 2006. – Vol. 21. – P. 138–146.
49. Edgren, C.S. Grip force vectors for varying handle diameters and hand sizes / C. S. Edgren, R. G. Radwin, C. B. Irwin // Hum. Factors. – 2004. – Vol. 46. – P. 244–251.
50. Parkhomenko, A. Investigation and realization of prototyping technologies for robotic-prostheses computer aided design / A. Parkhomenko, O. Gladkova, Ya. Zalyubovskiy // The Experience of designing and application of CAD systems in microelectronics: International conference, Lviv, February 2019: proceedings. – Los Alamitos: IEEE, 2019. – P. 7/5-7/8.
51. Modern technologies for biomedical systems prototyping / [A. Parkhomenko, O.Gladkova, A. Tulenkov et al.] // Teaching and subjects on bio-

medical engineering. Approachwes and experiences from the BIOART-project. P. Arras and D. Luengo (Eds.) Acco cv, Leuven, Belgium, 2021. – P. 171-185.

52. Интерфейс передачи данных – UART [Электрон. ресурс]. – Режим доступа: <https://3d-diy.ru/wiki/arduino-moduli/interfeys-peredachi-dannykh-uart/>.

53. UART – Последовательный интерфейс передачи данных [Электрон. ресурс]. - Режим доступа: <https://voltiq.ru/wiki/uart-interface/>.

54. UART vs SPI vs I2C Diferencias entre protocolos. [Electronic source]. – Access mode: <https://www.drouiz.com/blog/2018/06/25/uart-vs-spi-vs-i2c-diferencias-entre-protocolos/>.

55. Сравнение SPI- и I²C- интерфейсов [Электрон. ресурс]. – Режим доступа: <http://www.informaticspoint.ru/forpois-368-1.html>.

56. Сравнение последовательных протоколов связи [Электрон. ресурс]. – Режим доступа: <http://digitrode.ru/articles/1311-kratkiy-obzor-populyarnyh-posledovatelnyh-interfeysov-spi-i2c-uart-usart.html>.

57. Создаем компьютерное приложение [Электрон. ресурс]. – Режим доступа: <https://it.wikireading.ru/h3G195K7by>.

58. Arduino/Processing language comparison [Electronic source]. – Access mode: <https://www.arduino.cc/en/Reference/Comparison>.

59. Processing и Arduino [Электрон. ресурс]. – Режим доступа: <https://robocraft.ru/arduino/336>.

60. Блум, Дж. Изучаем Ардуино: инструменты и методы технического волшебства / Дж. Блум. - СПб: БХВ-Петербург, 2015. – 336 с.

61. Плата Arduino NANO v 3.0: распиновка, схемы, драйвер [Электрон.ресурс]. – Режим доступа: <https://arduinomaster.ru/platy-arduino/plata-arduino-NANO/>.

62. Arduino NANO V3.0 — характеристики, распиновка, драйвера, описание платы [Электрон. ресурс]. – Режим доступа: <https://micro-pi.ru/arduino-NANO-v3-0-обзор-драйвера/>.

63. Элементы графического интерфейса [Электрон. ресурс]. – Режим доступа: <https://remotexy.com/ru/help/editor/elements/>.

ДОДАТОК А
Текст програми

A.1 Файл hand.ino

```
#include "Servo.h"

Servo myservo,
myservo2,
myservo3,
myservo4,
myservo5;
int pos1 = 0;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  while (Serial.available() == 0);

  char val = Serial.read() ; // int - '0'

  if (val == '1')
  {

    Serial.println("Стискаемо кулак");
    myservo.attach(8);
    myservo2.attach(11);
    myservo3.attach(9);
    myservo4.attach(10);
    myservo5.attach(12);

    myservo.write(0);
```

```

myservo2.write(0);
myservo3.write(0);
myservo4.write(0);
myservo5.write(180);

Serial.println("Кулак стиснуто ");
delay(500);
myservo.detach();
myservo2.detach();
myservo3.detach();
myservo4.detach();
myservo5.detach();

}

else if (val == '2')
{
  Serial.println("Розтискаємо кулак");
  myservo.attach(8);
  myservo2.attach(11);
  myservo3.attach(9);
  myservo4.attach(10);
  myservo5.attach(12);

  myservo.write(180);
  myservo2.write(180);
  myservo3.write(180);
  myservo4.write(180);
  myservo5.write(0);
  delay(500);

  myservo.detach();
  myservo2.detach();
  myservo3.detach();
  myservo4.detach();
  myservo5.detach();
}

