Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Автоматики і комп'ютеризованих технологій_____
(повна назва)

Кафедра _Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки_
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти___перший (бакалаврський)_____
(рівень вищої освіти)

_____Розроблення програмного забезпечення моделі_____
_____кінематичного 3-ланкового маніпулятора_____
(тема)

Виконав:
здобувач ___3____ року навчання,
групи _____АКТСІу-22-1_____
_____Михайло ЛАЗАРЕНКО_____
(власне ім'я, прізвище)

Спеціальність ____151 Автоматизація та___
комп'ютерно-інтегровані технології_____
(код і повна назва спеціальності)
Тип програми __освітньо-професійна____
Освітня програма __Системна інженерія___
(повна назва освітньої програми)

Керівник _професор Олександр ЦИМБАЛ_
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри _____ _____ ___Ігор НЕВЛЮДОВ___
(підпис) (власне ім'я, прізвище)

2025 р

Підтримка політики закладу з академічної доброчесності

Я, Лазаренко Михайло Юрійович, як здобувач вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

"09" червня 2025 р. Михайло ЛАЗАРЕНКО

Харківський національний університет радіоелектроніки

Факультет___Автоматики і комп'ютеризованих технологій_____

Кафедра_Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки

Рівень вищої освіти_____перший (бакалаврський)_____

Спеціальність_151 Автоматизація та___ комп'ютерно-інтегровані технології
(код і повна назва)

Тип програми _____освітньо-професійна_____

Освітня програма_____Системна інженерія_____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві_____Лазаренку Михайлу Юрійовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення програмного забезпечення моделі кінематичного 3-ланкового маніпулятора_____
Затверджена наказом університету від _21_ __травня__2025 р. № _407 Ст__
2. Термін подання здобувачем роботи до екзаменаційної комісії 17 липня 2025 р.
3. Вихідні дані до роботи __Arduino IDE, Microsoft Visual Studio, Fusion 360, Arduino Uno Rev3, сервоприводи, робот-маніпулятор, кінематична модель, SerialPort_____
4. Перелік питань, що потрібно опрацювати в роботі _Дослідження кінематичної моделі робота-маніпулятора, вибір апаратного забезпечення, вибір програмного забезпечення, розробка 3D моделей маніпулятора та пульта керування, розробка програмного забезпечення для керування роботом-маніпулятором, реалізація передачі даних по SerialPort, презентація роботи_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _Графічний матерал у вигляді презентації формату .pptx в форматі А4 15 с._____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Створення теми кваліфікаційної роботи	24.06 – 30.06.2025	Виконано
2	Огляд методичної і наукової літератури	01.06 – 06. 06. 2025	Виконано
3	Розробка постановки задач згідно теми кваліфікаційної роботи	07.06 – 13.06.2025	Виконано
4	Розробка теоретичної частини кваліфікаційної роботи	14.06 – 19.06.2025	Виконано
5	Розробка практичної частини для отримання результатів	20.06 – 27.06.2025	Виконано
6	Аналіз отриманих результатів	28.06 – 30.06.2025	Виконано
7	Представлення проекту до захисту	31.06 – 01.06.2025	Виконано
8	Оформлення пояснювальної записки, підготовка	02.07 – 04.07.2025	Виконано

Дата видачі завдання _09_ ___червня___ 2025 р.

Здобувач _____ _____Михайло ЛАЗАРЕНКО___
(підпис)

Керівник роботи _____ професор Олександр ЦИМБАЛ
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 106 с., 4 табл., 51 рис., 3 дод., 18 джерел.

ПРОМИСЛОВИЙ РОБОТ-МАНІПУЛЯТОР, GUI, C++, SerialPort, OpenGL.

Об’єкт розробки – процес керування маніпулятором робота.

Предмет розробки – кінематична модель та програмне забезпечення маніпулятора робота.

Метою даної роботи є розробка програмного забезпечення для управління роботом, здатним виконувати конкретні дії на основі заданих сигналів з пульту.

Для досягнення поставлених цілей і завдань в роботі використовувалися CAD, компілятор VS, методи побудови інтерфейсу користувача. Робота має навчальний характер. 3D модель та анімація до ней була створена у програмі Fusion 360. Програмне забезпечення реалізоване у системі програмування Microsoft Visual Studio 2022.

Результати роботи відповідають Цілям сталого розвитку, зокрема Цілі 9 «Промисловість, інновації та інфраструктура» (підцілі 9.1 та 9.4).

THE ABSTRACT

Explanatory note: 106 p., 4 tables, 51 figures, 3 appendices, 18 sources.

INDUSTRIAL ROBOT-MANIPULATOR, GUI, C++, SerialPort, OpenGL.

The object of development is the process of controlling the robot manipulator.

The subject of development is the kinematic model and software of the robot manipulator.

The purpose of this work is to develop software for controlling a robot capable of performing specific actions based on given signals from the remote control.

To achieve the goals and objectives of the work, CAD, VS compiler, and methods for building a user interface were used. The work is educational in nature. The 3D model and animation for it were created in the Fusion 360 program. The software is implemented in the Microsoft Visual Studio 2022 programming system.

The results of the work are consistent with the Sustainable Development Goals, in particular Goal 9 “Industry, Innovation and Infrastructure” (sub-targets 9.1 and 9.4).

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ГВС – гнучкі виробничі системи;

КІТАР – кафедра Комп’ютерно-інтегрованих технологій, автоматизації та робототехніки;

ПЗ – програмне забезпечення;

ПР – промисловий робот;

САПР – система автоматизованого проєктування;

ХНУРЕ – Харківський Національний Університет Радіоелектроніки;

CAD – (Computer-Aided Design) – автоматизоване проєктування;

CAE – (Computer-Aided Engineering) – комп'ютерне проєктування;

CAM – (Computer-Aided Manufacturing) – автоматизоване виробництво;

DOF – (Degree of Freedom) – ступінь свободи;

IoT – (Internet of Things) – інтернет речей;

LED – (Light Emitting Diode) – світлодіод;

LiDAR – (Light Detection And Ranging) – виявлення та визначення дальності світла;

PCB – (Printed Circuit Board) – друкована плата;

RTOS – (Real Time Operating System) – операційна система реального часу;

GUI – (Graphical User Interface) – інтерфейс користувача.

ВСТУП

Роботизація виробничих процесів засобами автоматизації виробництва заснована на застосуванні промислових роботів (ПР). Роботизація є подальшим розвитком процесу автоматизації, оскільки застосування її дозволяє автоматизувати ті виробничі процеси або їхні частини, автоматизація яких найпростішими засобами недоцільна. Метою роботизації виробничих процесів є підвищення техніко-економічних показників роботи підприємства і поліпшення умов праці.

Промисловий робот – автоматична машина, стаціонарна чи пересувна, з виконавчим пристроєм у вигляді маніпулятора, який має декілька ступенів рухомості, і перепрограмовуваним пристроєм програмного керування для виконання у виробничому процесі рухових і керувальних функції.

Промислові роботи є важливими компонентами автоматизованих гнучких виробничих систем (ГВС), які дозволяють збільшити продуктивність праці. Типове застосування роботів стосується таких операцій, як зварювання, фарбування, складання, вибірка та встановлення, пакування, контроль продукції та випробування, котрі виконуються з високою надійністю, швидкістю, і точністю.

Мета роботи – розробка лабораторного макета робота-маніпулятора, з використанням джойстиків, для повного керування роботом та точного відображення положення його кінцівок на екрані.

Об’єкт розробки – процес керування маніпулятором робота.

Предмет розробки – кінематична модель та програмне забезпечення маніпулятора робота.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз існуючих конструкцій роботів маніпуляторів;

– розробити структурну схему макету;

– провести підбір елементної бази.

Робота відповідає дослідженню керування робота у двох режимах: повному та координатному, виведення положення маніпулятора у інтерфейс користувача та керування через нього маніпулятором.

У Додатку А можна ознайомитись з тезою на тему «Ефективність використання роботизованих систем у виробництві» для підтвердження актуальності теми проекту.

Кваліфікаційна робота виконана згідно ДСТУ 3008 – 15 [1] та керуючись навчальним посібником з кваліфікаційної роботи бакалавра [2] та методичними вказівками [3].

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Аналіз структури роботів-маніпуляторів

Роботи-маніпулятори посідають центральне місце у сучасній промисловій автоматизації. Вони здатні виконувати широкий спектр завдань – від транспортування деталей до складних технологічних операцій. Основу їх ефективної роботи становить чітко спроєктована механічна структура, яка забезпечує необхідну точність, гнучкість і надійність при взаємодії з об’єктами виробничого середовища.

Типова структура робота-маніпулятора (рис. 1.1) включає кілька основних компонентів: базу (станину), ланки (сегменти), приводи, з'єднання (суглоби), кінцевий виконавчий елемент (захват або інструмент) та сенсорні системи. Кожен з цих елементів виконує критично важливу функцію, а їх взаємодія визначає загальні можливості системи [4].

Рисунок 1.1 – «Типовий» маніпулятор

Основу маніпулятора становить кінематичний ланцюг, який складається з послідовно з’єднаних ланок, рух яких координується за допомогою

обертових або поступальних з’єднань. Залежно від типу з’єднань, маніпулятори поділяються на ротаційні (R), лінійні (P) або комбіновані системи. Наприклад, конфігурація RRR характерна для багатьох шарнірних маніпуляторів, які імітують рух людської руки.

Одним із ключових параметрів при аналізі структури є ступінь свободи (DoF – degrees of freedom) (рис. 1.2), що визначає кількість незалежних координат, які описують положення кінцевого ефектора в просторі. Для виконання тривимірних задач зазвичай необхідно щонайменше шість ступенів свободи, однак у залежності від конкретної задачі ця кількість може варіюватись [5].

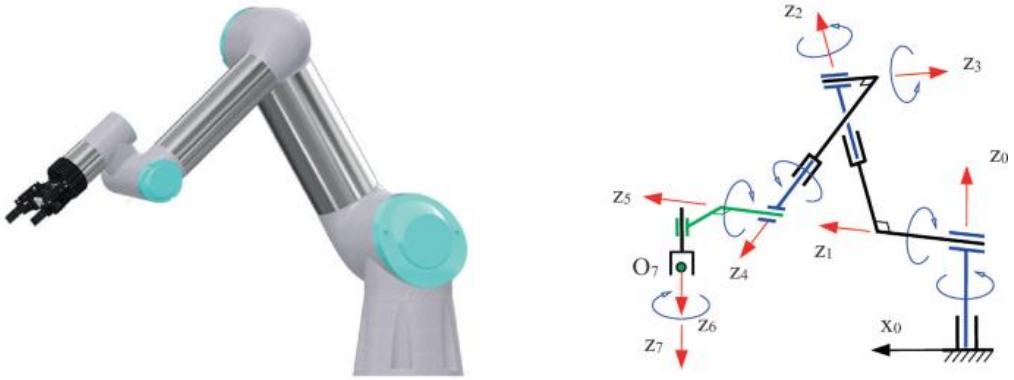

Рисунок 1.2 – 7-DOF (Degree of Freedom)

Згідно з класифікацією, існує кілька основних типів маніпуляторів за кінематичною структурою. Картезіанські маніпулятори (рис. 1.3) мають лінійні приводи по осях X, Y, Z, що забезпечує високу точність і жорсткість, але обмежує гнучкість. Циліндричні й сферичні моделі поєднують лінійні й обертові рухи, забезпечуючи більший робочий об’єм. Шарнірно-зчленовані (articulated) маніпулятори найпоширеніші на сучасному виробництві, оскільки поєднують велику гнучкість, досяжність і універсальність застосування [6].

Рисунок 1.3 – Картезіанський маніпулятор

Важливим аспектом аналізу є динаміка роботи маніпулятора. При високих швидкостях чи великих навантаженнях динамічні навантаження можуть спричиняти вібрації, втрати точності та зниження стабільності. Тому проектування структури вимагає розрахунку моментів інерції, жорсткості з’єднань, маси кожної ланки та характеристик приводів [7].

Сучасні роботи часто містять вбудовані сенсори – інкрементальні або абсолютні енкодери, гіроскопи, акселерометри, датчики сили та моменту. Завдяки цьому досягається точне позиціонування та контроль за динамікою у режимі реального часу. Інтеграція сенсорних систем зі зворотним зв’язком дає змогу маніпуляторам адаптуватися до непередбачуваних ситуацій, наприклад, зміни положення об’єкта або зовнішніх впливів.

Крім механічних характеристик, сучасна структура робота тісно пов’язана з програмним забезпеченням. Віртуальні моделі кінематики, динаміки й середовища експлуатації створюють у спеціалізованих CAD/CAE середовищах і симуляторах (наприклад, RoboDK або ROS), що дозволяє ще на етапі проєктування оптимізувати структуру та рухи маніпулятора.

Таким чином, аналіз структури роботів-маніпуляторів є багаторівневим процесом, що охоплює механічні, кінематичні, динамічні та програмні аспекти. Від

правильного вибору типу структури, конфігурації, ступенів свободи та інтеграції з системою керування залежить успішність функціонування робота в конкретних виробничих умовах.

1.2 Аналіз захватних пристроїв та інструментів сучасних промислових роботів

Автоматизація виробничих процесів дедалі більше потребує гнучких і високоточних рішень. Центральною ланкою у цій трансформації виступають промислові роботи, які виконують широкий спектр завдань – від простих переміщень об’єктів до складних технологічних операцій. Одним із ключових елементів їхньої ефективності є захватні пристрої, які забезпечують фізичну взаємодію з об'єктами в робочому середовищі. Сучасна наука й інженерна практика розглядає ці системи як окрему підсистему роботизованого комплексу, що вимагає детального аналізу та оптимізації.

Захватний пристрій (рис. 1.4) (англ. end effector) виконує функції, аналогічні людській кисті, і може бути реалізований у різних формах залежно від прикладної задачі. Найпоширенішими є механічні гріппери, які використовують принцип стиснення об’єкта між двома або більше "пальцями". Їх основною перевагою є конструктивна простота і універсальність, проте вони потребують точного контролю сили стиснення, щоб уникнути пошкодження крихких предметів [8].

Рисунок 1.4 – Різноманітність захватів для маніпулятора

Для роботи з гладкими або пласкими матеріалами часто застосовуються вакуумні захвати (рис. 1.5), що функціонують за принципом створення розрідженого середовища між захватом і об'єктом. Вони особливо ефективні в логістичних системах і при переміщенні листових матеріалів [9].

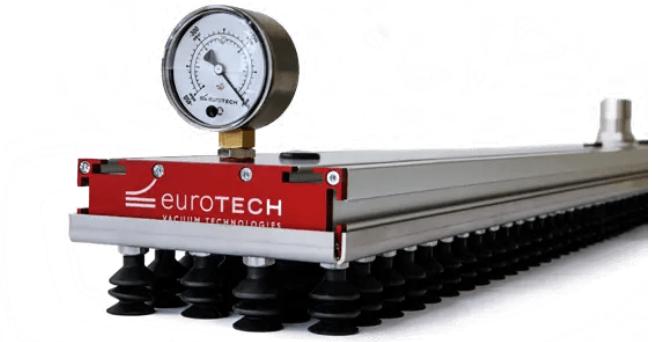

Рисунок 1.5 – Вакуумний захват

Натомість магнітні захвати (рис. 1.6) використовуються для взаємодії з феромагнітними об’єктами, забезпечуючи швидке і надійне утримання без механічного контакту.

Рисунок 1.6 – Маніпулятор з магнітним захватом

Окрему категорію становлять пневматичні системи, які характеризуються швидкою реакцією та простотою конструкції. Їх часто використовують у швидкісних сортувальних або пакувальних процесах. Однак точність позиціонування у таких системах є обмеженою, що знижує їх придатність до задач високої точності [10].

Значним кроком уперед у розвитку захватних технологій стало створення адаптивних гріпперів. Вони здатні підлаштовуватись під форму, розмір і навіть текстуру об'єкта завдяки гнучким елементам і сенсорній системі. Такі пристрої дедалі частіше поєднуються з технологіями машинного зору та штучного інтелекту для досягнення максимальної автономності [11]. Їх застосування особливо важливе в умовах непередбачуваного середовища, наприклад, у харчовій промисловості або при роботі з біооб’єктами (рис. 1.7).

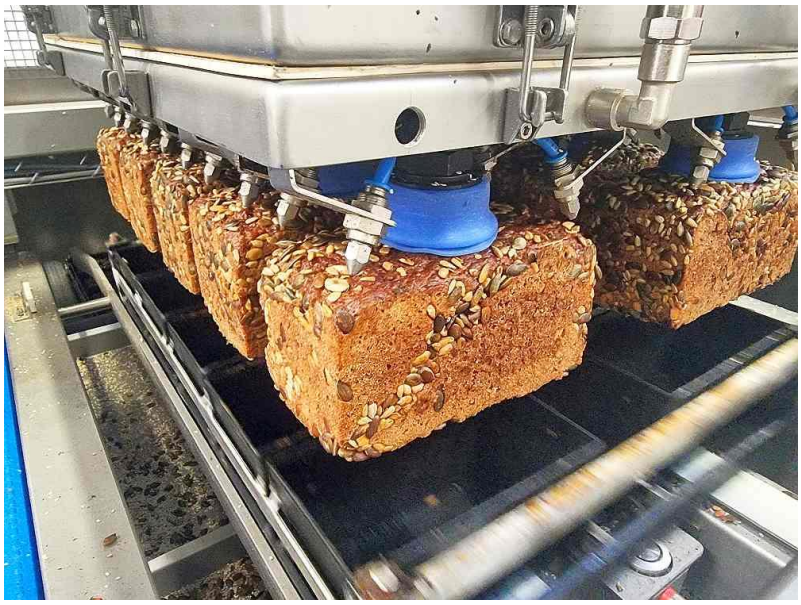

Рисунок 1.7 – Маніпулятор у харчовій промисловості

Крім класичних захватів, роботи часто використовують спеціалізовані інструментальні маніпулятори, що виконують зварювання, свердління, різання або інші технологічні операції. У таких випадках важливішим є не захоплення, а точність, повторюваність і сумісність із виробничими стандартами. Для підвищення універсальності маніпулятори можуть оснащуватись системами автоматичної заміни інструменту, що значно скорочує простої на виробництві [12].

Сучасна тенденція розвитку в галузі захватних пристроїв спрямована на інтеграцію інтелектуальних систем керування, зокрема засобів візуального аналізу, сенсорики та глибокого навчання. Це дозволяє маніпуляторам самостійно визначати найоптимальнішу точку захвату та адаптувати поведінку залежно від контексту задачі [13].

Таким чином, захватні пристрої є не менш важливими за сам маніпулятор у структурі сучасного робота. Їх ефективність визначає межі застосування робототехнічної системи загалом. Подальший розвиток у цьому напрямку пов'язаний із підвищенням гнучкості, універсальності, точності й здатності до самонавчання.

1.3 Аналіз програмного та апаратного забезпечення роботів-маніпуляторів

У сучасній промисловій робототехніці ефективність і функціональність роботів-маніпуляторів визначається не лише їхньою механічною конструкцією, а й якістю програмного та апаратного забезпечення. Ці компоненти є критично важливими для забезпечення точного керування, адаптивності до змін середовища, інтеграції з іншими системами та реалізації складних алгоритмів автоматизації.

До апаратного забезпечення роботів-маніпуляторів належать сенсори, приводи, контролери, плати обробки даних, інтерфейси зв'язку та джерела живлення. Основою будь-якого робота є контролер, який забезпечує зв’язок між програмним забезпеченням та фізичними компонентами. Сучасні контролери зазвичай побудовані на базі високопродуктивних мікропроцесорів або FPGA-систем, що дозволяють реалізовувати як базові, так і складні алгоритми в реальному часі [5].

Для реалізації руху використовуються електроприводи постійного або змінного струму, а також сервоприводи з високою точністю позиціювання. Ці приводи поєднуються з енкодерами (інкрементальними чи абсолютними), що забезпечують зворотний зв'язок для системи керування.

Сенсорні системи дозволяють роботу орієнтуватися в просторі та взаємодіяти з об'єктами. До них належать датчики положення, швидкості, сили та моменту, а також візуальні системи (камери, 3D-сканери). В останнє десятиліття все ширше застосовуються LiDAR-сенсори, що забезпечують тривимірну реконструкцію навколишнього середовища в режимі реального часу [11].

Програмне забезпечення забезпечує інтерпретацію команд користувача, планування траєкторій, обробку сенсорних даних, виконання алгоритмів машинного навчання та забезпечення безпеки. Базові рівні програмного забезпечення зазвичай вбудовані у контролери, однак для складніших завдань використовують операційні системи реального часу (RTOS), а також універсальні програмні середовища, такі як ROS (Robot Operating System).

ROS є де-факто стандартом для досліджень та розробки у галузі робототехніки. Вона надає засоби для побудови розподілених систем, обміну повідомленнями між компонентами, симуляції (наприклад, через Gazebo) та візуалізації даних (Rviz) [14].

Іншим важливим елементом є моделювання та віртуальне тестування рухів робота. Для цього застосовують такі програмні платформи, як RoboDK, V-REP (CoppeliaSim) або MATLAB Robotics Toolbox. Вони дозволяють здійснювати точне моделювання кінематики та динаміки, відлагодження програм, аналіз траєкторій і виявлення потенційних колізій.

Сучасні тренди передбачають активну інтеграцію штучного інтелекту, зокрема машинного навчання та глибоких нейронних мереж. Завдяки цьому роботи здатні навчатися виконанню нових завдань, адаптуватися до непередбачуваних ситуацій та покращувати точність виконання завдань на основі зворотного зв’язку [15].

Також важливо відзначити зростаюче значення хмарних сервісів і IoT у керуванні роботизованими системами. Взаємодія з хмарними платформами дає змогу обробляти великі обсяги даних, зберігати історію роботи, здійснювати віддалене оновлення прошивки та моніторинг стану обладнання.