```

```
Serial.println("Кулак розтиснуто");

}

else if (val == '3')

{
    Serial.println("Згинаємо великий палець");
    myservo.attach(8);
    myservo.write(0);
    delay(500);
    myservo.detach();
}

else if (val == '4')

{
    Serial.println("Згинаємо вказівний палець");
    myservo2.attach(11);
    myservo2.write(0);
    delay(500);
    myservo2.detach();
}

else if (val == '5')

{
    Serial.println("Згинаємо середній палець");
    myservo3.attach(9);
    myservo3.write(0);
    delay(500);
    myservo3.detach();
}

else if (val == '6')
```

```

{
    Serial.println("Згинаємо безіменний палець");
    myservo4.attach(10);
    myservo4.write(0);
    delay(500);
    myservo4.detach();
}

else if (val == '7')

{
    Serial.println("Згинаємо мізинець");
    myservo5.attach(12);
    myservo5.write(180);
    delay(500);
    myservo5.detach();
}

else if (val == '8')

{
    Serial.println("Розгинаємо великий палець");
    myservo.attach(8);
    myservo.write(180);
    delay(500);
    myservo.detach();
}

else if (val == '9')

{
    Serial.println("Розгинаємо вказівний палець ");
    myservo2.attach(11);
    myservo2.write(180);
    delay(500);

```

```

        myservo2.detach();
    }

    else if (val == 'a')

    {
        Serial.println("Розгинаємо середній палець ");
        myservo3.attach(9);
        myservo3.write(180);
        delay(500);
        myservo3.detach();
    }

    else if (val == 'b')

    {
        Serial.println("Розгинаємо безіменний палець ");
        myservo4.attach(10);
        myservo4.write(180);
        delay(500);
        myservo4.detach();
    }

    else if (val == 'c')

    {
        Serial.println("Розгинаємо мізинець");
        myservo5.attach(12);
        myservo5.write(0);
        delay(500);
        myservo5.detach();
    }

    else
    {
        Serial.println("Invalid");
    }

```

```

    }
    Serial.flush();
}

```

A.2 Файл hand.pde

```

import processing.serial.*;

int rectX, rectY;
int circleX, circleY;
int circleSize = 40;
int rectSize = 40;

color rectColor, circleColor, baseColor;
color rectHighlight, circleHighlight;
color currentColor;
boolean rectOver = false;
boolean circleOver = false;

PImage img;

Serial serial; // створюємо об'єкт послідовного порта

MyButton btn1 = new MyButton(60, 500, "Згинання великого
пальцю"); // Великий
MyButton btn2 = new MyButton(253, 160, "Згинання вказівного
пальцю"); // Вказівний
MyButton btn3 = new MyButton(376, 125, "Згинання середнього
пальцю"); // Середній
MyButton btn4 = new MyButton(478, 180, "Згинання
безіменного пальцю"); // Безіменний
MyButton btn5 = new MyButton(594, 307, "Згинання мізинцю
"); // Мізинець
MyButton btn6 = new MyButton(300, 700, "Згинання "); //
Стиснення

```

```

        MyButton btn7 = new MyButton(400, 700, "Розгинання "); //
Розтиснення

        MyButton btn8 = new MyButton(145, 565, "Розгинання великого
пальцю"); // Великий

        MyButton btn9 = new MyButton(266, 263, "Розгинання
вказівного пальцю"); // Вказівний

        MyButton btnA = new MyButton(374, 235, "Розгинання
середнього пальцю"); // Середній

        MyButton btnB = new MyButton(462, 273, "Розгинання
безіменного пальцю"); // Безіменний

        MyButton btnC = new MyButton(550, 387, "Розгинання
мізинцю"); // Мізинец

        boolean btn1pres = btn1.btnClicked;

void setup() {
    size(640, 900);
    String port = Serial.list()[0];
    serial = new Serial(this, port, 9600);
    img = loadImage("hand.jpg");
    textSize (18);
}

void draw()
{
    //background(128, 166, 255);
    image(img,0,0, 640,900);

    if (btn1.ButtonDraw())
    { text("Великий палець згинаємо",10,400);
      serial.write('3');
      btn1.btnClicked = false;
    }

    if (btn2.ButtonDraw())
    { text("Вказівний палець згинаємо",10,400);
      serial.write('4');
    }

```

```

        btn2.btnClicked = false;
    }
    if (btn3.ButtonDraw())
    { text("Середній палець згинаємо",10,400);
      serial.write('5');
      btn3.btnClicked = false;
    }
    if (btn4.ButtonDraw())
    { text("Безіменний палець згинаємо",10,400);
      serial.write('6');
      btn4.btnClicked = false;
    }
    if (btn5.ButtonDraw())
    { text("Мізинець згинаємо",10,400);
      serial.write('7');
      btn5.btnClicked = false;
    }
    if (btn6.ButtonDraw())
    { text("Стискаємо кулак",10,400);
      serial.write('1');
      btn6.btnClicked = false;
    }
    if (btn7.ButtonDraw())
    { text("Розтискаємо кулак",10,400);
      serial.write('2');
      btn7.btnClicked = false;
    }
    if (btn8.ButtonDraw())
    { text("Великий палець розгинаємо",10,400);
      serial.write('8');
      btn8.btnClicked = false;
    }
    if (btn9.ButtonDraw())
    { text("Вказівний палець розгинаємо",10,400);
      serial.write('9');
      btn9.btnClicked = false;
    }

```

```

    }
    if (btnA.ButtonDraw())
    { text("Середній палець розгинаємо",10,400);
      serial.write('a');
      btnA.btnClicked = false;
    }
    if (btnB.ButtonDraw())
    { text("Безіменний палець розгинаємо",10,400);
      serial.write('b');
      btnB.btnClicked = false;
    }
    if (btnC.ButtonDraw())
    { text("Мізинець розгинаємо",10,400);
      serial.write('c');
      btnC.btnClicked = false;
    }
  }

class MyButton
{
  float x;
  float y;
  float w;
  float h;
  String info;

  MyButton () {}
  MyButton (float X, float Y, String Info)
  {
    x = X;
    y = Y;
    w = 40;
    h = 40;
    info = Info;
  }
}

```

```

boolean btnClicked = false;
boolean isActive = false;

boolean ButtonDraw()
{
    if (mouseX > x - w/2 && mouseX < x + w/2 && mouseY > y
- h/2 && mouseY < y + h/2) // тыт
    {
        fill(0, 140, 0); //
        if (mousePressed)
        {
            fill(0, 0, 139);
            btnClicked = true;
        }
        else
        {
            text(info,10,400);
        }
    }
    else
    {
        fill(175);
        btnClicked = false;
    }
    ellipse(x,y,w,h);
    return btnClicked;
}
}