2 РОЗРОБКА АРХІТЕКТУРИ АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Вибір апаратних модулів

Для розробки системи керування маніпулятором була використана репліка оригінального Arduino Uno Rev3. Виробництво – Китай (рис. 2.1). Від попередніх версій відрізняється мікросхемою USB-UART перехідника (ATMega16u2), яка відрізняється високою швидкістю передачі даних та для якої не потрібно додаткова встановлення драйверів – драйвера встановлюються автоматично при встановленні середи розробки ARDUINO IDE, додатковими контактами SDA і SCL (I2C інтерфейс) й виводами AREF джерела опорної напруги для АЦП контролера та IOREF – виходом напруги живлення портів вводу-виводу (для автоматичного перемикання периферійної напруги за допомогою контролерів 5 В та 3,3 В). У всіх інших аспектах це все той самий контролер Arduino UNO на основі мікроконтролера Atmega328 з великою кількістю прикладів програм, бібліотек та опису будівництва готових конструкцій.

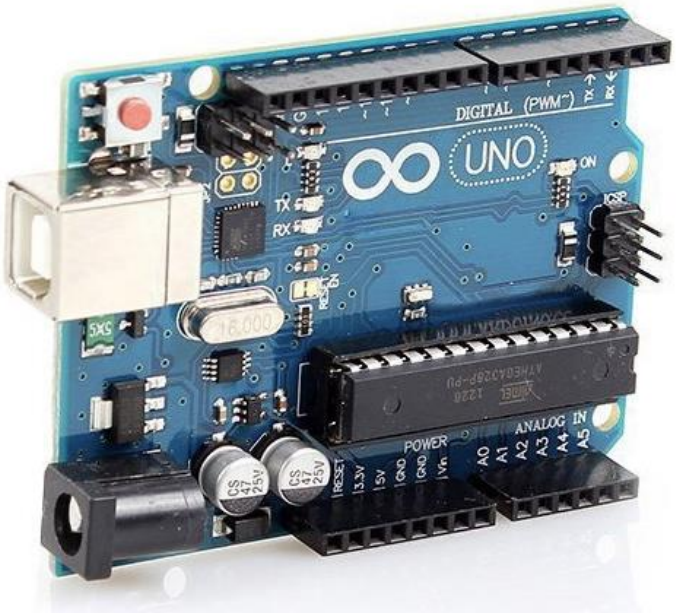

Рисунок 2.1 – Arduino Uno Rev3

Мікроконтролер Atmega16U2 дозволяє використовувати контролер Arduino UNO як USB-пристрій HID. При блиманні цього мікроконтролера контролер може використовуватися як клавіатура, миша або джойстик гри.

Arduino – це відкрита платформа з відкритим кодом програм та бібліотек, яка дозволяє збирати всілякі електронні пристрої. Плата Arduino UNO буде цікавою для творчих працівників, дизайнерів, програмістів та всіх допитливих розумів, які хочуть зібрати свій пристрій або контрольований дизайн (рис. 2.2). Для цієї платформи пишеться величезна кількість різних прикладних програм та бібліотек. Можливо, немає датчиків, дисплеїв та виконавчих механізмів, для яких Arduino не написаний бібліотекою чи програмою, в якій вони використовуються.

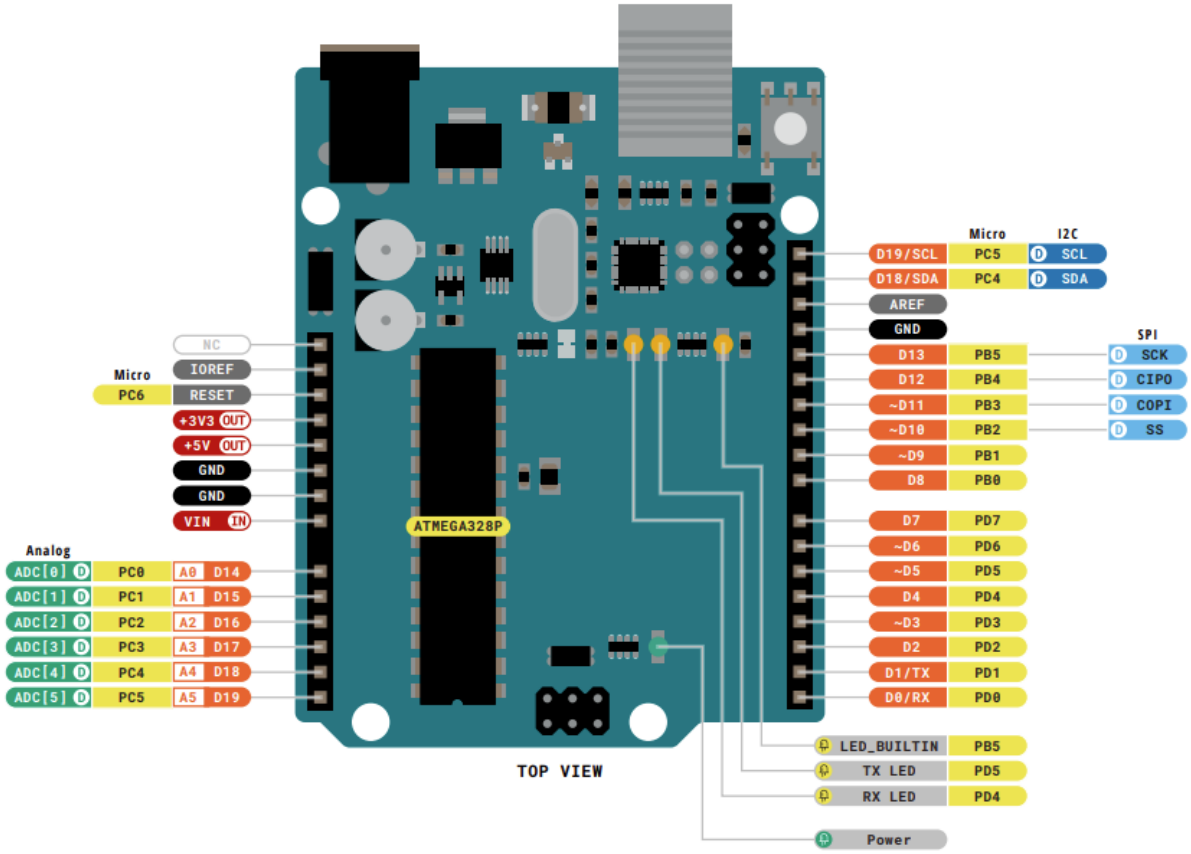

Рисунок 2.2 – Arduino Uno Rev3 розпіновка

Для програмування використовується спрощена версія C++, розробка може здійснюватися за допомогою безкоштовного середовища Arduino IDE та за допомогою довільних інструментів C/C++. Для програмування та спілкування з комп'ютером вам знадобиться USB-кабель, а для автономної роботи вам знадобиться джерело живлення для 7,5 мм - 12 мм у роз'ємі 5,5 мм х 2,1 мм.

Характеристики мікроконтролера Atmega328:

- USB -вхід: 5 В;
- вхід VCC: 5 В;
- вхід VIN: 7,5 В - 12 В;
- цифрові входи/виходи: 14 (6 з них ШІМ);
- аналогові входи: 6;
- пам'ять Flash: 32 КБ;
- ОЗП: 2 КБ;
- частота: 16 МГц;
- розмір: 68 мм х 53 мм х 15 мм;
- виробництво: Китай.

Сенсор-шилд для Arduino UNO (рис. 2.3, 2.4) призначений для збільшення кількості контактних груп та дозволяє підключити велику кількість різноманітних модулів: сенсори, серво-приводи, реле, кнопки, потенціометри та ін. У новій версії окремо виведені роз'єми для підключення аналогових датчиків, а також роз'єми: UART, I2C, SPI, APC220, стандартних Bluetooth-модулів та ультразвуковий інтерфейс URF01.

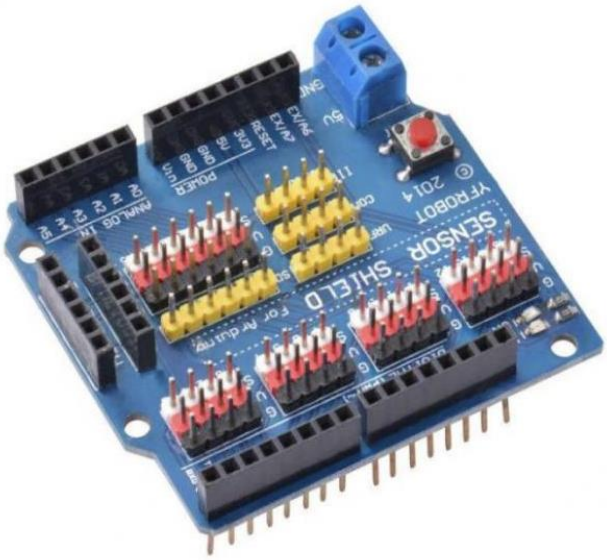

Рисунок 2.3 – Шилд підключення датчиків (сервоприводів)

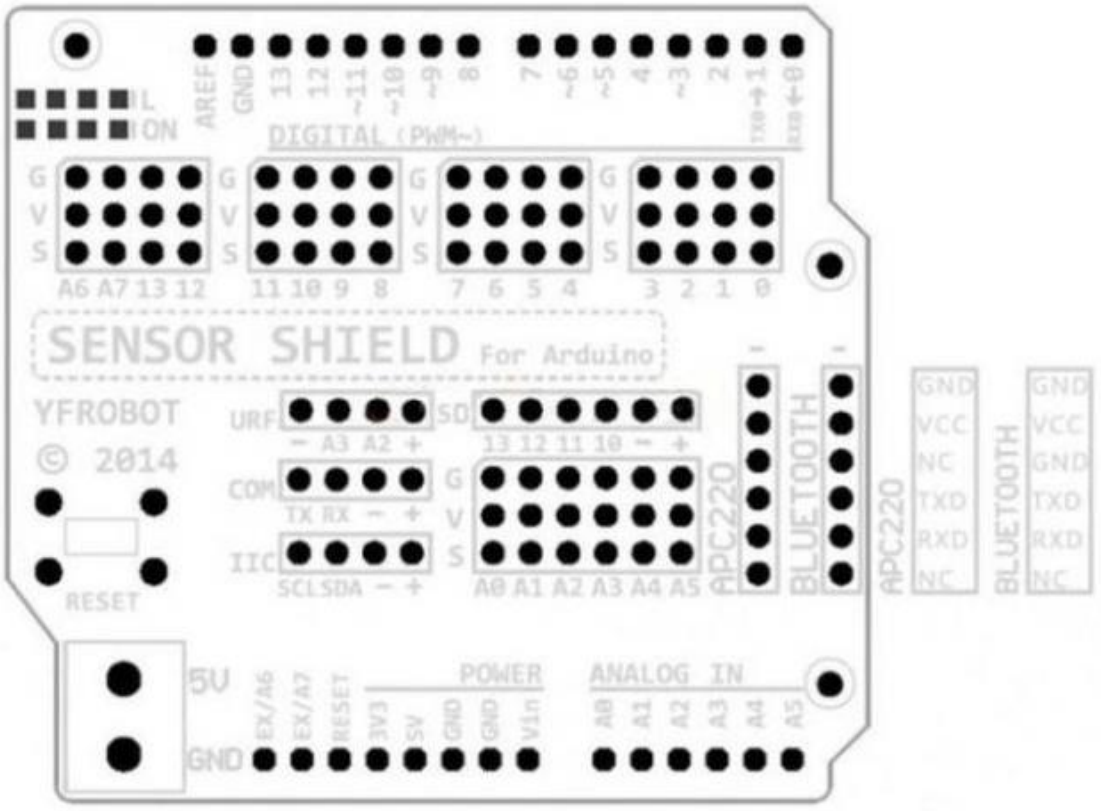

Рисунок 2.4 – Плата розширення для Arduino Uno Rev3

На платі передбачений роз'єм для використання зовнішнього джерела живлення (тільки 5 В!). Також на всі основні контакти проведена лінія живлення. Додатково є кнопка скидання та індикатор живлення. Для зручного

сприйняття, крім стандартних позначення на платі модуля, зроблені кольорові контакти.

Призначення груп роз'ємів:

- SD card (SPI) – послідовний інтерфейс, перший контакт має позначення «+» (інші: «-», D10, D11, D12, D13). призначений для підключення карти пам'яті, на яку буде записуватися робота пристрою, показання сенсорів або інших пристроїв потребуючих швидкісного обміну даними;
- URF (URF01) – призначений для підключення ультразвукового датчика. Також його можна використовувати для інших аналогових датчиків, перший вивід позначається «+» (інші: A2, A3, «-»);
- Power terminal – призначений для підключення зовнішнього живлення. Напруга зовнішнього живлення – 5 Вольт. Додаткове живлення необхідно в разі, коли до шилда підключено багато периферійних модулів;
- Analog IO – (інтерфейс контактів, які можуть працювати як вхід або вихід) позначаються A0, A1, A2, A3, A4, A5 і мають по три контакти, позначених V (живлення), G (мінус), S (сигнал);
- Digital IO – (цифрові піни контролера) вони можуть працювати як вхід і вихід. Позначено 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, GND, AREF. Кожен порт має по три контакти, позначених V, G, S. Вивід AREF – це опорна напруга для аналогових входів. Всього колодка цифрових виходів дозволяє підключити 16 пристроїв;
- IIC (I2C або TWI) – призначений для зв'язку інтегральних схем, він же є послідовною шиною. Має чотири контакти, позначених SCL (Serial Clock), SDA (Serial Data), «-», «+». Завдяки цьому порту з'являється можливість підключати додатковий мікроконтролер і робити з ним обмін даними;
- COM – (шина UART) за допомогою якого можна передавати дані на комп'ютер через USB порт. Інтерфейс має чотири контакти: TX, RX, «-», «+».

Мікросервопривід MG90S 180° (рис. 2.5), розміри 3,2 см х 3,5 см х 1,3 см. Крутний момент 1,6 кг/см (при живленні 4,8 В), 1,8 кг/см (при живленні 6 В).

Рисунок 2.5 – MG90S 180° (3 штуки)

Характеристики:

- модель: MG90S;
- робоча напруга: 4,8 В - 6 В;
- рекомендована робоча напруга: 4,8 В;
- початковий момент: до 1,8 кг/см при 6 В / до 1,6 кг/см при 4,8 В;
- кут повороту: 180°;
- матеріал редуктора: внутрішні передачі – пластиковий, зовнішній метал;
- час обертання 60° (при 4,8 без навантаження): 0,1 с;
- час обертання 60° (при 6 без навантаження): 0,09 с;
- розміри: 32 мм х 32 мм х 12,5 мм;
- діаметр для насадки: 4 мм;
- довжина кабелю: 25 см;
- матеріал: пластик;
- вага набору: 15 г;
- вага сервоприводу з побудованим кабелем: 12 г.

Сервопривід Tower Pro MG995 (рис. 2.6) – клон популярного серводвигуна. Управління не на цифровому микроконтроллере як у оригінальній, а на аналоговій мікросхемі (як правило KC2462 або AA51880), за рахунок чого вдалося істотно знизити ціну при тих же параметрах приводу. Потужний сервомотор – крутний момент до 13 кг/см (при напрузі живлення 6 В).

Рисунок 2.6 – MG995 для захвату

Характеристики:

- редуктор: пластик (нейлон);
- кут повороту: 180 градусів;
- робоча швидкість: 0,17 с / 60 градусів (4,8 В без навантаження);
- робоча швидкість: 0,13 с / 60 градусів (6 В без навантаження);
- пусковий момент: 8,5 кг / см при живленні 4,8 В;
- пусковий момент: 10 кг / см при живленні 6 В;
- робоча напруга: 4,8 В – 7,2 В;
- робоча температура середовища: 0 ~ +55 °C;
- діаметр вала: 5,7 мм;
- довжина дроту 30 см;
- розміри 40,7 мм x 19,7 мм x 43 мм;
- маса 55 г.

Серводвигун MG996R (рис. 2.7) широко використовується в автомоделях для керування поворотом передніх коліс, в авіамоделізмі для повороту керма і закрилків моделі. Серводвигун MG996R застосовується для повороту деталей різних механізмів. Завдяки редуктору з металевими шестернями на вихідному валу розвивається достатнє зусилля для застосування в рухомих роботах. Цей серводвигун є покращеною версією моделі MG995, за точністю та швидкістю позиціювання. Всі технічні характеристики такі ж, як у MG995, але новий серводвигун набагато точніший і безпечніший у використанні на літаках, що потребують підвищеної точності руху керма.

Рисунок 2.7 – MG996R для обертання на 360 градусів

Усередині корпусу знаходиться невеликий модуль керування, який під дією вхідного сигналу подає живлення відповідної полярності електродвигуну. Інформація про необхідний напрямок та швидкість обертання валу міститься в скважності імпульсів керуючого сигналу. Частота сигналу, що управляє, повинна бути постійна і становити 50 Гц. Скважність – відношення тривалості імпульсу до періоду. Найчастіше при аналізі параметрів сигналу, що управляє, розглядають тривалість імпульсу. Для

формування такого сигналу зручно використовувати мікроконтролер, що має функцію широтно-імпульсної модуляції вихідного сигналу.

Працюючи над проектами з урахуванням контролера Arduino при програмуванні МК використовують спеціальну програмну бібліотеку функцій Servo.

Призначення контактів серводвигуна:

- коричневий: мінус живлення;
- червоний: плюс живлення;
- помаранчевий: управління.

Характеристики:

- матеріал редуктора: метал;
- тип серводвигуна: цифровий;
- напруга живлення: 4,8 В;
- кут повороту: 360° (без обмежувача, обертається весь час);
- швидкість повороту: 0,17 с/60° при 4,8 В;
- зусилля на валу: 9,4 кг/см при 4,8 В;
- розміри: 41 мм х 20 мм х 43 мм;
- вага: 55 г.

Джойстик (рис. 2.8) має 5 виводів: VCC (+ 5 В), GND (земля), X (переміщення по осі X), Y (переміщення по осі Y) і Button (сигналізує чи натиснута кнопка). Підключіть живлення і знімайте з виводів X, Y і Button статус поточного становища джойстика. На виходах X і Y сигнал диференційований, тобто змінюється від кута нахилу ручки.

Рисунок 2.8 – Джойстик для Arduino

При використанні напруги живлення 5 В, за замовчуванням на аналогових виводах X і Y буде 2,5 В. При переміщенні джойстика в одну сторону напруга буде наростати до 5 В, при русі джойстика в іншу сторону напруга буде падати до 0 В. Виводи підключаються до аналогових входів Arduino. Таким чином можна отримувати точне положення ручки джойстика і реагувати на кут нахилу, а не тільки на сам факт нахилу ручки.

Характеристики:

- розміри плати джойстика: 24 мм x 36 мм;
- відстань між кріпильними отворами: 27 мм x 21 мм;
- діаметр отворів для кріплення: 3 мм.

Комплект перемичок для макетування тато-тато, мама-мама, мама-тато довжиною 20 см (рис. 2.9). Загальна кількість 120 шт.

Рисунок 2.9 – Комплект перемичок мама-мама, тато-тато, мама-тато 120 шт. 20 см

«Клешня» робота зроблена з легкого металу (рис. 2.10). Клешня відкривається приблизно на 55 мм і в залежності від використовуваного серво-двигуна може піднімати досить важкі предмети. Суглоби надійно скріплені гвинтами з прогумованими контргайками.

Рисунок 2.10 – «Клешня»

Параметри:

- матеріал: алюміній;
- вага: 60 г (без сервоприводу і втулки);

- максимальний кут відкриття: ≈ 50 мм;
- довжина: ≈ 100 мм (в закритому стані);
- ширина: ≈ 90 мм (у відкритому стані).

Монтажна втулка використовується для кріплення до валів двигунів і сервоприводів, наприклад MG995, MG996, MG996R, MG945, S3003, TR213, DS3115, або інших, а також для різних механізмів, коліс, шестерень (рис. 2.11). У комплекті йдуть 4 гвинтика М3.

Рисунок 2.11 – Монтажна втулка

Характеристики:

- матеріал: алюміній;
- колір: металік;
- кількість зубців: 25;
- кріплення: гвинт М3;
- відстань між отворами (по осі): 14 мм;
- маса: 2 г;
- зовнішній діаметр: 20 мм;
- висота: 4,5 мм;
- внутрішній діаметр: 5,3 мм;
- товщина фланця: 2,5 мм.

Акумулятор універсальний Baseus 30000 мАг Bipow Digital Display 15 В Black (рис. 2.12) використовується для живлення периферійних пристроїв (5

сервоприводів), підключається через саморобний кабель: USB Type A – 2 жили до терміналу живлення на 5 В (Power terminal).

Рисунок 2.12 – Повербанк для додаткового живлення (5 В, 3 А)

Розміри та кількість задіяних болтів та шурупів:

- М1 х 11 мм – 6 шт. (болт), кріплення насадки до надрукованого корпусу;
- М2 х 4 мм – 3 шт. (болт), кріплення насадки до мікросерво;
- М2 х 8 мм – 6 шт. (шуруп), кріплення мікросерво до корпусу;
- М2 х 10 мм – 4 шт. (шуруп), гніздо на 4 отвори серво на 360;
- М2 х 14 мм – 2 шт. (болт), кріплення схвату до надрукованої кисті корпусу;
- М3 х 4 мм – 7 шт. (болт), для кріплення втулок (на серво 360 та 180);
- М3 х 5 мм – 5 шт. (болт), кріплення втулки до шестерні-важіля схвату, кріплення плати до стійок;
- М3 х 8 – 8 шт. (болт), та 8 гайок М3 до них для джойстиків.

2.2 Розробка архітектури системи керування маніпулятором

На цій схемі (рис. 2.13) відображено, як компоненти під'єднуються до Arduino Uno Rev3 через Sensor Shield V5.0 для керування роботом-маніпулятором з 5 сервомоторами та двома джойстиками, також у таблиці 2.1 можна побачити з’єднання кожного піна на сенсор шилді.

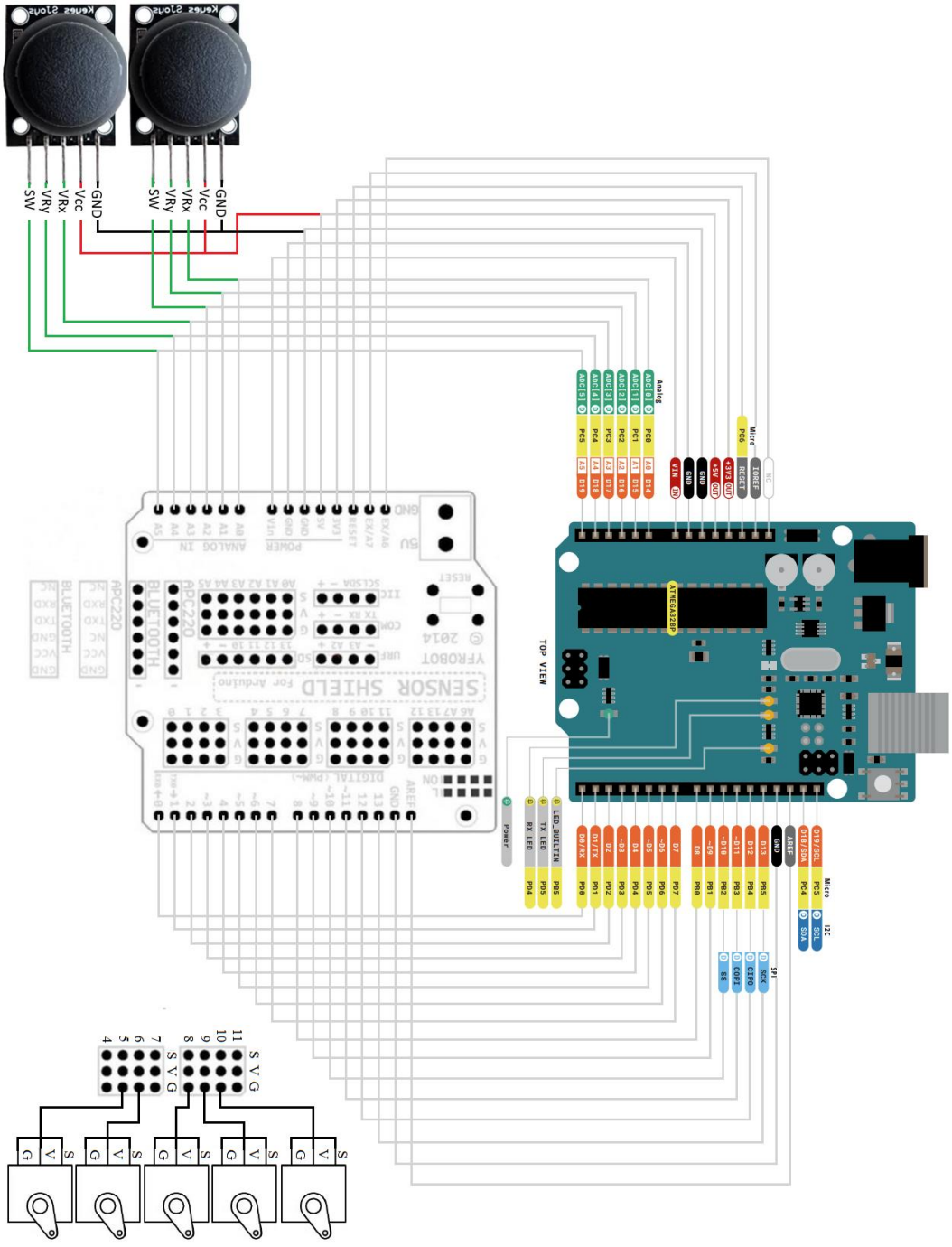

Рисунок 2.13 – Принципова схема

Таблиця 2.1 – Загальне підключення до пінів та їх функція.

Джойстик 1 VRx	A0	Рух X
Джойстик 1 VRy	A1	Рух Y
Джойстик 1 SW	A2	Кнопка
Джойстик 2 VRx	A3	Рух X
Джойстик 2 VRy	A4	Рух Y
Джойстик 2 SW	A5	Кнопка
Серво 1 (Base)	D11	Оберт основи
Серво 2 (Shoulder)	D10	Підйом/опуск плеча
Серво 3 (Elbow)	D9	Згин/розгин ліктя
Серво 4 (Rotator)	D8	Оберт кисті
Серво 5 (Gripper)	D7	Відкриття/закриття клешні

На рисунку 2.14 зображено структуру підключення усієї апаратної частини між собою.

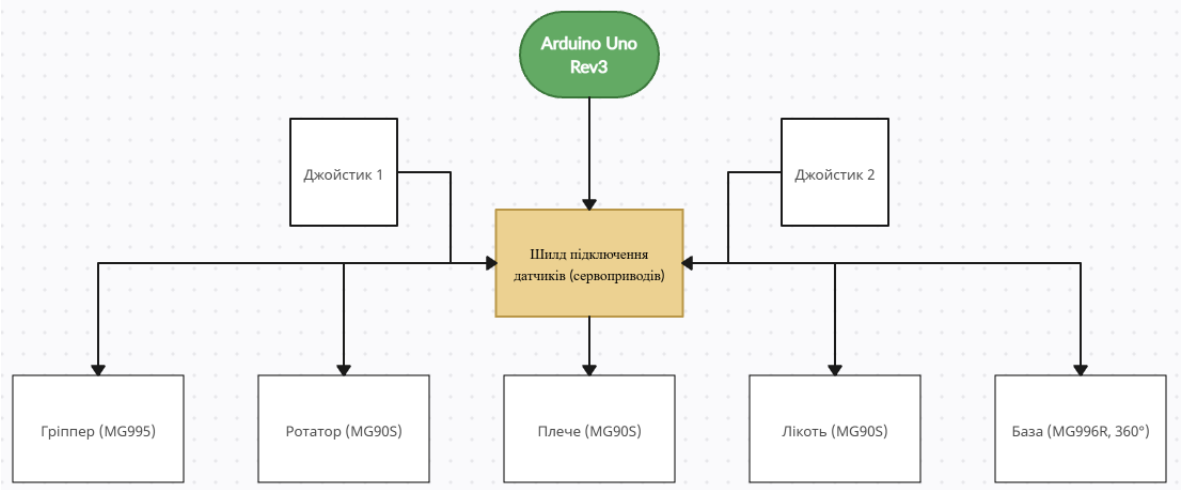

Рисунок 2.14 – Архітектура системи керування роботом-маніпулятором

Sensor Shield V5.0 – це модуль розширення, який полегшує підключення зовнішніх пристроїв до Arduino, зокрема сервомоторів, датчиків і джойстиків. Sensor Shield встановлюється зверху на Arduino Uno через штифти.

Всі цифрові піни D0 – D13 і аналогові A0 – A5 виведені у вигляді 3-контактних роз’ємів (S = signal, V = VCC, G = GND).

Це дозволяє легко підключати серво, джойстики без паяння.

Робот-маніпулятор має два аналогових джойстика. Кожен джойстик має 5 виводів:

- GND (чорний): земля → GND на Sensor Shield;
- VCC (червоний): живлення 5 В → VCC на Sensor Shield;
- VRx (зелений): вісь X → підключено до A0 / A3 (ліворуч на схемі);
- VRy (зелений): вісь Y → підключено до A1 / A4;
- SW (зелений): кнопка натиску джойстика → A2 / A5.

У правій частині схеми – 5 сервомоторів. Вони підключаються до цифрових пінів Arduino через Sensor Shield.

Кожен сервопривід має 3 дроти:

- S (білий/жовтий) – сигнал (PWM), підключений до відповідного цифрового піна: D11, D10, D9, D8, D7;
- V (червоний) – VCC (5 В), йде з Sensor Shield;
- G (чорний) – GND, загальна земля.

Sensor Shield може подавати живлення на серво через VCC. Це 5 В з Arduino або зовнішнє джерело (через клему VIN).

3 РОЗРОБКА 3D МОДЕЛЕЙ МАНІПУЛЯТОРА ТА ПУЛЬТА КЕРУВАННЯ У CAD СИСТЕМІ

3.1 Система Fusion 360 та її використання

Fusion 360 (Фьюжн 360) – комерційне програмне забезпечення (рис. 3.1), розроблене Autodesk і включає системи автоматизованого проектування (CAD), автоматизованого виробництва (CAM), автоматизованої інженерії (CAE) та проектування друкованих плат (PCB). Воно доступне на операційних системах Windows і MacOS, а також у вигляді спрощених програм на Android і IOS. Ліцензія на використання Fusion 360 доступна за платною підпискою, але також існує обмежена безкоштовна домашня некомерційна версія для особистого використання.

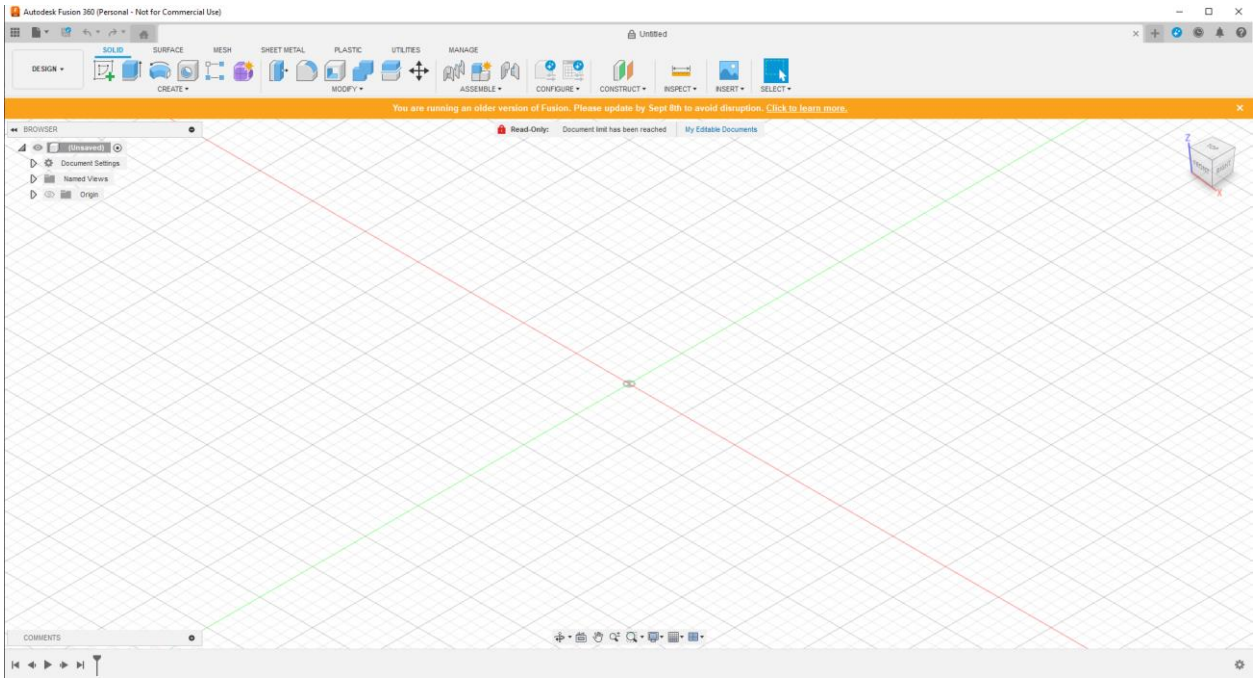

Рисунок 3.1 – Інтерфейс Fusion 360

Fusion 360 має вбудовані можливості для 3D-моделювання, симуляції та документації. Програма може керувати виробничими процесами такими як: механічна обробка, фрезерування, токарна обробка та адитивне виробництво.

У ній також є функції автоматизації проектування електроніки, такі як проектування схем, друкованих плат та керування радіокомпонентами.

На рисунках 3.2 – 3.12 зображено 3D моделі робота-маніпулятора та його частин корпусу у розмірі 1:1, розмірність у міліметрах. На рисунках 3.13 – 3.19 зображено креслення деталей робота-маніпулятора.

На рисунку 3.2, 3.3 зображено робота-маніпулятора у повній комплектації у стані бездії/спокою, або перевезення.

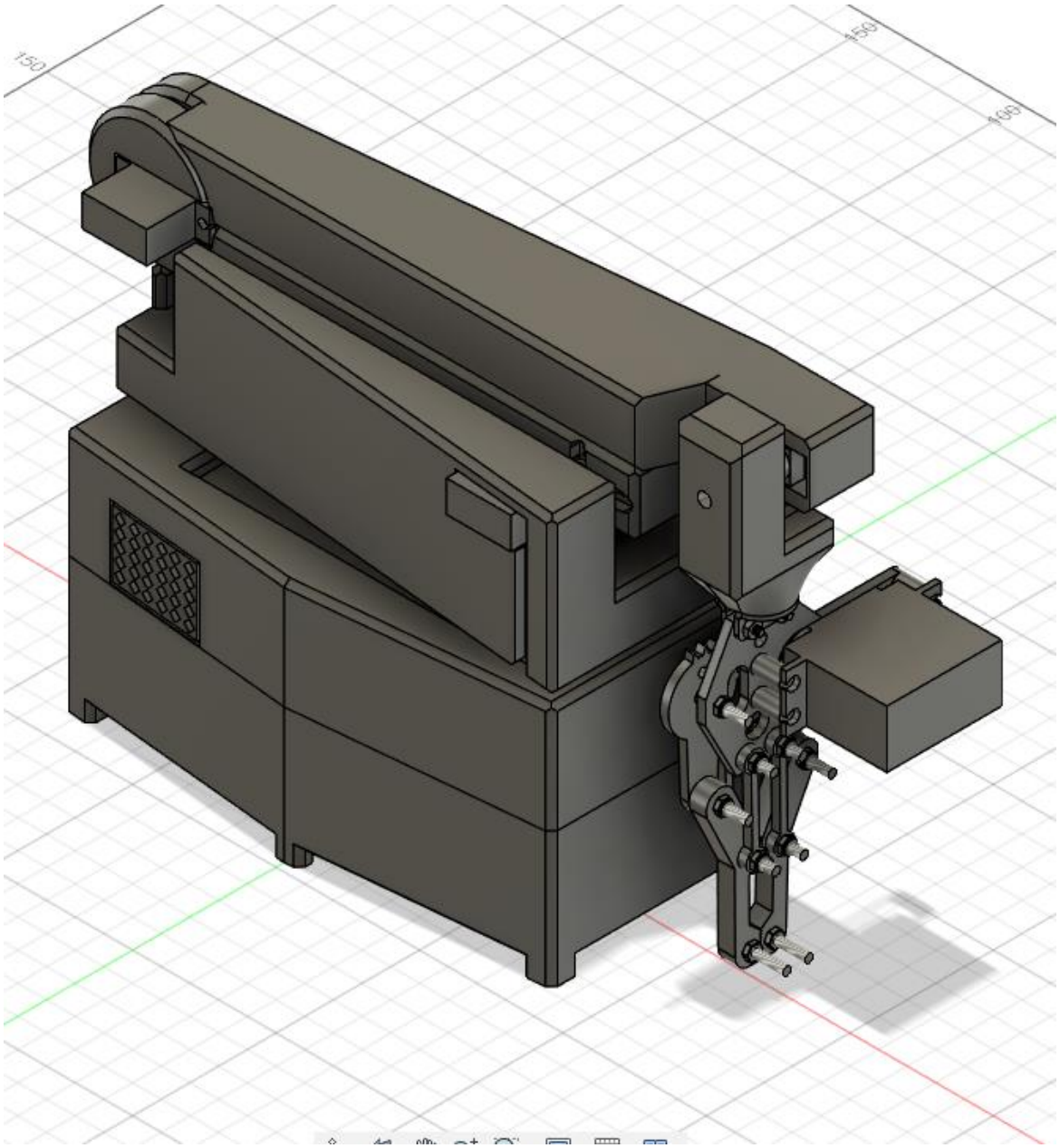

Рисунок 3.2 – Ізометричний вид на корпус маніпулятора в стані «сну» (Вид 1)

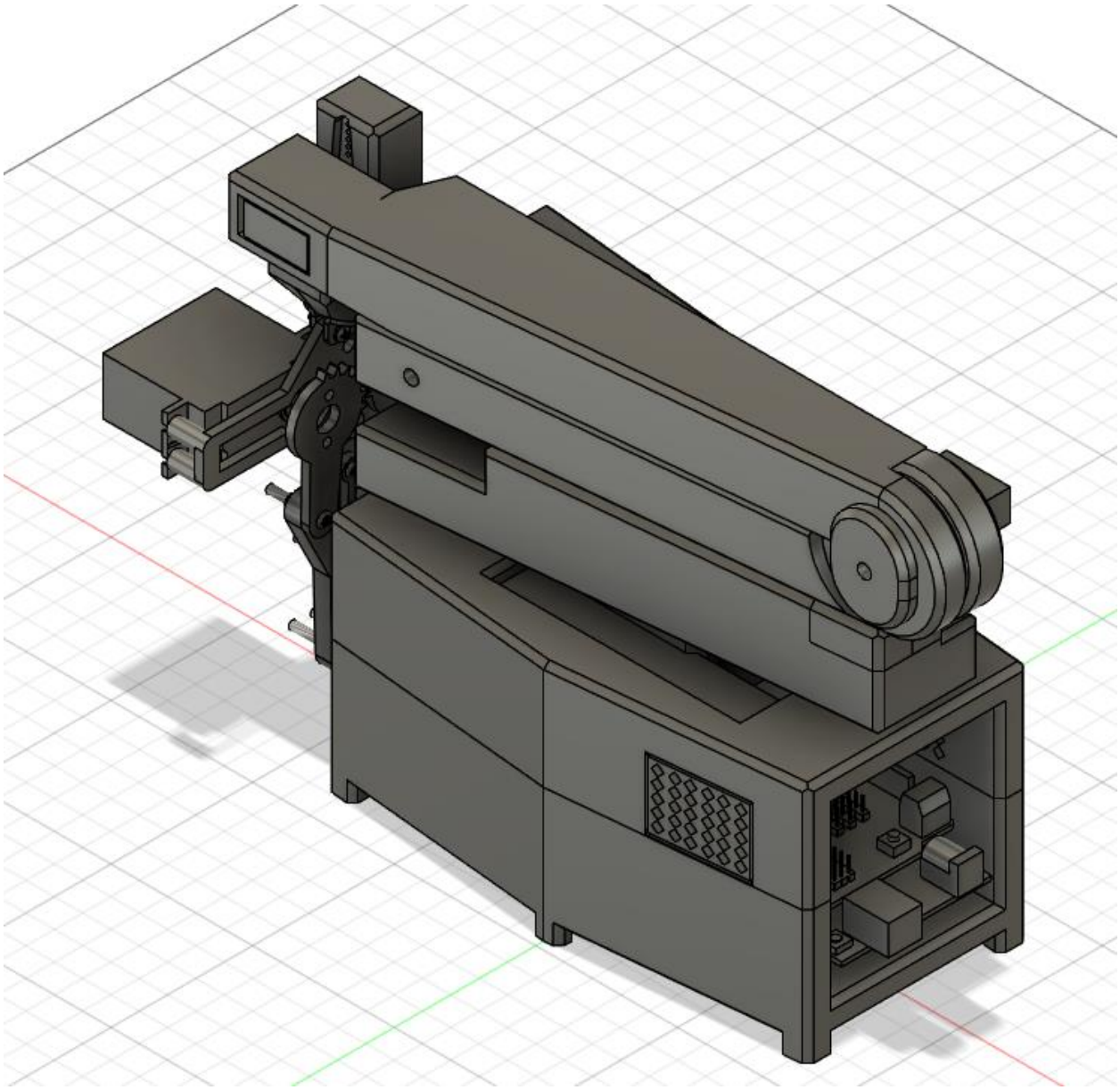

Рисунок 3.3 – Ізометричний вид на корпус маніпулятора в стані «сну» (Вид 2)

На рисунку 3.4 зображено мікроконтролер (Ardiono Uno Rev3) та плату розширення (Sensor shield) у виді надбудови/башти, вставлено в спеціальнийй простір на латунних ніжках (для кращого охолодження), також можна побачити вентеляційні отвори на боках корпусу, ще можна побачити отвір, що трішки вище за плату роширення, для сервоприводу, що виконує функцію «колони» маніпулятора та обертає її на 360 градусів.

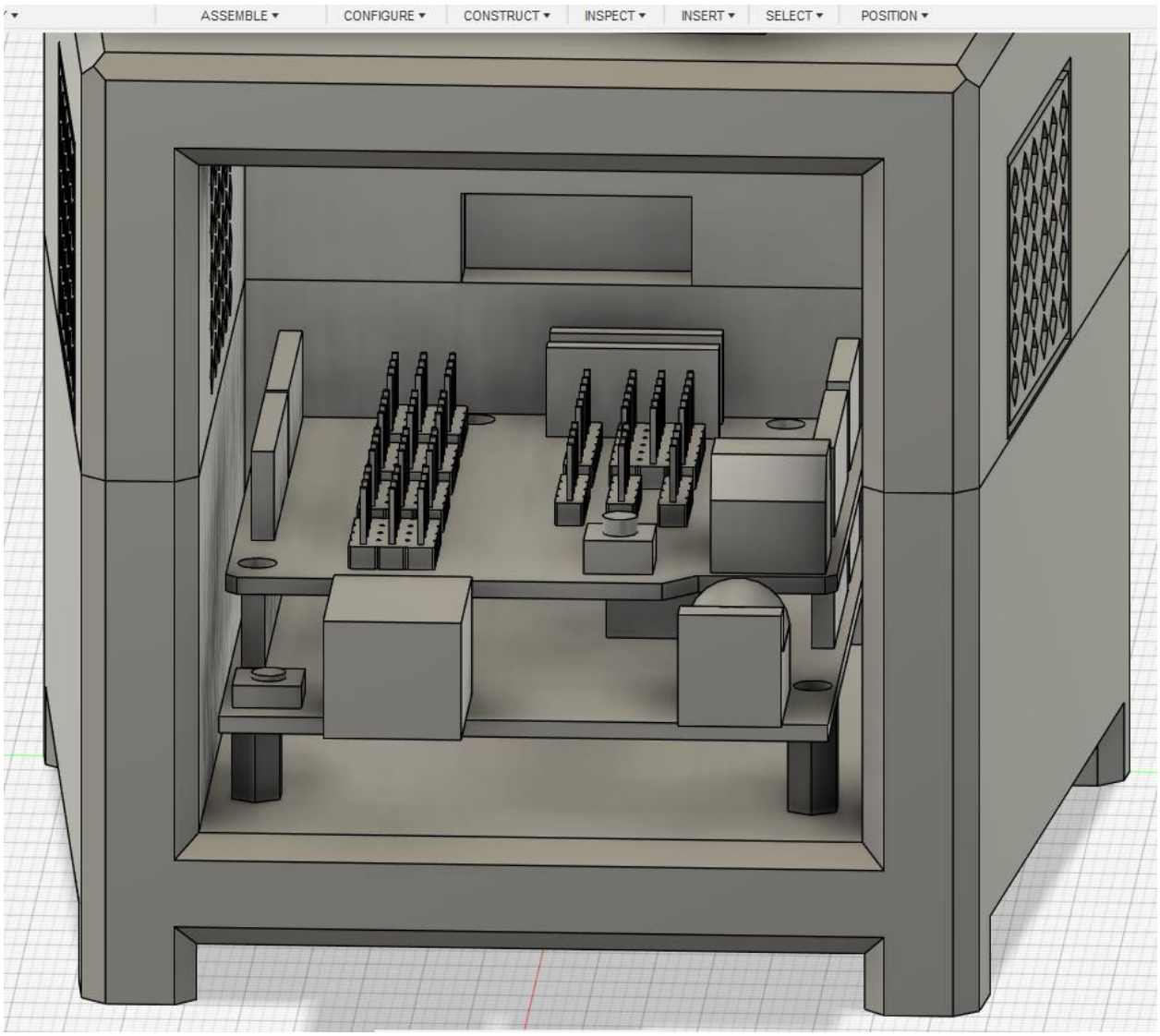

Рисунок 3.4 – Вид на апаратне забезпечення в корпусі маніпулятора

На рисунку 3.5 зображено задіяні частини робота-маніпулятора у вертикальному просторі, відносно маніпулятора (не задіяно «колону»).