```

ДОДАТОК Б
Слайди презентації



Рисунок Б.1 – Слайд №1



Рисунок Б.2 – Слайд №2



Рисунок Б.3 – Слайд №3

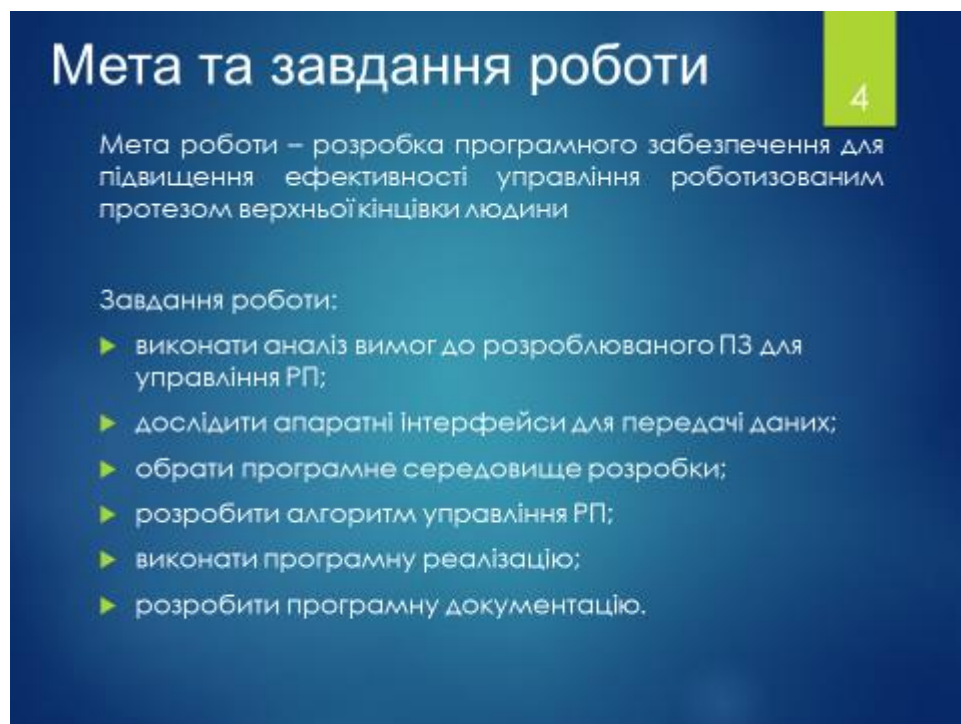


Рисунок Б.4 – Слайд №4



Рисунок Б.5 – Слайд №5

Проектування ПЗ

6

Характеристика	UART	SPI	IPC
Схема			
Переваги	Дуже простий, дозволяє швидко підключити два пристрої, зазвичай використовується, наприклад, RS232 або RS485.	Простий, має повнодуплексний зв'язок, використовує push-pull і високу швидкість передачі даних, вимагає менше енергії, ніж PC.	Більше одного Майстра; для зв'язку знадобиться всього два кабелі; простіше, ніж SPI, оскільки вимагає менше радіо; концепція відкритого колектора; має контроль потоку.
Недоліки	Обмежується зв'язком між двома пристроями; швидкість має бути узгоджена на початку з'єднання, зв'язок; якщо вона не однакова, це дасть помилкові показання.	Чим більше підпорядкованих пристроїв, тим більше потрібно зв'язок; щоб додати пристрій, потрібно додаткове підключення; немає контролю потоку.	Складність схеми збільшуватися при збільшенні кількості ведучих та підлеглих; I2C є напівдуплексним, ПЗ може переважувати процесор.

Порівняльна характеристика інтерфейсів передачі даних

Рисунок Б.6 – Слайд №6

Проектування ПЗ

7

Характеристики	Середовище розробки	
	Arduino IDE	Processing
Простота	+	+
Підтримка	+	+
Мова програмування	C / Wiring	Processing, /Java
Доступність	Open source	Open source

Порівняння та вибір середовища розробки

Рисунок Б.7 – Слайд №7

Проектування ПЗ

7

Характеристики	Середовище розробки	
	Arduino IDE	Processing
Простота	+	+
Підтримка	+	+
Мова програмування	C / Wiring	Processing, /Java
Доступність	Open source	Open source

Порівняння та вибір середовища розробки

Рисунок Б.8 – Слайд №8

Конструювання ПЗ

9

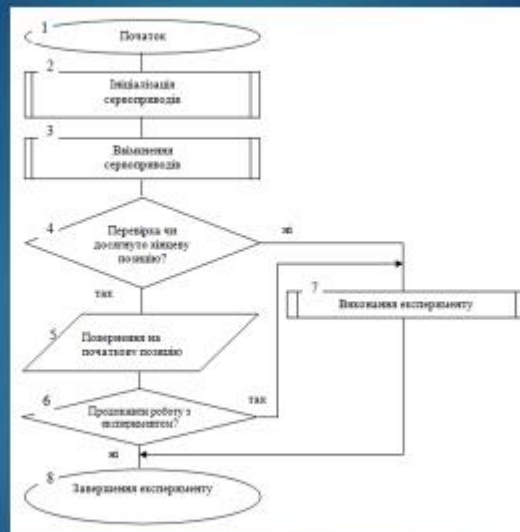


Схема алгоритму програми для Arduino

Рисунок Б.9 – Слайд №9

Інтерфейс програми

10



Рисунок Б.10 – Слайд №10

Висновки

11

В результаті виконання дипломної кваліфікаційної роботи бакалавра:

- ▶ досліджено особливості розробки біонічних та тягових протезів верхньої кінцівки людини;
- ▶ проаналізовано особливості реалізації фізичної та електронної частин роботизованого протезу верхньої кінцівки людини;
- ▶ обрано апаратні та програмні засоби для реалізації управління роботизованим протезом верхньої кінцівки людини;
- ▶ з використанням мови програмування Processing розроблено графічний інтерфейс управління роботизованим протезом шляхом взаємодії з певними контрольними точками для формування базових рухів та захоплень.

Рисунок Б.11 – Слайд №11

12

ДЯКУЮ ЗА УВАГУ!

Рисунок Б.12 – Слайд №12