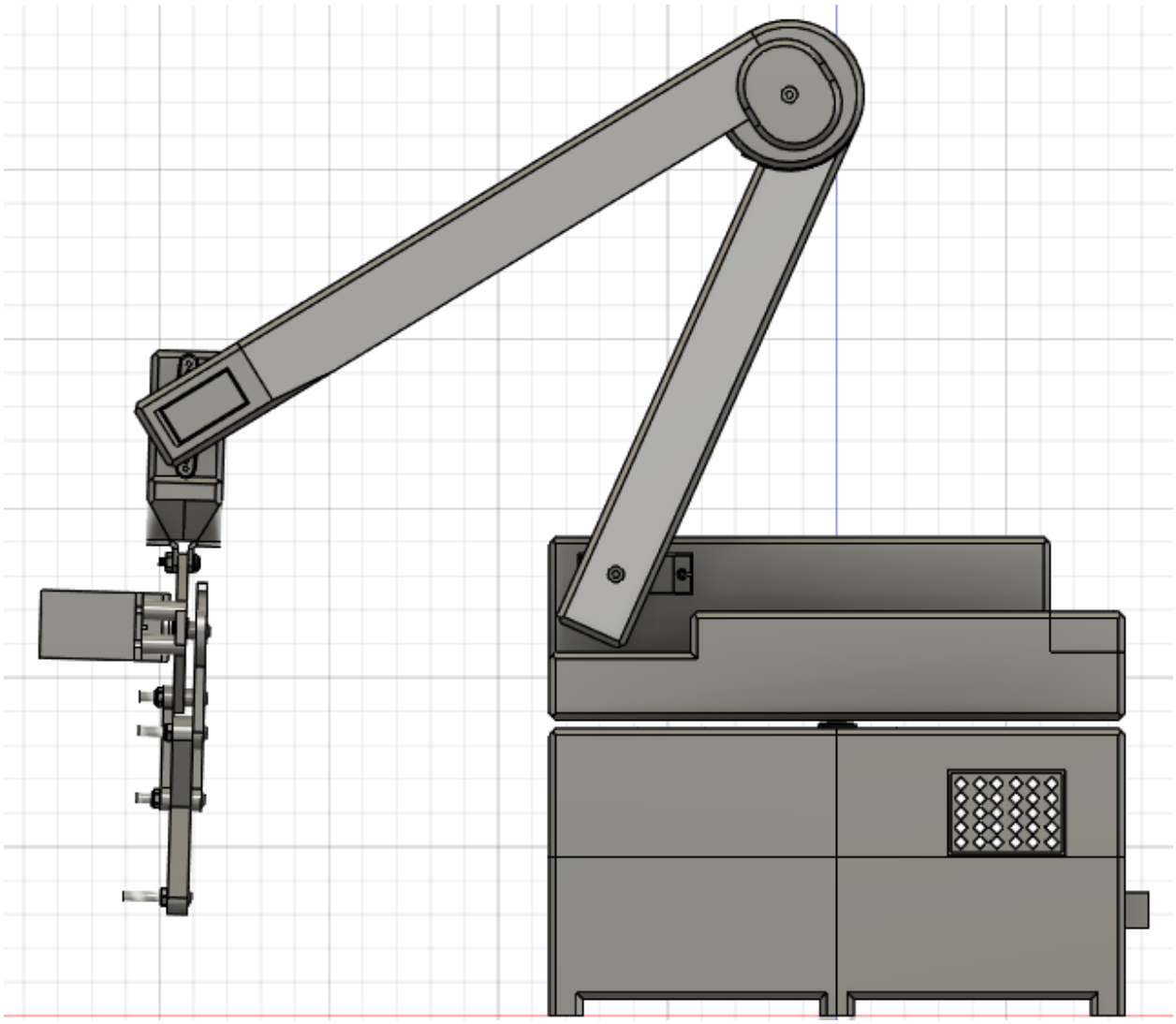

Рисунок 3.5 – Вид на маніпулятор зі сторони FRONT

На рисунку 3.6 можна побачити як задіяна «колона» обертає інші деталі робота-маніпулятора і це не заважає руху. Тому маніпулятор має робочий простір на 360 градусів. Мінімальна відстань початкового робочго простору (при задіянні тільки схвату) 37 мм від корпусу до точки утримання вантажу схватом. 65 мм від корпусу до точки утримання вантажу схватом, якщо повернути на 90 градусів.

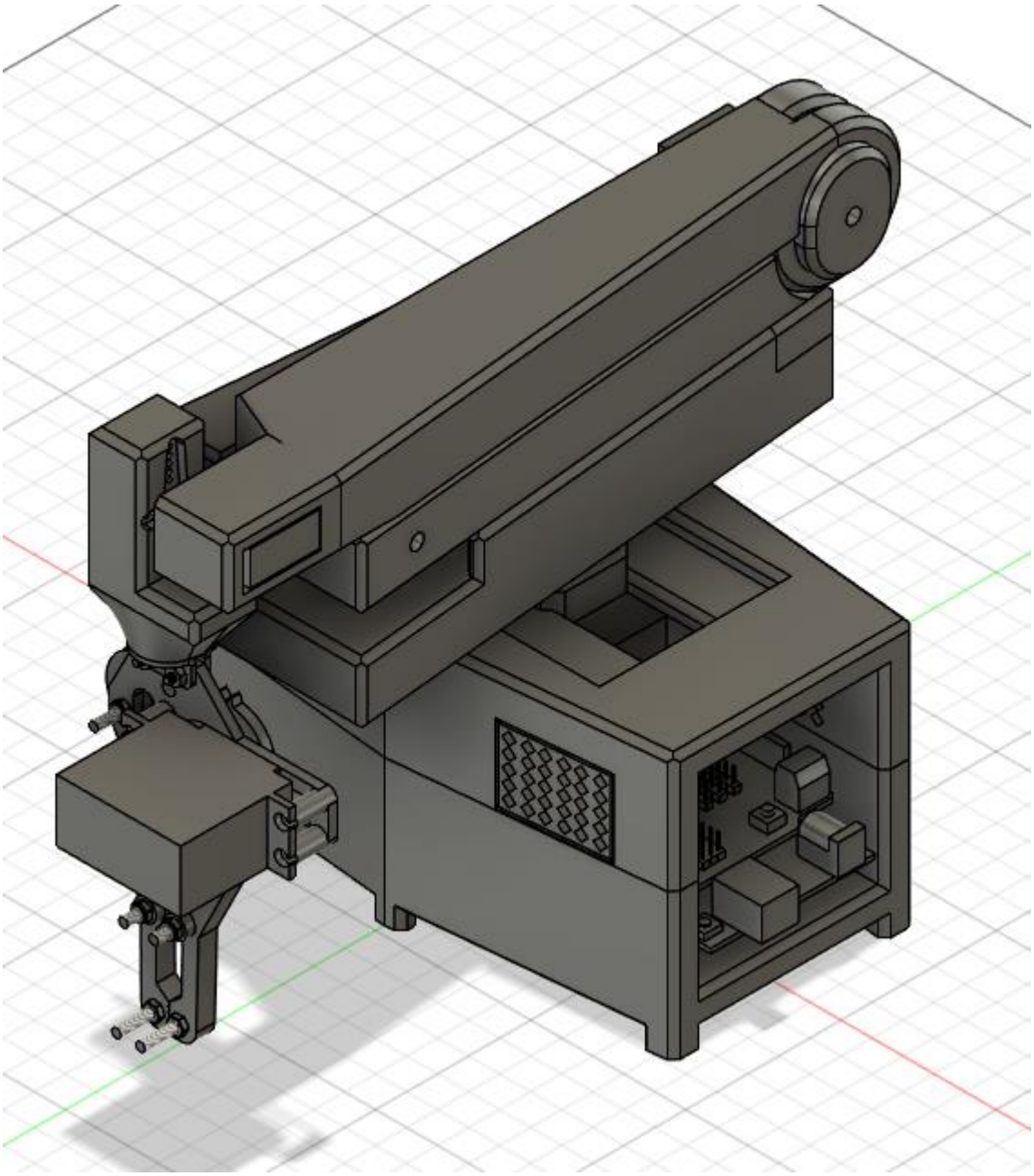

Рисунок 3.6 – Задіяння сервоприводу Base

На рисунку 3.7 зображено максимальну довжину уперед на яку може витягнутись «рука» робота – 343 мм, якщо враховувати довжину схвату (137 мм від точки обертання схвату до точки утримання вантажу), то 343 мм + 137 мм = 480 мм – максимальна відстань на якій може знаходитись вантаж у робочосу просторі для робота-маніпулятора.

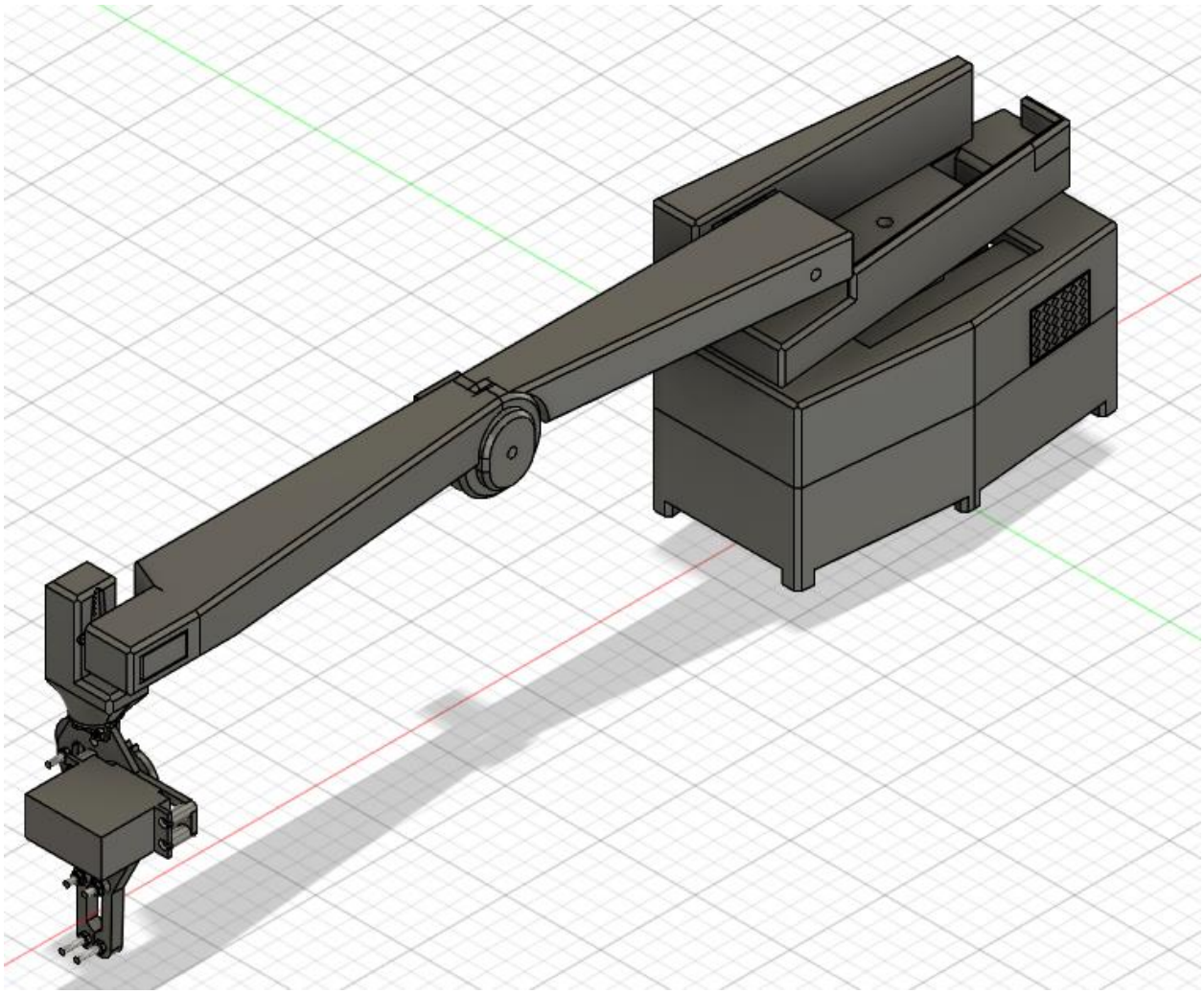

Рисунок 3.7 – Рука маніпулятора витягнута на максимальну довжину

На рисунку 3.8 зображено використання однієї з кінцівок робота, яка дає змогу витягнути схват на 215 мм назад, відносно самого робота. Від корпусу до точки у крайньому положенні для схвату 215 мм + 137 мм = 352 мм, відстань до якої може дотягнутися робот у положені назад та схватити вантаж.

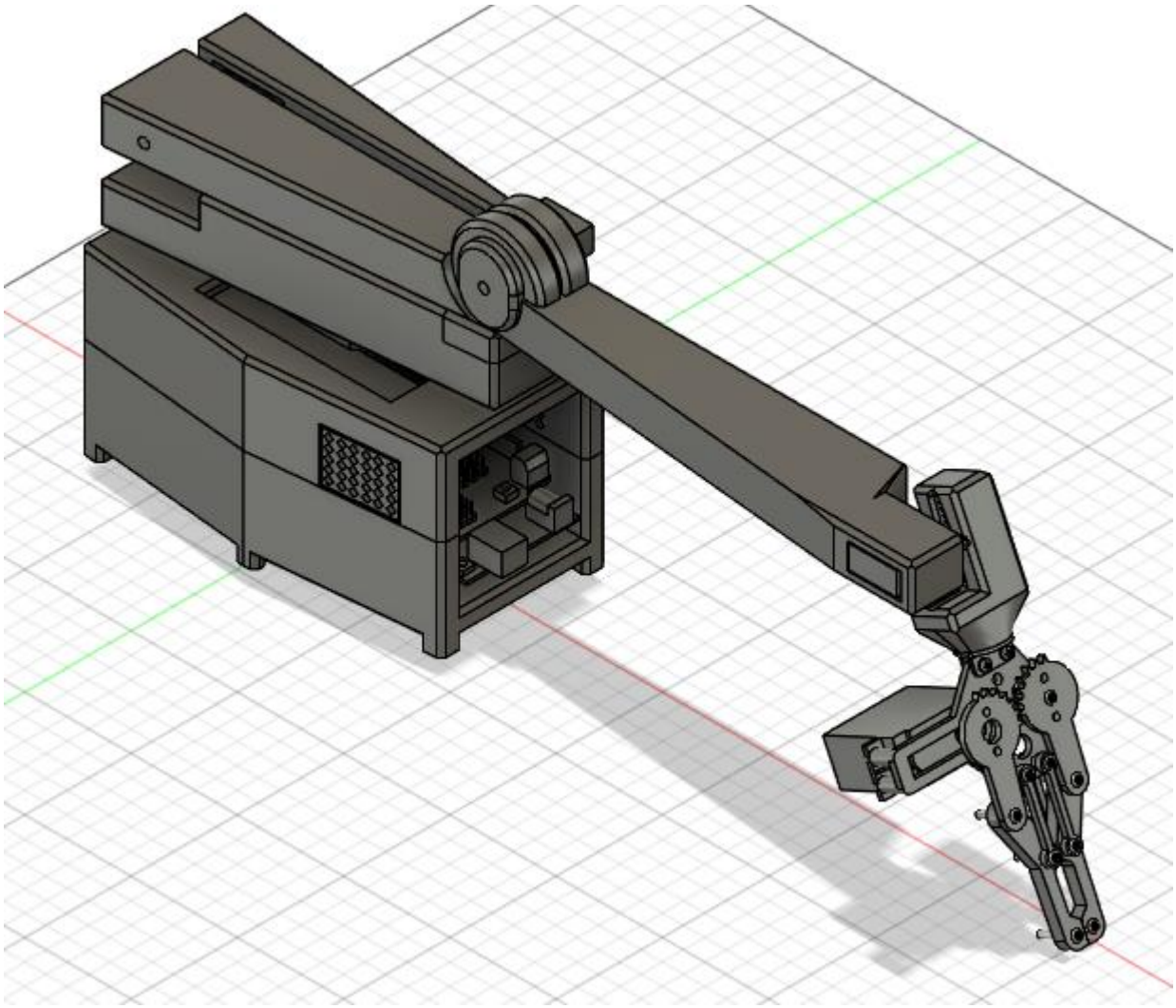

Рисунок 3.8 – Розміщення схвату у задній частині маніпулятора без розвороту

На рисунку 3.9 зображено на яку висоту може робот-маніпулятор підняти свій схват, довжина ліктьового суглоба 201 мм (від точки обертання цієї кістки до точки обертання ліктьового суглоба), 201 мм + 137 мм = 338 мм, «перша» висота на якій може оперувати робот.

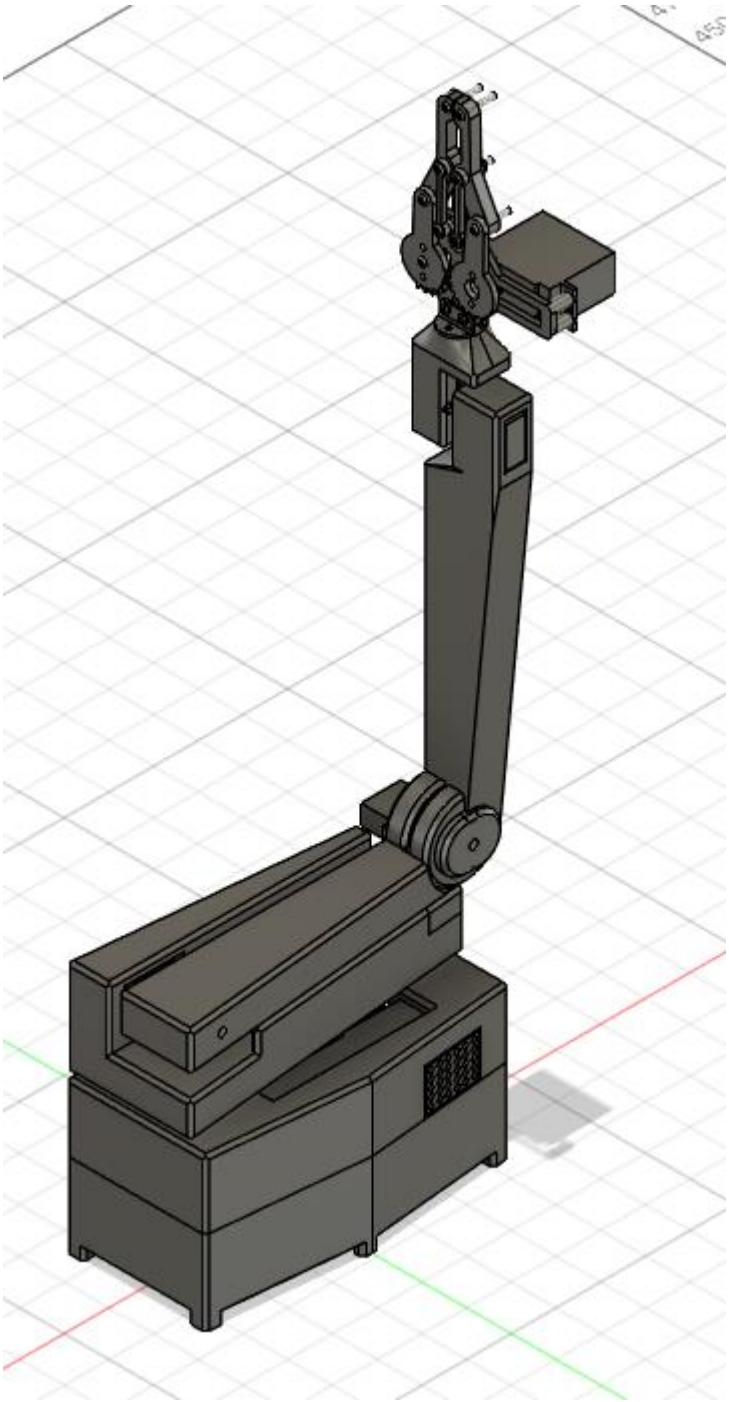

Рисунок 3.9 – Задіяння передпліччя

На рисунку 3.10 зображено «другу» максимальну висоту, яку робот здадній доясягти, 142 мм довжина плеча, 142 мм + 201 мм + 137 = 480 мм, максимальна висота точки для вантажу, яку може схватити маніпулятор.

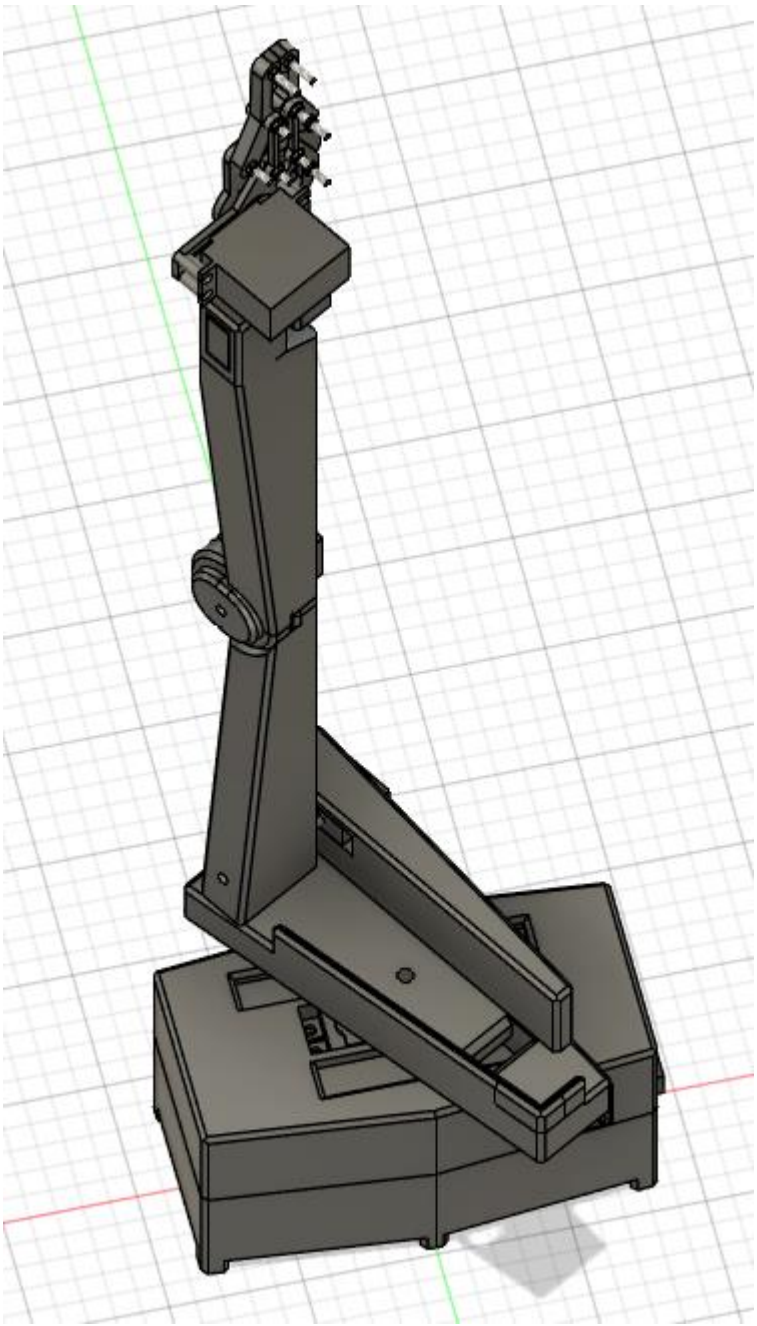

Рисунок 3.10 – Максимальна висота

На рисунку 3.11 зображено повністю задіяний робот-маніпулятор у якого робочій простір 360 градусів, максимальна висота 480 мм, максимальна довжина 480 мм, при задіянні тільки кисті, що обертає схват мінімальна точка робочого простору 37 мм – 65 мм, в залежності від позиції, але при задіянні всіх кінцівок ця мінімальна точка нівелюється.

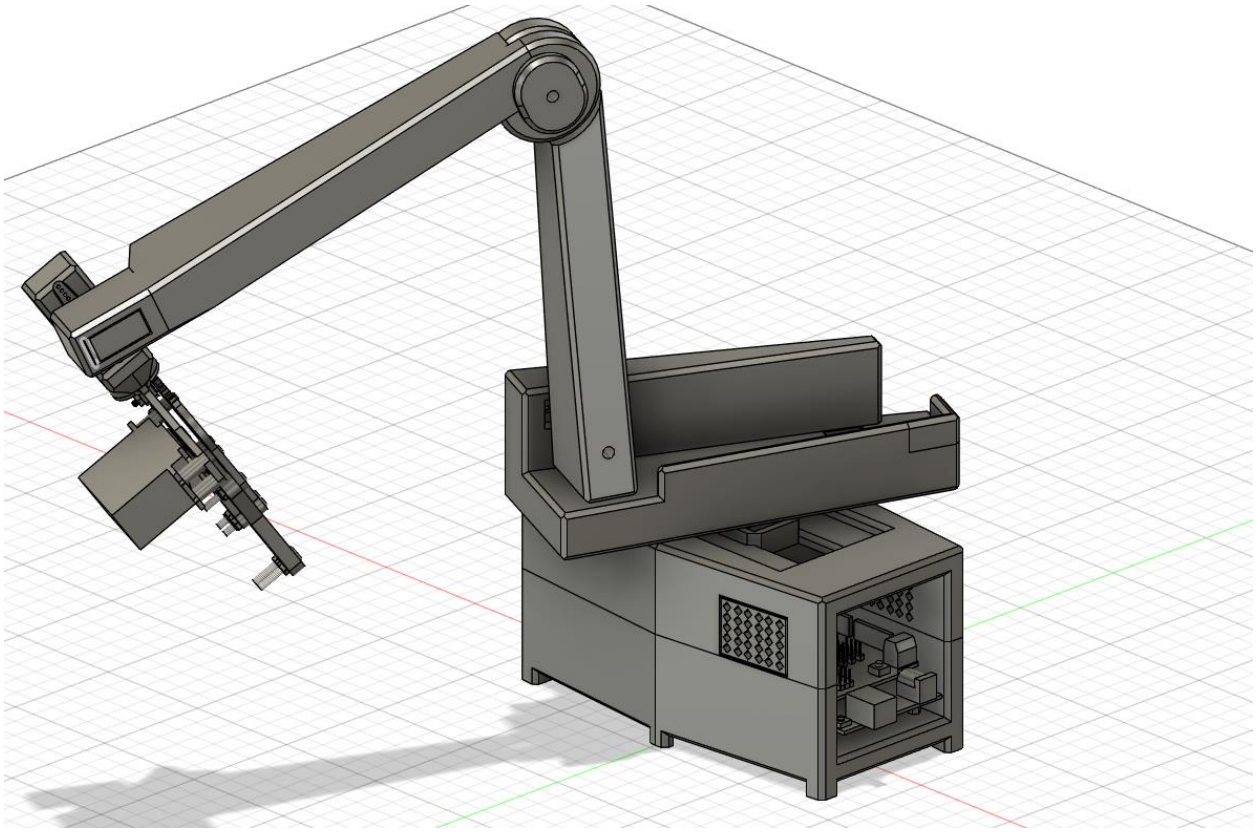

Рисунок 3.11 – Повний вигляд маніпулятора

На рисунку 3.12 зображено мінмалістичний корпус геймпаду, цього достатньо для монтування двух джойстиків, за допомогою яких буде відбуватися керування всім роботом-маніпулятором.

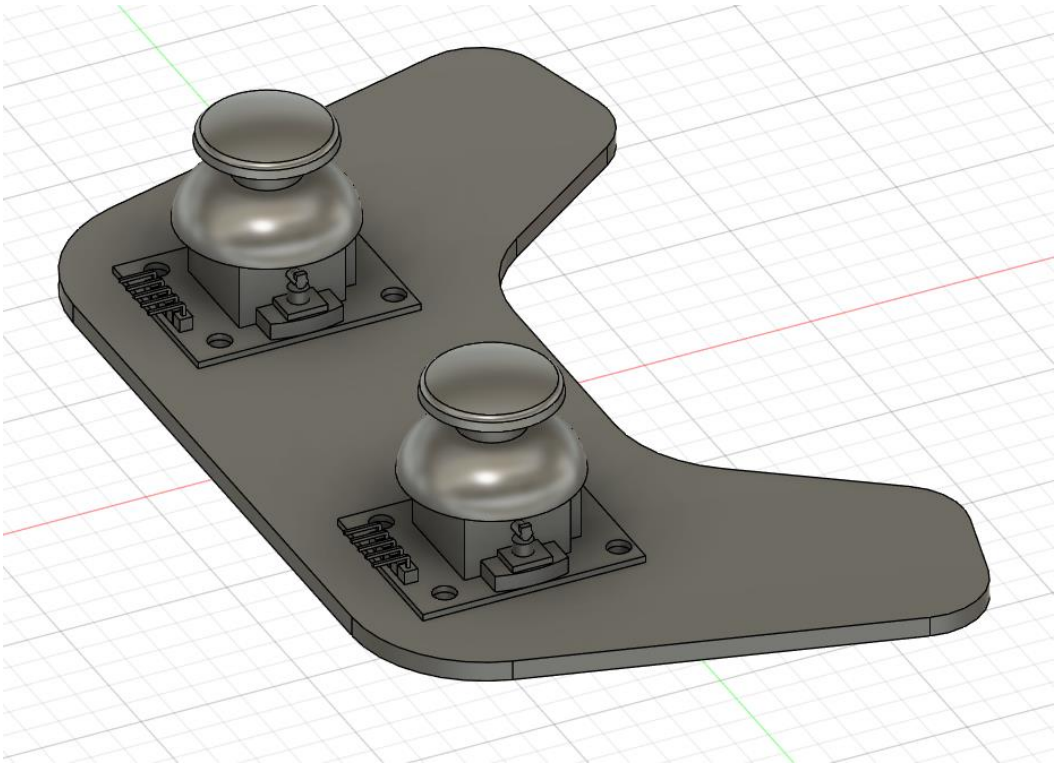

Рисунок 3.12 – «Геймпад» для керування маніпулятором

3.2 Креслення деталей робота-маніпулятора

На рисунку 3.13 зображено креслення головного корпусу, як можна побачити в ньому є спеціальний простір для апаратного забезпечення (для компактності), отвори вентиляції та канали для проводів від сервоприводів, що будуть надходити до плат.

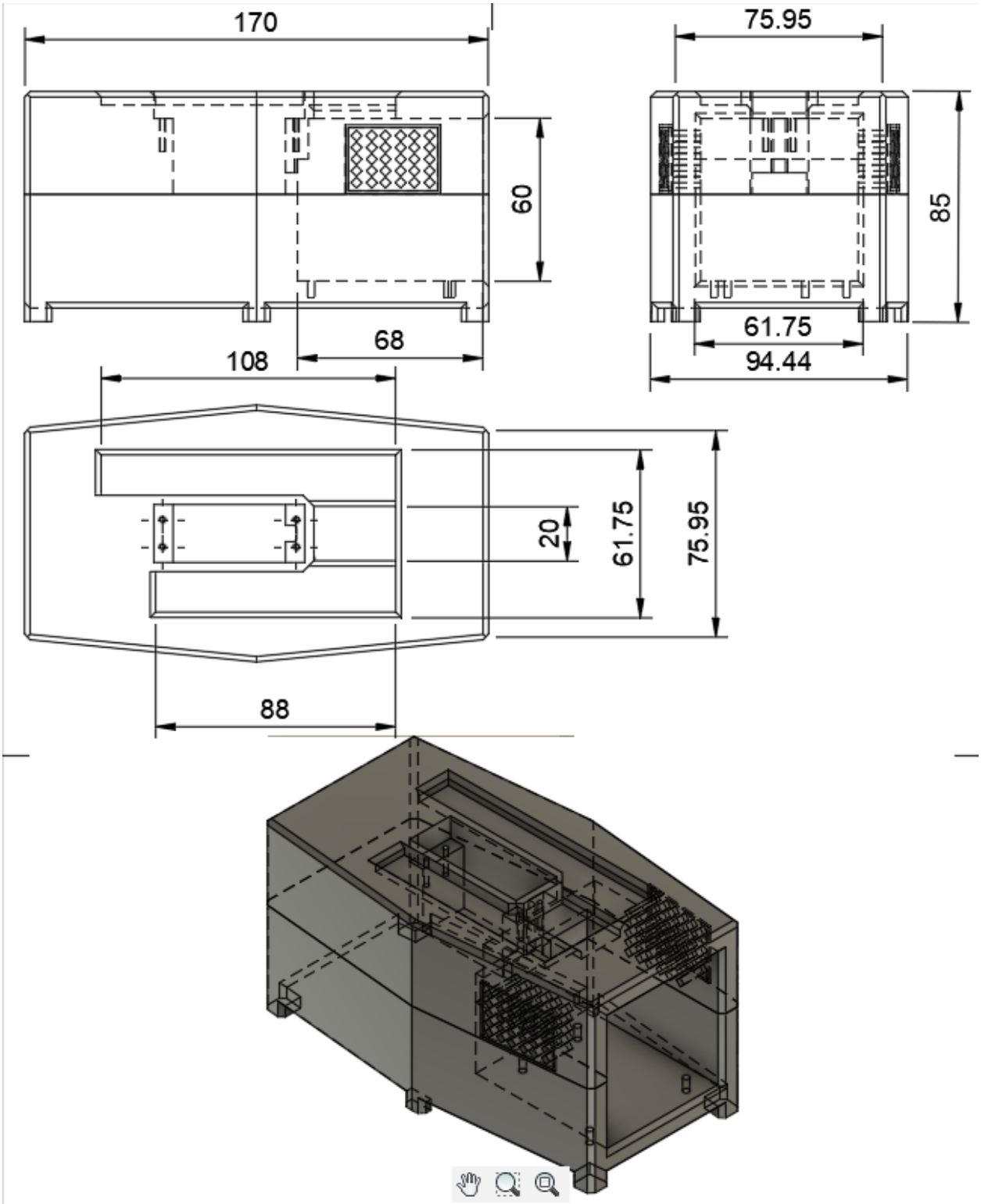

Рисунок 3.13 – Креслення головного корпусу

На рисунку 3.14 зображено плече що обертається на головному корпусі, з’єднане через монтажну втулку, яка надівається на вал сервопривода, а сама втулка кріпиться до корпусу плеча.

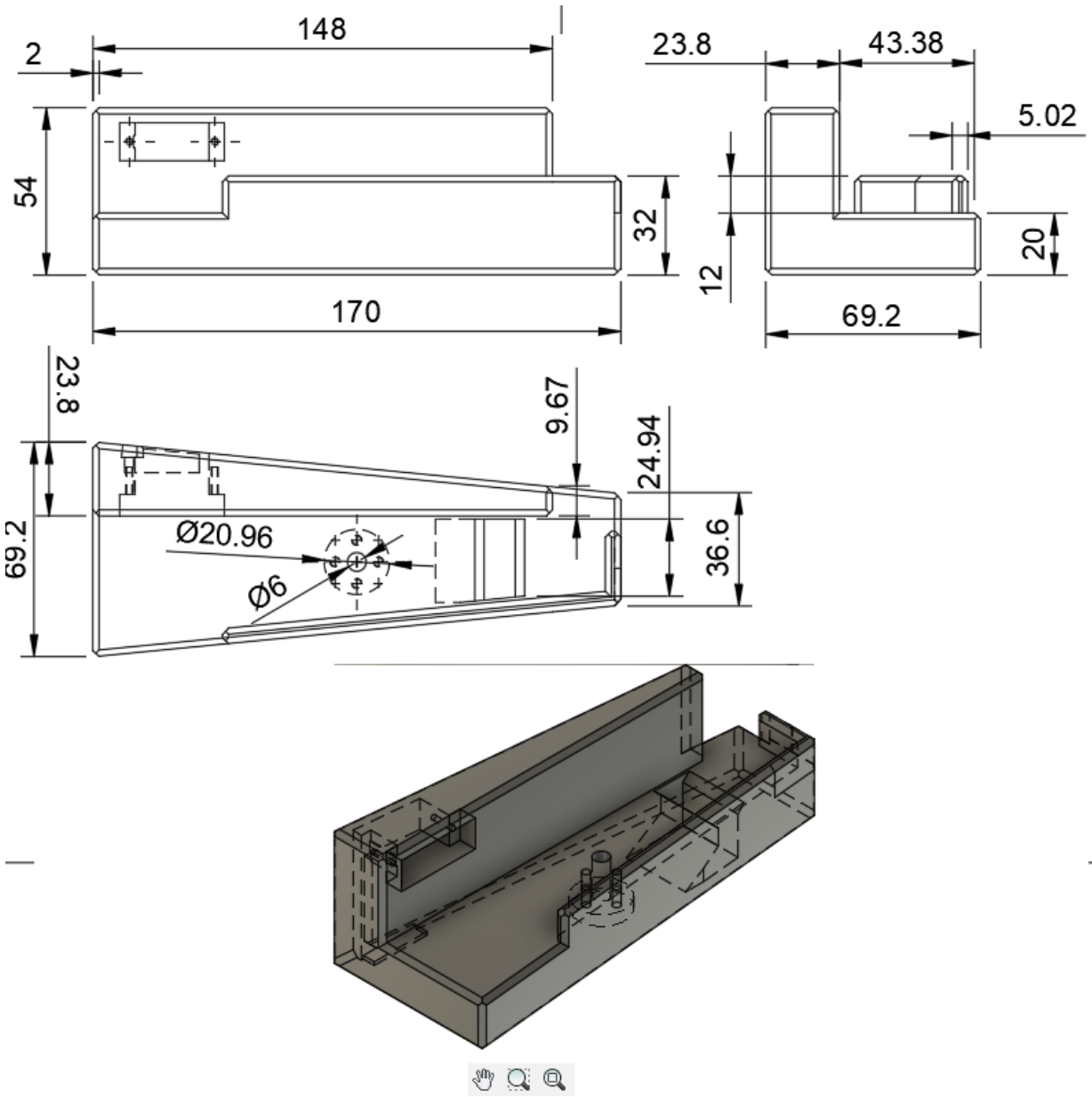

Рисунок 3.14 – Креслення плеча

На рисунку 3.15 зображено плече маніпулятора, яку вбудований сервопривід обертає по вертикалі через хрестоподібну монтажну втулку з отворами, кругла частина задіяна у обертанні ліктя через отвір для сервопривода.

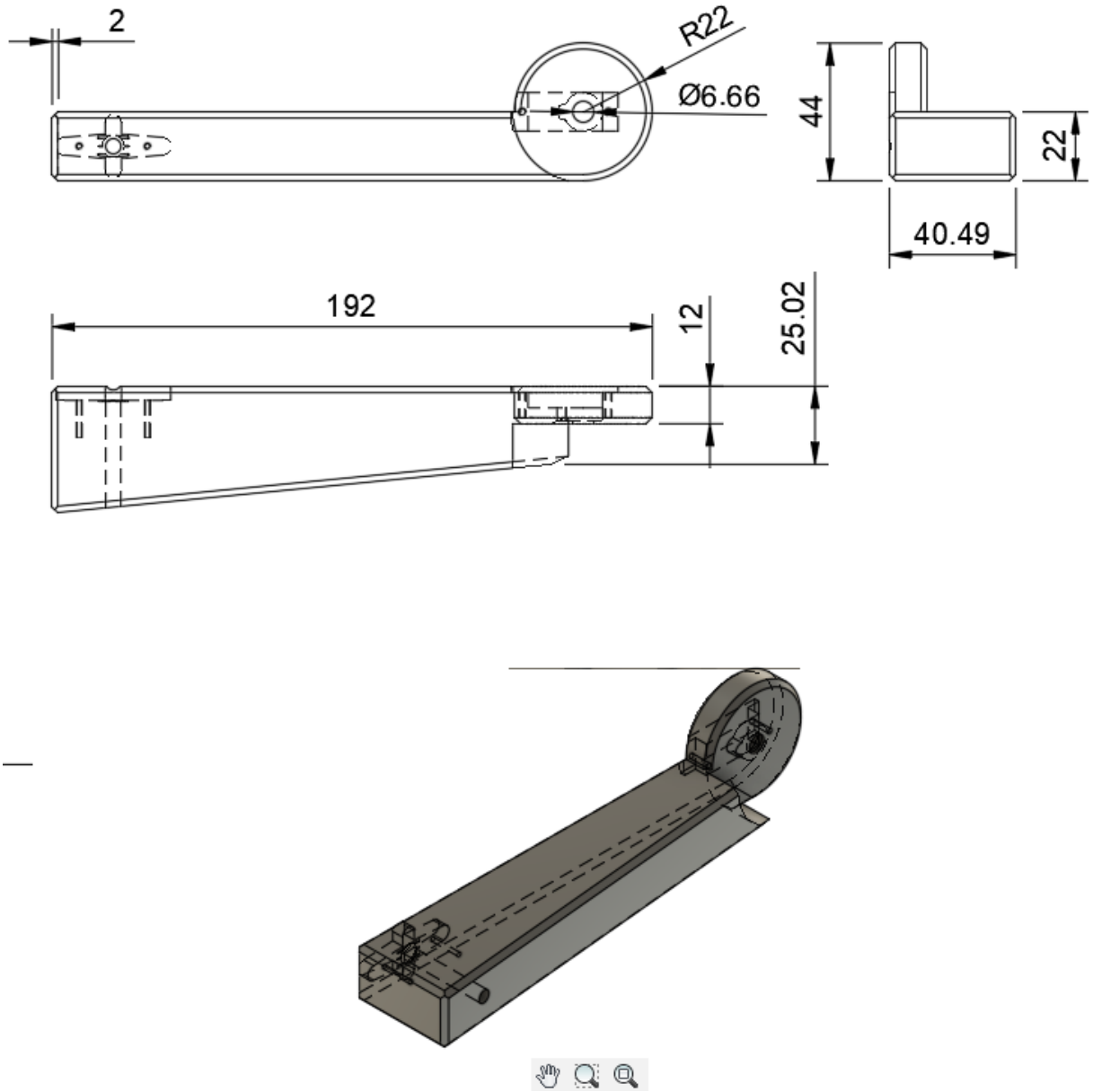

Рисунок 3.15 – Креслення плечового суглоба

На рисунку 3.16 зображено лікоть, який кріпиться через хрестоподібну монтажну втулку з отворами та обертається по вертикалі, має на іншому кінці отвір для іншого сервопривода, що обертатиме кисть.

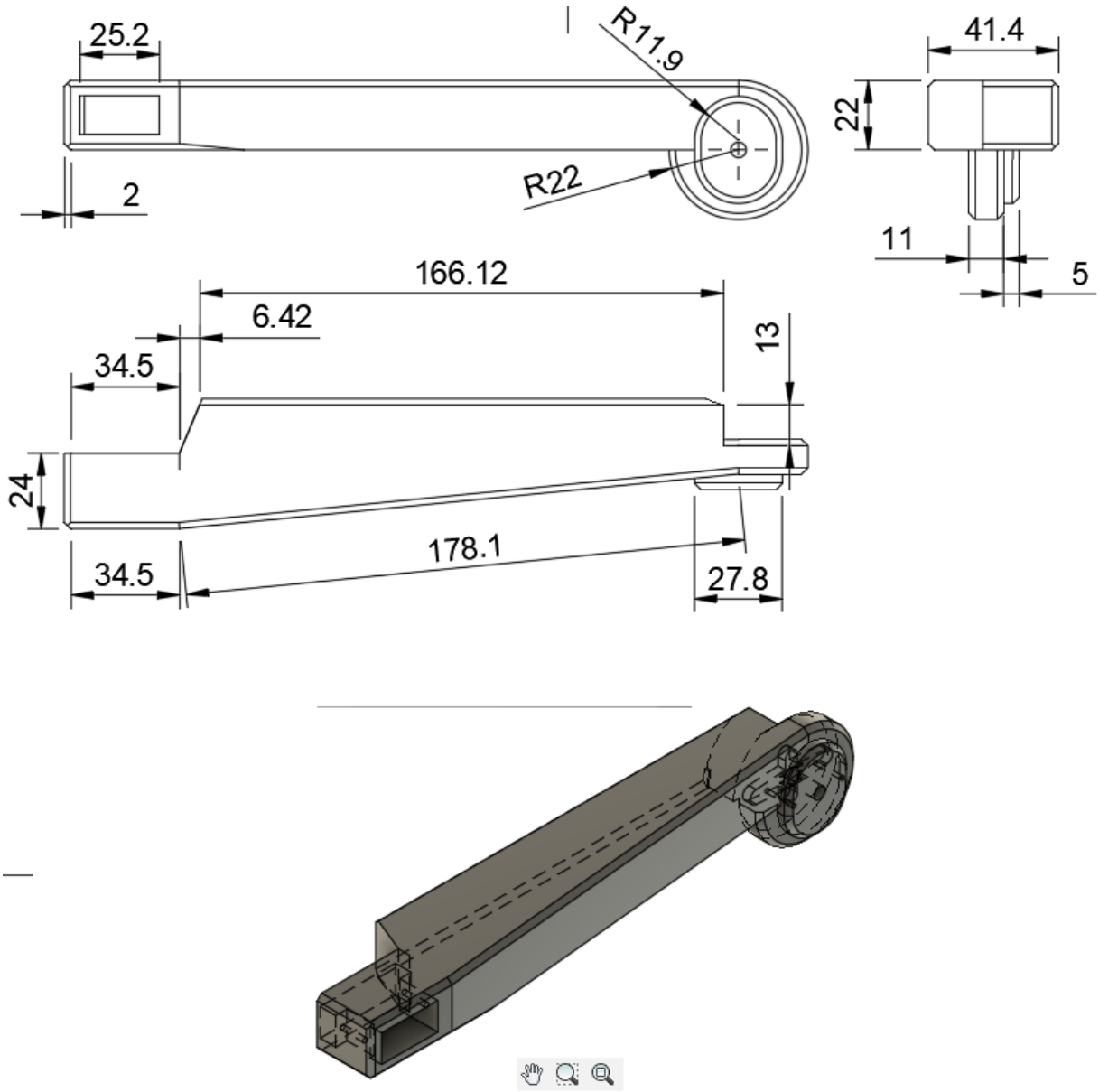

Рисунок 3.16 – Креслення ліктьової частини

На рисунку 3.17 зображено кисть руки маніпулятора, яка обертається вбудованим сервоприводом у ліктя через таку ж саму втулку, кисть має два отвори під болти у еліпсоподібній частині, що кріплять схват до неї.

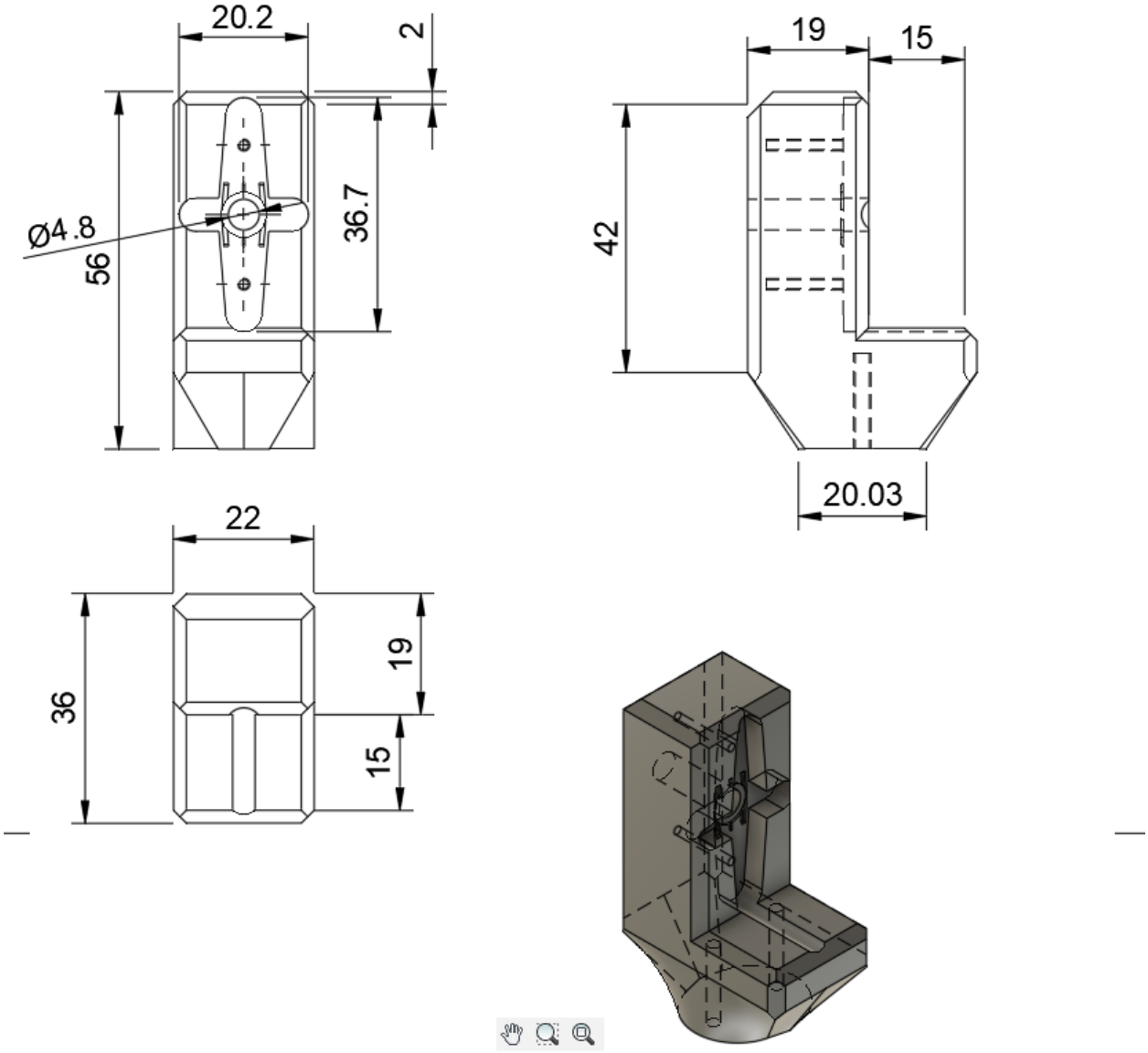

Рисунок 3.17 – Креслення кисті

На рисунку 3.18 зображено креслення спроєктованого корпусу геймпаду з отворами під джойстики.

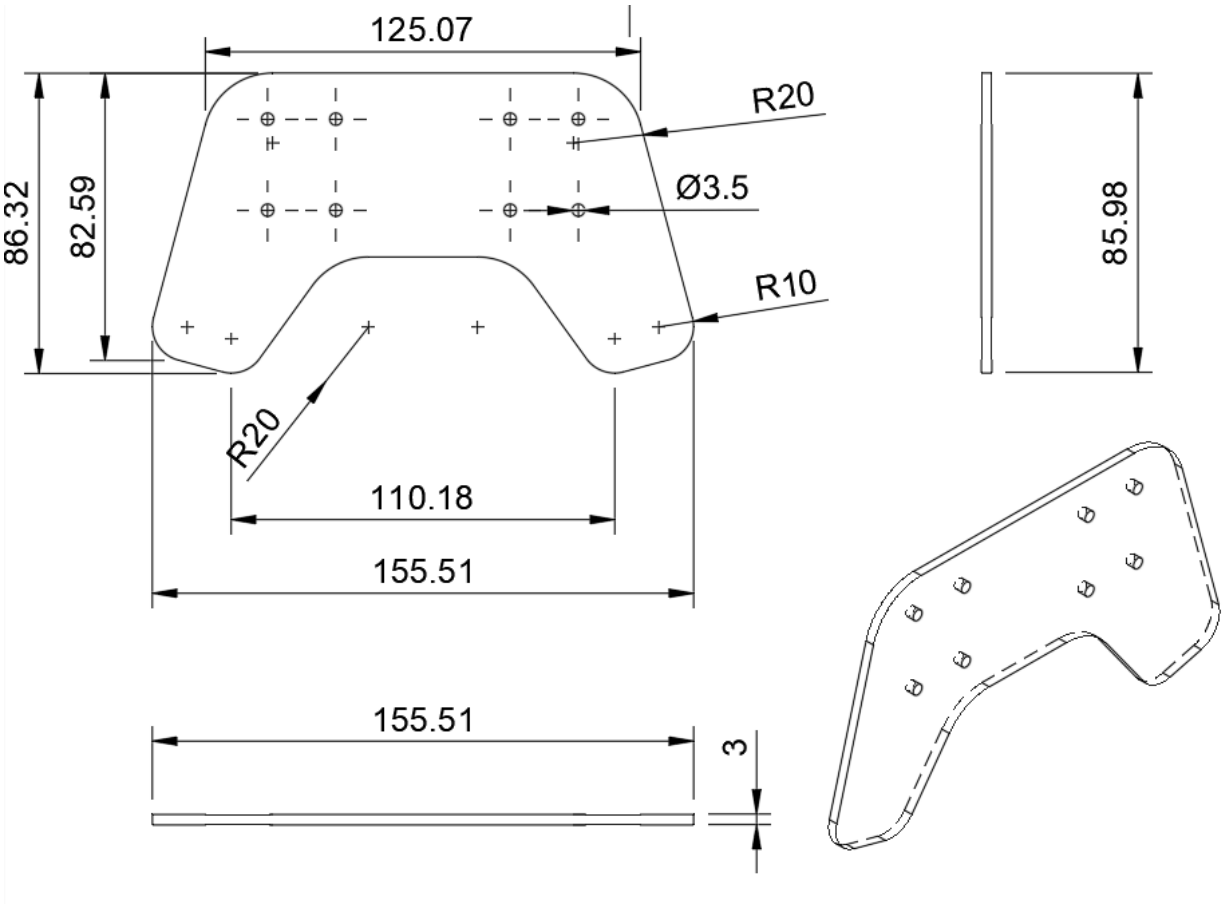

Рисунок 3.18 – Креслення «геймпаду»

На рисунку 3.19 зображено креслення схвату, яке працює за допомогою сервопривода, що кріпиться на 4 болти у спеціальному місці для підведення валу на який надівається втулка, кріпиться до важіля, що рухає через зубці інший важіль, ці два важіля у свою чергу рухають пальці, якими можна утримувати вантаж при стисканні цими пальцями.

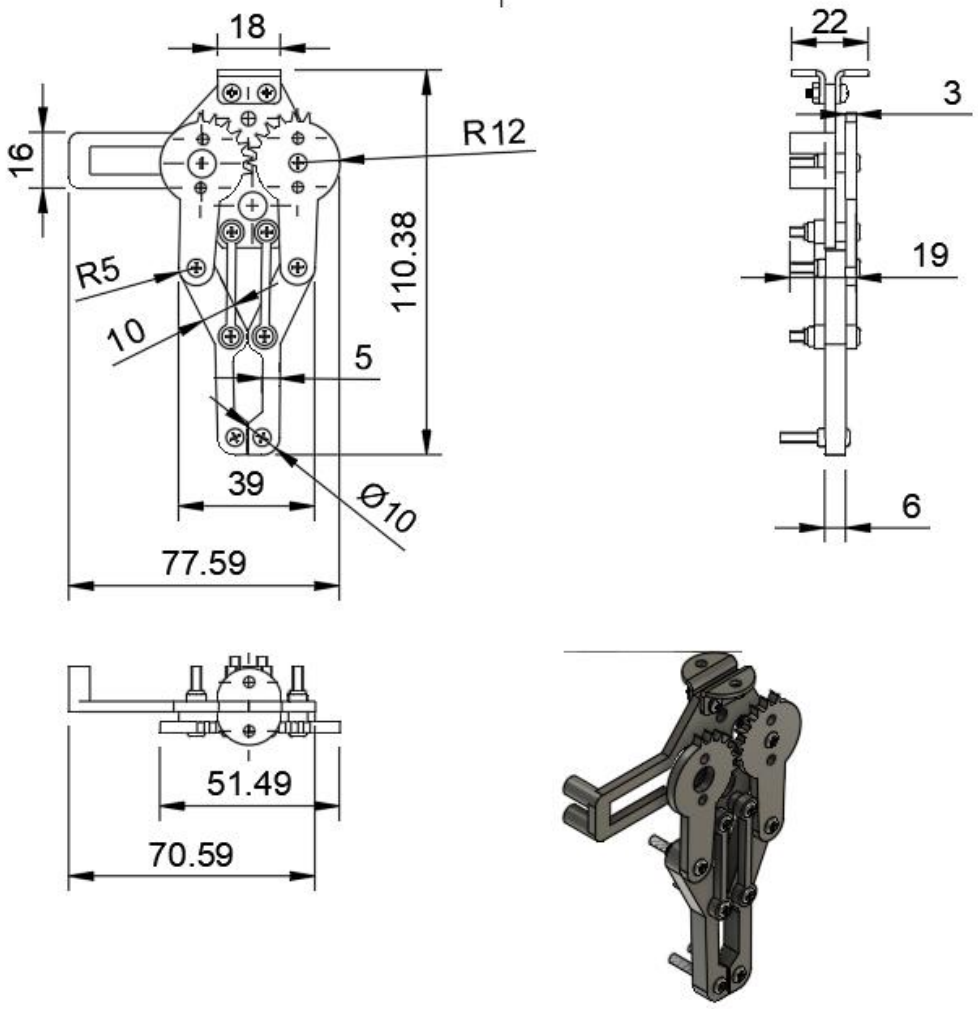

Рисунок 3.19 – Креслення схвату

4 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ МАНІПУЛЯТОРА І КІНЕМАТИКИ

4.1 Середовище розробки Arduino IDE

Arduino IDE – інтегроване середовище розробки для Windows, macOS та Linux, розроблене на C та C++, призначене для створення та завантаження програм на Arduino-сумісні плати, а також на плати інших виробників (рис. 4.1).

Рисунок 4.1 – Arduino IDE

Вихідний код для середовища випущений під загальнодоступною ліцензією версії GNU 2. Підтримує мови C і C++ з використанням спеціальних правил структурування коду. Arduino IDE надає бібліотеку програмного забезпечення з проекту Wiring, яка надає безліч загальних процедур введення та виведення. Для написаного користувачем коду потрібні лише дві базові функції для запуску ескізу та основного циклу програми, які скомпільовані та пов'язані із заглушкою програми main () у виконувану циклічну програму з ланцюжком інструментів GNU, також включеною до дистрибутиву IDE. Використовує програму avrdude для перетворення коду, що виконується, в текстовий файл у

шістнадцятковому кодуванні, який завантажується в плату Arduino програмою-завантажувачем у вбудованому програмному забезпеченні плати.

Розробка коду керування маніпулятором:

```
void setup() {
Serial.begin(9600);
base.attach(pin_base);
shoulder.attach(pin_shoulder);
elbow.attach(pin_elbow);
rotator.attach(pin_rotator);
gripper.attach(pin_gripper);

pinMode(btn_key1, INPUT_PULLUP);
pinMode(btn_key2, INPUT_PULLUP);

loadFromEEPROM();
applyServoAngles();
```

У void setup() ініціалізується серійний порт на швидкості 9600 бод – потрібно для обміну даними з ПК (Visual Studio або монітор порту).

Також підключаються кожний сервопривід до відповідного цифрового піна. Метод attach() встановлює зв’язок між об’єктом Servo і піном на Arduino.

Кнопки джойстиків налаштовані як вхід з підтягуючим резистором – тобто, натискання = LOW (0).

На старті завантажуються останні збережені кути з EEPROM, і ці кути одразу застосовуються до сервоприводів – маніпулятор «відновлює» свою позицію.

```
void loop() {
// === Зчитування значень ===
int x1 = analogRead(x_key1);
int y1 = analogRead(y_key1);
int x2 = analogRead(x_key2);
int y2 = analogRead(y_key2);
bool btn1 = digitalRead(btn_key1) == LOW;
bool btn2 = digitalRead(btn_key2) == LOW;
// === Переключення режиму (антидребезг) ===
if (btn1 != lastButton1State) {
lastDebounceTime = millis();
  }
```

```
        if ((millis() - lastDebounceTime) > debounceDelay) {
        if (btn1 != mode) {
        mode = !mode;
        EEPROM.update(ADDR_MODE, mode);
        Serial.print("Mode switched to: ");
        Serial.println(mode == 0 ? "Manual" : "Coordinate");
        delay(500);
          }
       }
       lastButton1State = btn1;
        // === Режим 0: Ручне керування ===
        if (mode == 0 && millis() - lastMoveTime > moveDelay) {
          // === Керування 360° сервоприводом base ===
          if (x1 < threshold) {
          base.write(0);   // обертання вліво
          } else if (x1 > 1023 - threshold) {
            base.write(180); // обертання вправо
          } else {
            base.write(90);  // стоп
          }
      // === Інші серво по кроку ===
      if (y1 < threshold) shoulderAngle = constrain(shoulderAngle - step, 0,
180);
      if (y1 > 1023 - threshold) shoulderAngle = constrain(shoulderAngle + step,
0, 180);

      if (x2 < threshold) elbowAngle = constrain(elbowAngle - step, 0, 180);
      if (x2 > 1023 - threshold) elbowAngle = constrain(elbowAngle + step, 0,
180);

      if (y2 < threshold) rotatorAngle = constrain(rotatorAngle - step, 0, 180);
      if (y2 > 1023 - threshold) rotatorAngle = constrain(rotatorAngle + step,
0, 180);
      if (btn2) gripperAngle = (gripperAngle == 0) ? 45 : 0;

      applyServoAngles();
      saveToEEPROM();
      lastMoveTime = millis();
      }
```

У цій частині кода зчитуються положення X/Y обох джойстиків, а також чи натиснута кнопка – використовується для керування або перемикання режимів.

Здійснюється перевірка на зміну стану кнопки з урахуванням затримки (антидребезг контактів), і якщо кнопка натиснута, відбувається перемикання між ручним і координатним режимом. Новий режим записується у пам’ять.

Управління основою маніпулятора (360° серво):

- якщо джойстик відхилений вліво – обертає вліво (0);
- вправо – обертає вправо (180);
- не рухаємо – 90 (нейтральна позиція).

Керування іншими сервомоторами відбувається покроково – при кожному циклі кут змінюється на певне значення step, обмежене діапазоном 0…180.

Кожного разу, коли відбувається зміна кута – вона зберігається та застосовується до сервоприводів. Таким чином, стан оновлюється і фіксується.

```
// Прийом команд с ПК
if (Serial.available()) {
String input = Serial.readStringUntil('\n');
input.trim();
if (input.startsWith("SET")) {
int s, e, r, g;
int n = sscanf(input.c_str(), "SET S:%d E:%d R:%d G:%d", &s, &e, &r, &g);
if (n == 4) {
        shoulderAngle = constrain(s, 0, 180);
        elbowAngle    = constrain(e, 0, 180);
        rotatorAngle  = constrain(r, 0, 180);
        gripperAngle  = constrain(g, 0, 180);
        applyServoAngles();
        saveToEEPROM();
      }
    }
  }

  sendAnglesOverSerial();
  delay(100);  // затримка для стабільної передачі

  sendAnglesOverSerial();
}

// === Встановлення кутів ===
void applyServoAngles() {
  // base — 360° сервопривід, керується окремо
  shoulder.write(shoulderAngle);
```

```
        elbow.write(elbowAngle);
        rotator.write(rotatorAngle);
        gripper.write(gripperAngle);
      }

      // === Відправка кутів в Serial ===
      void sendAnglesOverSerial() {
        Serial.print("S:"); Serial.print(shoulderAngle);
        Serial.print(" E:"); Serial.print(elbowAngle);
        Serial.print(" R:"); Serial.print(rotatorAngle);
        Serial.print(" G:"); Serial.print(gripperAngle);
        Serial.print(" | Mode: "); Serial.println(mode == 0 ? "Manual" :
"Coordinate");
      }
```

У цій частині кода перевіряється, чи прийшло щось по COM-порту. Якщо так – читає рядок до символу \n.

Також розбирається команда типу SET S:90 E:45 R:30 G:0 – задає кути плеча, ліктя, ротатора та клешні.

Далі воно надсилає поточні кути та режим у вигляді текстового рядка в COM-порт. Цей рядок читається у Visual Studio-програмі для візуалізації положення маніпулятора.

Функція void sendAnglesOverSerial() одночасно виставляє всім серво їх поточні кути та формує рядок такого типу:

S:90 E:45 R:30 G:0 | Mode: Manual

4.2 Розробка програмного забезпечення для реалізації обрахунків кінематики маніпулятора у Visual Studio (2022)

Microsoft Visual Studio – це лінійка систем розробки програмного забезпечення від компанії Microsoft (рис. 4.2, 4.3). У своєму складі мають інтегроване середовище розробки (IDE) та низку інших інструментів.

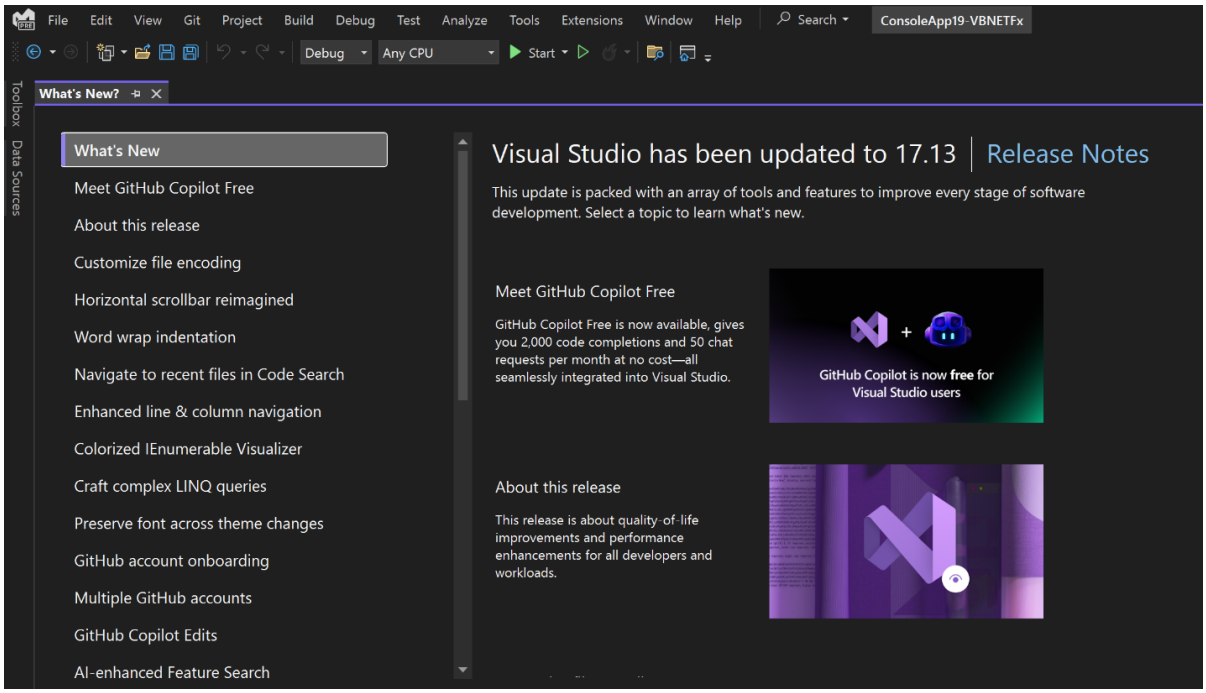

Рисунок 4.2 – Microsoft Visual Studio

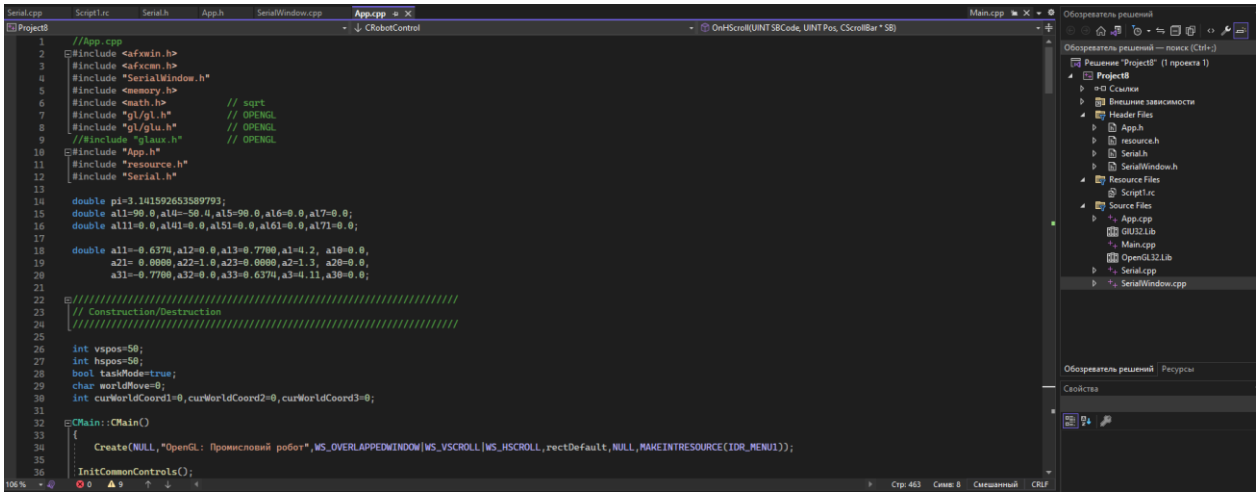

Рисунок 4.3 – Головний робочій простір

Microsoft Visual Studio дозволяє розробляти як консольні програми, так і ігри та програми з графічним інтерфейсом (GUI). Зокрема, підтримуються технології:

– Windows Forms;

– UWP;

– веб-сайти, веб-застосунки, веб-служби як у «рідних» кодах, так і в керованому коді для всіх платформ, що підтримуються операційними

системами сімейства Windows (у тому числі Windows Mobile та Windows CE, Windows Phone).

Visual Studio включає редактор вихідного коду, що підтримує технологію IntelliSense і можливістю найпростішого рефакторингу коду. Вбудований налагоджувач Microsoft Visual Studio Debugger може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Інший вбудований інструментарій включає редактор форм для спрощення створення користувальницького графічного інтерфейсу, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio також дозволяє створювати та підключати сторонні доповнення-плагіни, що розширюють функціонал практично на кожному рівні.

Код визову вікон:

```
BOOL CApp::InitInstance()
{
	m_pMainWnd = new CMain;
	m_pMainWnd->ShowWindow(m_nCmdShow);
	m_pMainWnd->UpdateWindow();
// Запускає два вікна
	((CMain*)m_pMainWnd)->OnRobotControl();   // Діалог MFC
	((CMain*)m_pMainWnd)->OnSerialWindow();   // WinAPI вікно
	return TRUE;
}
```

Функція InitInstance() забезпечує запуск програми (CMain) – саме в ньому буде рендеритись 3D-сцена OpenGL.

Також в InitInstance() відбувається запуск двох додаткових вікон:

OnRobotControl() – відкриває MFC-діалог для керування маніпулятором (слайдери, кнопки, Edit-поля);

OnSerialWindow() – WinAPI-вікно для зчитування та моніторингу COM-порту.

Всі три частини – 3D-сцена, панель керування та серійне з’єднання – активуються при запуску програми.

Код розрахунку матриці:

```
void CRobotControl::JointsGen(int joint)
{	double L2=115.0,L3=110.0,L4=83.0;
	for(double a2=-45.0;a2<90.0;a2=a2+0.25)
```

```
            for(double a3=-90.0;a3<90.0;a3=a3+0.25)

{double a,b,q2,q3,e;
 a=cos(a2*pi/180.0)*L2-sin(fi23)*(L3+L4);
 b=sin(a2*pi/180.0)*L2+cos(fi23)*(L3+L4);
q2=atan((b*L2*(1-sin(a3*pi/180.0))-a*cos(a3*pi/180.0)*(L3+L4))/(a*L2*(1-sin(a3*pi/180.0))-
b*cos(a3*pi/180.0)*(L3+L4)));
 q3=asin((L2*L2+(L3+L4)*(L3+L4)-a*a-b*b)/(2*L2*(L3+L4)));
e=q2+q3;rColumn2=180.0*fi1/pi;rShoulder2=180.0*q2/pi;rElbow2=180.0*q3/pi;
if(fabs(fi23-e)<0.1){break;}

            }
            CString ss;
            ss.Format(_T("%f %f %f"),rColumn2,rShoulder2,rElbow2);
            SetWindowText(ss);
}
```

Також задаються довжини сегментів та перебір кутів суглобів a2 та a3 для знаходження комбінації, яка відповідає заданій конфігурації.

Обчислюється положення кінцевої точки (X, Y) маніпулятора в просторі, обчислюються кути суглобів зворотною кінематикою – знаходяться значення, які відповідають цільовому положенню руки.

Результати (кути) виводяться у вікно діалогу – наприклад, у поле Edit.

Створення класу з буфером для SerialPort:

```
class Serial {
private:
   HANDLE hSerial;
public:
   Serial(const char* portName);
   ~Serial();
   int read(char* buffer, unsigned int buf_size);
};
```

Клас Serial{} інкапсулює роботу з COM-портом (через WinAPI). Він відкриває порт (CreateFile), читає дані (ReadFile) та закриває його в деструкторі. Це дозволяє взаємодіяти з Arduino (читати, наприклад, кути положення серво або надсилати команди).

Оголошення діалогового вікна:

```
class CRobotControl : public CDialog
{

CSliderCtrl *sl1, *sl2, *sl3, *sl4, *sl5;
CButton *rb1,*rb2,*ch1;
CEdit *ed[19];
```

```
CSpinButtonCtrl spin1,spin2,spin3;
        public:
CRobotControl();
void JointsGen(int);
void OnOK();
void OnHandShake();
void OnRadio();
void OnSetMode(bool);
void OnHScroll(UINT SBCode,UINT Pos,CScrollBar *SB);
void OnVScroll(UINT SBCode,UINT Pos,CScrollBar *SB);
BOOL OnInitDialog();

CRobotControl(UINT id,CWnd *Owner):CDialog(id,Owner){}
//CRobotControl():CDialog(){}
DECLARE_MESSAGE_MAP()
//protected:
//        virtual void DoDataExchange(CDataExchange* pDX);
};
```

Оголошено основні елементи інтерфейсу: слайдери для кожного ступеня свободи, набір текстових полів для введення чи виводу.

Метод OnHScroll(UINT SBCode,UINT Pos,CScrollBar *SB) обробляє рух слайдерів. Коли користувач змінює значення, кути оновлюються й передаються на Arduino.

Метод OnInitDialog() ініціалізації вікна – створює слайдери, задає початкові значення, підключає поля введення, кнопки тощо.

Нижче наводиться код функції OnPaint(), в якому оголошується об'єкт пристрою виводу для малювання у вікні.

```
void CMain::OnPaint()
{
        CPaintDC dc(this);

        CPaintDC pDC(this);
   CPalette* ppalOld = NULL;
        if (m_pPal)
        {       ppalOld = pDC.SelectPalette(m_pPal, 0);
        pDC.RealizePalette();
        }
   wglMakeCurrent(pDC.m_hDC, m_hrc);
        GLInit();
        OnScene();
        SwapBuffers(pDC.m_hDC) ;
   if (ppalOld) pDC.SelectPalette(ppalOld, 0);
wglMakeCurrent(NULL, NULL) ;
}
```

wglMakeCurrent(...) – активує OpenGL-контекст.

GLInit() – ініціалізує параметри сцени (освітлення, проєкція, глибина).

OnScene() – головна функція рендерингу (малює маніпулятор).

SwapBuffers(...) – оновлює екран (подвійна буферизація).

Функція OnPaint() вимикає контекст після завершення рендерингу, щоб інші частини програми могли без конфліктів взаємодіяти з GUI.

На рисунку 4.4 зображено вікно зі сценою, на якій зображено за допомогою бібліотеки OpenGl проволочну версію реального маніпулятора, присутні повзунки для керування положенням сцени.

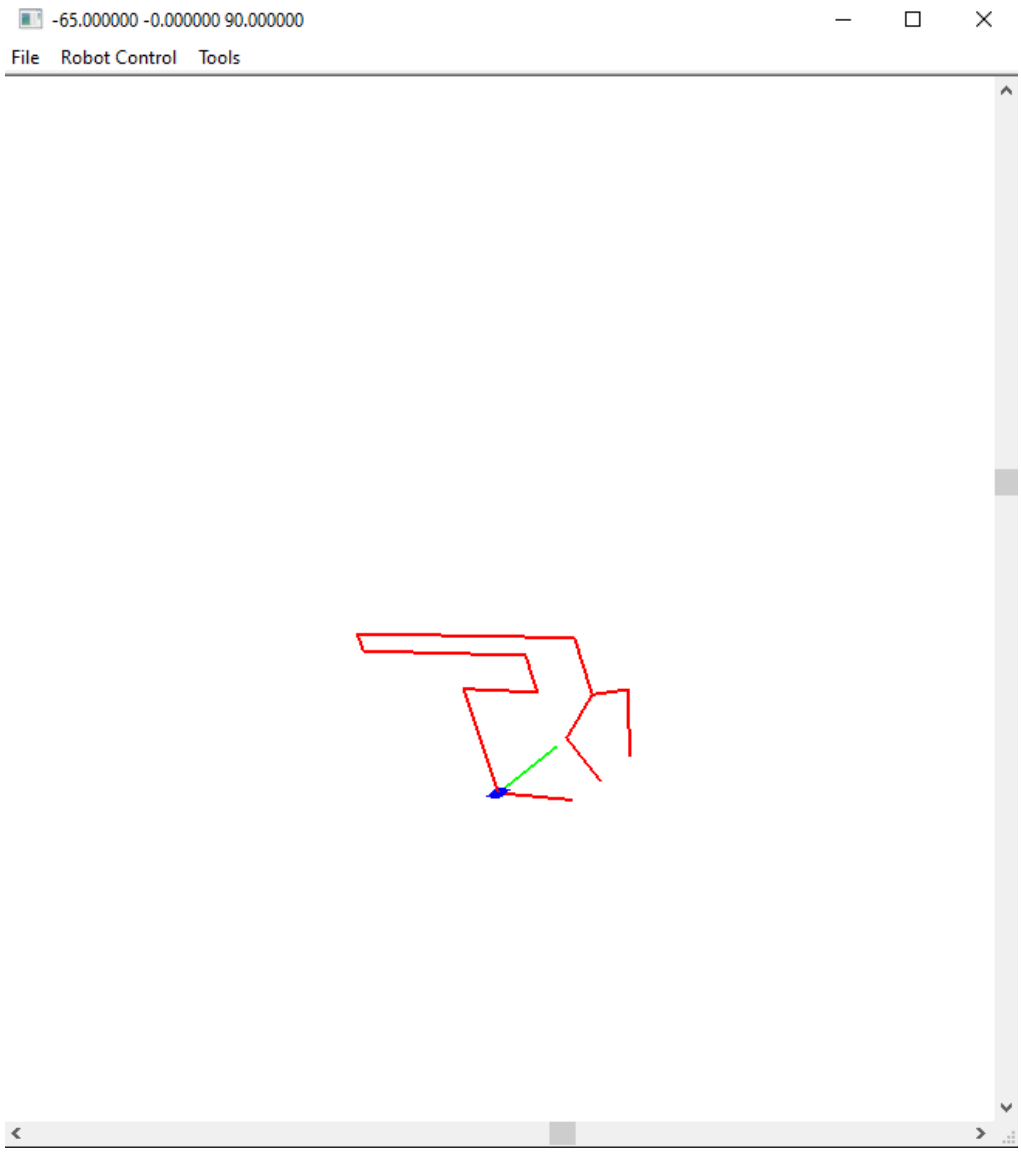

Рисунок 4.4 – Вікно з маніпулятором з OpenGL

На рисунку 4.5 зображено 2 вікна: діалогове, вікно зі сценою, через діаогове вікно відбувається керування проволочним маніпулятором, є 2 режими: звичайний та координатний, координати положення кінцівок виводяться у «едітбокси» – маленькі віконця.

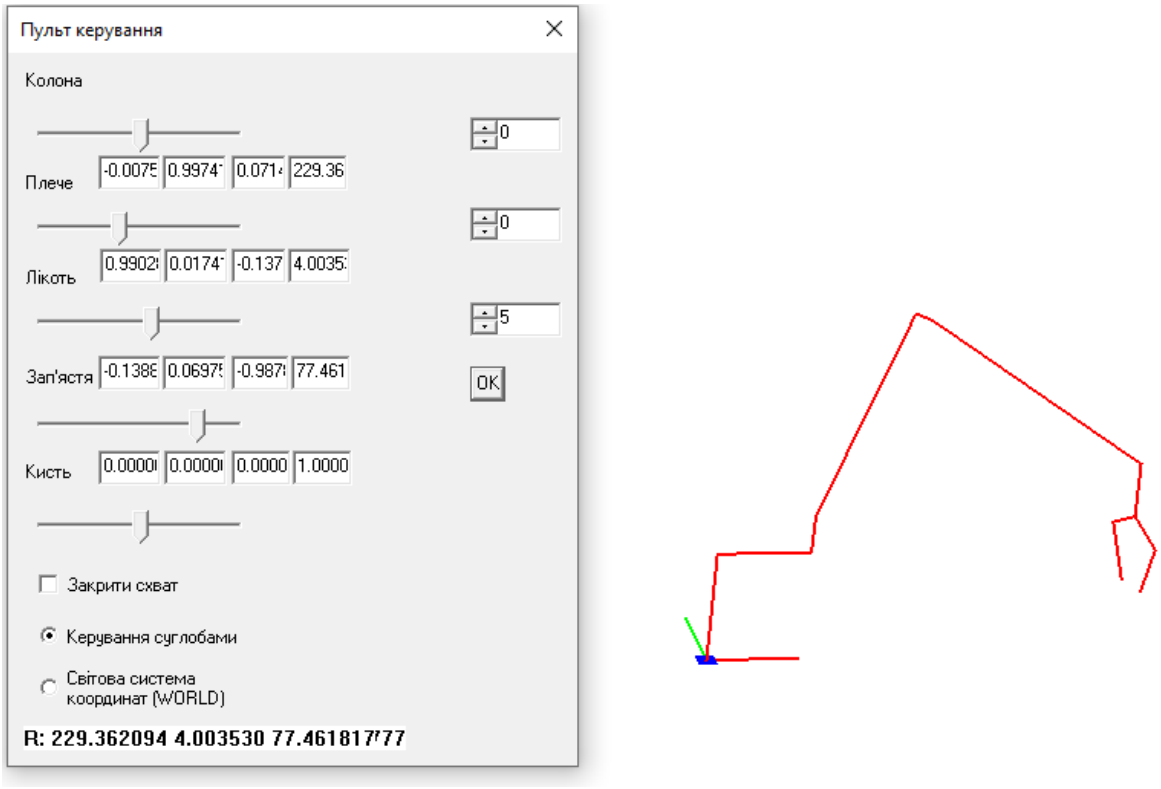

Рисунок 4.5 – Кінематична модель у Visual Studio

На рисунку 4.6 зображено діалогове вікно, через яке відбувається керування проволочним маніпулятором, побудованим у OpenGL, на сцені.

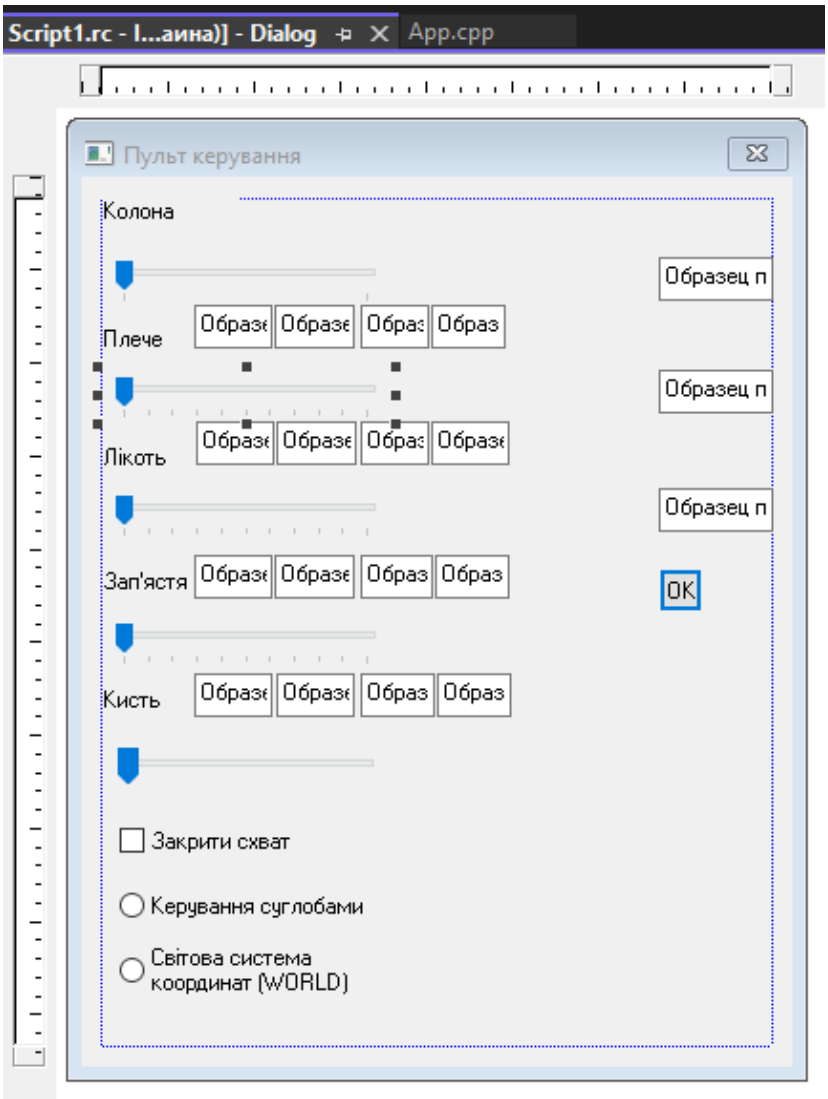

Рисунок 4.6 – Діалогове вікно кінематики з повзунками

4.3 Комп’ютерне моделювання системи автоматичного управління

У межах проєкту реалізовано повноцінне комп’ютерне моделювання системи автоматичного управління роботизованим маніпулятором, що дозволило протестувати функціональність механізмів, програмну логіку та зворотний зв’язок до створення фізичного пристрою.

На етапі проєктування геометрії всі елементи конструкції маніпулятора (база, ланки, захват, кріплення) були змодельовані у середовищі Fusion 360. Це забезпечило точну віртуальну візуалізацію кінематичних зв'язків, можливість перевірки рухливості системи, уникнення механічних колізій та оптимізацію

габаритів. Таким чином, ще до фізичного складання була змодельована уся конструкція й перевірена її працездатність.

На рівні програмного забезпечення створено власну візуалізацію роботи маніпулятора в середовищі Visual Studio 2022 з використанням OpenGL. Створена тривимірна сцена відображає положення маніпулятора в реальному часі, синхронізуючись із фактичними даними, що надходять з Arduino через COM-порт. Графічний інтерфейс дозволяє користувачу не лише задавати положення за допомогою повзунків або координат, а й отримувати зворотний зв’язок у вигляді зміни просторового положення моделі. Ця частина системи є авторською спробою створити власну альтернативу RoboDK – універсального середовища для моделювання роботів.

Крім візуалізації, моделювання охоплює й математичні обчислення – у програмі реалізовано розв'язання задачі зворотної кінематики, зокрема метод JointsGen(), який виконує числову оптимізацію для знаходження необхідних кутів повороту сервомеханізмів, виходячи з бажаної кінцевої позиції. Це дозволяє симулювати поведінку маніпулятора ще до його фізичного переміщення, що значно підвищує надійність і точність керування.

У функції JointsGen() реалізується зворотна задача кінематики для обчислення кутів суглобів на основі геометричних параметрів маніпулятора. Розрахунки базуються на тригонометричних співвідношеннях у формулах (4.1, 4.2).

Обчислення координат точки:

$$a = L_2 \cdot \cos(\alpha_2) - (L_3 + L_4) \cdot \sin(\varphi_{23}), \tag{4.1}$$

$$a = L_2 \cdot \sin(\alpha_2) - (L_3 + L_4) \cdot \cos(\varphi_{23}), \tag{4.2}$$

де α_2 – кут у плечі;

φ_{23} – загальний кут до цілі.

Кут у плечі (використовується арктангенс) (4.3, 4.4):

$$q_2 = \arctan\left(\frac{b \cdot L_2 \cdot (1 - \sin(\alpha_3) - a \cdot cos(\alpha_3) \cdot (L_3 + L_4)}{a \cdot L_2 \cdot (1 - sin(\alpha_3)) - b \cdot cos(\alpha_3) \cdot (L_3 + L_4)}\right). \quad (4.3)$$

Кут у лікті:

$$q_3 = \arcsin\left(\frac{L_2^2 + (L_3 + L_4)^2 - a^2 - b^2}{2 \cdot L_2 \cdot (L_3 + L_4)}\right). \quad (4.4)$$

Умова зупинки перебору (4.5):

$$|q_2 + q_3 - \varphi_{23}| < \varepsilon, \quad (4.5)$$

де ε – похибка наближення (в коді 0,1 радіан ≈ 5,7°).

Перетворення з радіан у градуси (для відображення) (4.6):

$$\theta_{\text{кут}} = \frac{180 \cdot \theta_{\text{рад}}}{\pi}. \quad (4.6)$$

Розрахунок передавальної функції сервомотора MG996R для перевірки на стійкість є завданням, що вимагає розуміння його внутрішньої структури та динаміки.

Типовий сервомотор складається з:

– двигуна постійного струму (dc-мотор): має свою електричну та механічну динаміку;

– редуктора: зменшує швидкість і збільшує крутний момент. вносить люфти та тертя;

– потенціометра: забезпечує зворотний зв'язок за кутовим положенням;

– контролера: зазвичай під-регулятор (або його варіації), який порівнює бажане положення (з ШІМ-сигналу) з фактичним (з потенціометра) і генерує керуючий сигнал для двигуна.

Передавальна функція типового сервомотора другого порядку має вигляд (4.7):

$$G(s) = \frac{\theta_{\text{факт}}(s)}{\theta_{\text{баж}}(s)} = \frac{K \cdot \omega_n^2}{s^2 + 2 \cdot \zeta \cdot \omega_{n^8} + \omega_n^2} \qquad (4.7)$$

де s – оператор Лапласа;

K – коефіцієнт підсилення (gain): показує, наскільки сильно вихідне положення реагує на зміну бажаного положення. В ідеалі K має бути близьким до 1 для точності позиціонування;

ωn – власна частота без загасання (natural frequency): це частота коливань системи, якби не було демпфування. Впливає на швидкість реакції системи;

ζ – коефіцієнт загасання (damping ratio): визначає, наскільки швидко загасають коливання системи після збурення.

Приклад передавальної функції:

Нехай K = 1 (якщо вхідний сигнал вже є бажаним кутом, що подається на внутрішній контролер сервопривода).

Візьмемо ωn = 20 рад/с та ζ = 0,6.

Тоді передавальна функція буде (4.8):

$$G(s) = \frac{1 \cdot (20)^2}{s^2 + 2 \cdot 0{,}6 \cdot 20 \cdot s + (20)^2} = \frac{400}{s^2 + 24 \cdot s + 400} \qquad (4.8)$$

В Matlab: W = tf([400], [1 24 400]).

Команда step(W) (рис. 4.7):

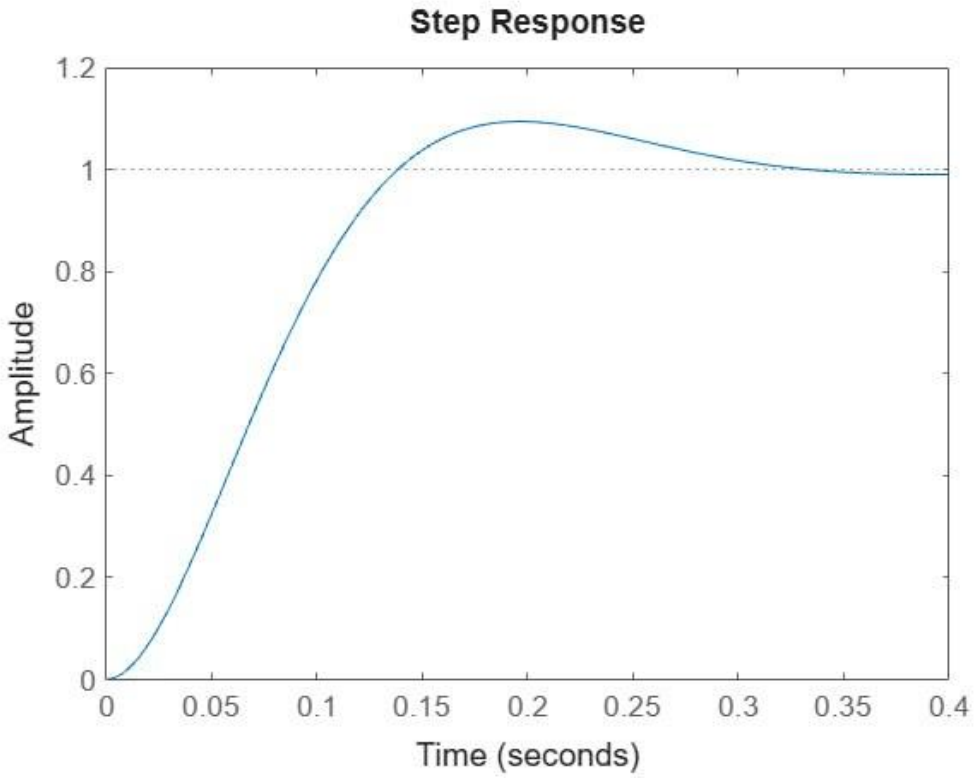

Рисунок 4.7 – Крокова відповідь

Команда nyquist(W) (рис. 4.8):

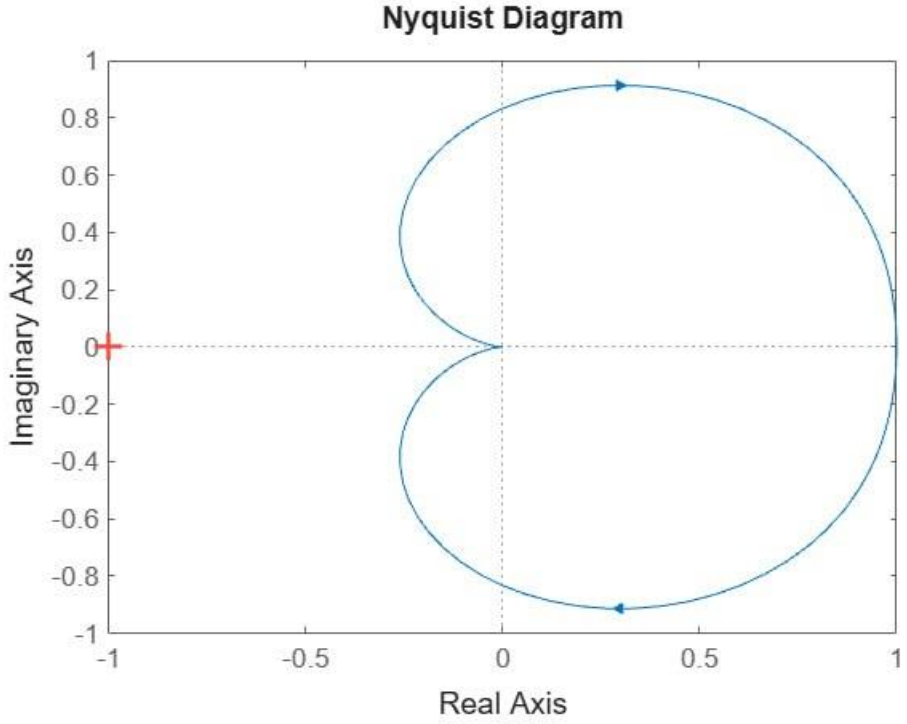

Рисунок 4.8 – Діаграма Найквіста

Команда pzmap(W) (рис. 4.9):

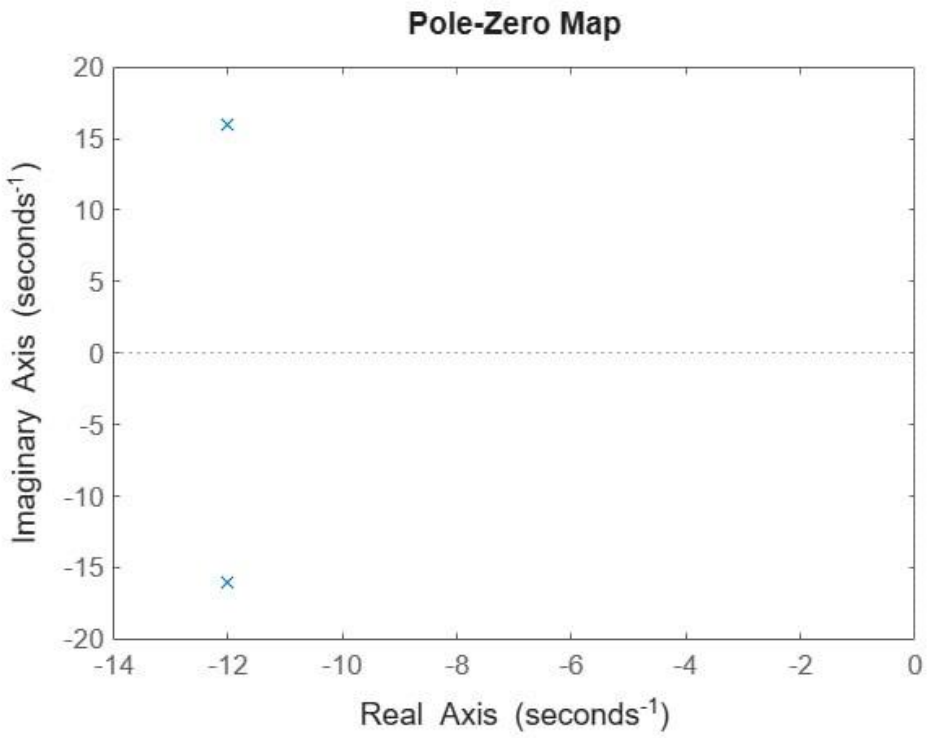

Рисунок 4.9 – Карта нульового полюса

Таким чином, комп’ютерне моделювання в даній роботі виконує не лише роль допоміжного інструмента, а є ключовим етапом розробки системи. Завдяки йому вдалося повноцінно протестувати логіку, рухи та алгоритми, досягти узгодженості між апаратною і програмною частинами та забезпечити зручну взаємодію з користувачем через інтуїтивний інтерфейс.

4.4 Охорона праці

Розробка програмного забезпечення проводиться в спеціальному приміщенні. Воно має параметри: п’ять робочих місць; один ПК (с світлодіодним монітором). Електрична мережа приміщення має такі характеристики: трифазна чотирипровідна мережа напругою 380/220 В

змінного струму, частота 50 Гц, з глухозаземленою нейтраллю. Функціональна схема одного робочого місця представлена на рисунку 4.10.

Рисунок 4.10 – Функціональна схема робочого місця

Приміщення відділу для розробки програмного забезпечення відноситься до приміщень без підвищеної небезпеки поразки людей електричним струмом згідно з НПАОП 40.1–1.21–98, так як немає умов створюють підвищену або особливу небезпеку. У приміщенні знаходиться електрощит, на якому встановлено пристрій струмового захисту. Всі розетки мають застережливий напис «220». З робітниками проводяться інструктажі з охорони праці відповідно до НПАОП 0.00-4.12-05.

Згідно з вимогами НПАОП 0.00–4.12–05 необхідно проводити вступний, первинний на робочому місці, повторний, а при необхідності – позаплановий і цільовий інструктажі.

Схема занулення представлена на рисунку 4.11, параметри цієї схеми відображені в таблиці 4.1.

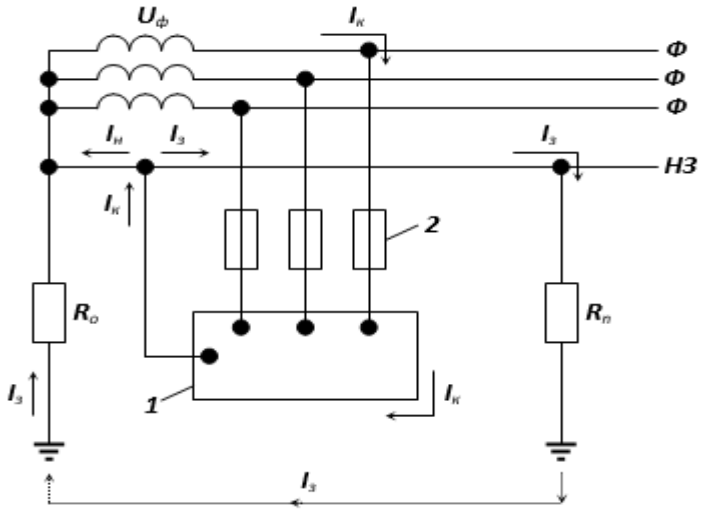

Рисунок 4.11 – Принципова схема занулення в трифазній мережі до 1000 В

Таблиця 4.1 – Параметри схеми занулення в трифазній мережі до 1000 В

1	корпус електроустановки;
2	апарати захисту від струмів короткого замикання (КЗ)
Ф	фазний провід;
НЗ	нульовий захисний провідник;
$R_о$	опір заземлення нейтрали обмотки джерела струму;
$R_п$	опір повторного заземлення нульового захисного провідника;
$I_к$	ток КЗ;
$I_н$	частина струму КЗ, що протікає через нульовий захисний провідник;
$I_з$	частина струму КЗ, що протікає через землю.

Проведено огляд робочого місця, визначенні основні параметри , наведено необхідні вимоги, визначенні можливі фактори небезпеки.

ВИСНОВКИ

Кваліфікаційна робота присвячена повноцінній реалізації інтерактивної системи керування роботизованим маніпулятором, що охоплює всі ключові етапи розробки сучасних робототехнічних систем – від ідеї й 3D-моделювання до візуалізації рухів у реальному часі на екрані та передачі команд між мікроконтролером і комп'ютером. Результатом проєкту стало створення багаторівневої системи, яка об'єднує апаратне та програмне забезпечення, візуальну симуляцію та ручне або дистанційне керування.

У процесі реалізації особливу увагу було приділено етапу конструювання, який здійснювався за допомогою платформи Fusion 360. У цьому середовищі було спроєктовано всі основні механічні елементи майбутнього маніпулятора, зокрема базову частину, плече, передпліччя, захват, корпус керування та опорні елементи. Віртуальне моделювання дозволило уникнути помилок у геометричних з'єднаннях, оптимізувати розміри під конкретні сервомеханізми та сформувати конструкцію, що може бути зручно виготовлена з доступних матеріалів, зокрема методом 3D-друку. Сам робот-маніпулятор був спроєктований як компактний, мобільний та універсальний пристрій, який за необхідності можна легко переміщувати, підключати до інших систем або змінювати функціональність. Його конструкція передбачає можливість модульного підходу – наприклад, зміну насадки чи захватного механізму (клешні) на інший, відповідно до цільового завдання.

З апаратної точки зору, керування реалізоване на базі мікроконтролера Arduino Uno Rev3, до якого підключено п’ять сервоприводів, два аналогові джойстики, блок живлення та допоміжні компоненти. Усі електричні з’єднання й алгоритми побудовані з урахуванням двох основних режимів: ручного – через фізичне керування джойстиками, та координатного – через

комп’ютерний інтерфейс. Стан системи зберігається в енергонезалежній пам’яті EEPROM, що дає змогу відновити останню конфігурацію навіть після вимкнення живлення.

Програмне забезпечення було реалізоване у вигляді двох незалежних компонентів. Перша частина – це прошивка для мікроконтролера Arduino, яка виконує обробку сигналів, керування сервоприводами, обробку кнопок та режимів, а також зчитування й передачу даних через COM-порт. Друга частина – це комп’ютерний додаток, написаний у середовищі Visual Studio 2022 із використанням бібліотек OpenGL для візуалізації. Графічний інтерфейс, створений у рамках цього проєкту, став авторською альтернативою програмам на зразок RoboDK – користувач отримує можливість у реальному часі спостерігати за рухом 3D-моделі маніпулятора на екрані, керувати його положенням за допомогою слайдерів, задавати координати, а також надсилати команди через серійний порт. Таким чином, розроблений графічний інтерфейс не лише доповнює керування, а й виконує функції діагностики, симуляції та візуального супроводу процесу.

Ключовими елементами програмної логіки стали методи зчитування аналогових сигналів (analogRead()), збереження в EEPROM (EEPROM.update()), застосування положень до сервомеханізмів (servo.write()), а також функції обробки й інтерпретації команд з комп’ютера. Уся система працює у режимі двостороннього зв’язку, що дозволяє підтримувати актуальний стан сервомеханізмів та інтерфейсу, незалежно від напрямку взаємодії.

Проєкт має значний потенціал для подальшого розширення. Завдяки використанню відкритих платформ та стандартів його можна легко адаптувати під інші моделі маніпуляторів, додати зворотний зв’язок, візуальні сенсори, функції машинного навчання або навіть інтегрувати у ROS-середовище для розробки автономних роботизованих систем.

Аналіз конструкцій, захватних пристроїв, програмного та апаратного забезпечення роботів-маніпуляторів дозволяє зробити висновок, що сучасна робототехніка є надзвичайно динамічною й міждисциплінарною галуззю, яка об’єднує механіку, електроніку, інформатику, штучний інтелект та автоматизацію. Захватні пристрої, як ключовий елемент взаємодії робота з об'єктами, також зазнали значного розвитку: від простих двопальцевих механізмів до складних адаптивних кінцівок і спеціалізованих інструментів. Їхня ефективність забезпечується завдяки інтеграції сенсорики, прецизійних механізмів і розумного керування.

Не менш важливими є апаратні та програмні складові, які дозволяють реалізовувати складні алгоритми планування руху, адаптації до змін середовища, візуального розпізнавання та навчання на основі даних. Застосування ROS, хмарних платформ, алгоритмів машинного навчання та віртуального моделювання стало новим етапом в еволюції робототехнічних систем.

Отже, представлена система є прикладом комплексного підходу до створення сучасного маніпулятора, який об’єднує 3D-моделювання, мікроелектроніку, алгоритмічне програмування та візуалізацію. Ефективність і надійність цього робота досягається через повну інтеграцію усіх його підсистем – механіки, електроніки, програмного забезпечення та інтерфейсу користувача. Такий підхід дозволяє створювати роботизовані рішення, які не лише виконують задані функції, а й адаптуються до вимог конкретного середовища, підтримують зручну взаємодію з оператором і відкривають широкі перспективи для застосування у виробництві, навчанні, логістиці та дослідженнях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України, 2017. – 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології освітньої програми «Системна інженерія» / Упоряд.: І.Ш. Невлюдов, О.М. Цимбал, О.В. Токарєва, А.І. Бронніков. Харків: ХНУРЕ, 2022. – 66 с.

3. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології освітньої програми «Системна інженерія» / Упоряд.: І.Ш. Невлюдов, О.М. Цимбал, О.В. Токарєва, А.І. Бронніков. Харків: ХНУРЕ, 2022. – 66 с.

4. Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G.. Robotics: Modelling, Planning and Control (2nd ed.). Springer, 2009. – 644 p.

5. Craig, J. J. (2005). Introduction to Robotics: Mechanics and Control (3rd ed.). Pearson Prentice Hall, 2005. – 387 p.

6. Groover, M. P. (2015). Automation, Production Systems, and Computer-Integrated Manufacturing (4th ed.). Pearson, 2016. – 82 p.

7. Spong, M. W., Hutchinson, S., & Vidyasagar, M.. Robot Modeling and Control. Wiley, 2006. – 419 p.

8. Kragten, G. A., & van der Helm, F. C. T. (2010). The Design of an Underactuated Robotic Hand. Springer, 2012. 741 – 755 p.

9. Bicchi, A., & Kumar, V. (2000). Robotic grasping and contact: A review. Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings, 1, 348 – 353 p. [Електронний ресурс] / – Режим доступу: https://doi.org/10.1109/ROBOT.2000.844081

10. Billard, A., & Kragic, D. (2019). Trends and challenges in robot manipulation. Science, 364(6446), eaat8414, 2019. 8 p. [Електронний ресурс] / – Режим доступу: https://doi.org/10.1126/science.aat8414

11. ISO 9787:2013. (2013). Robots and robotic devices – Coordinate systems and motion nomenclatures. International Organization for Standardization, 2013. 12 p.

12. Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., & Quillen, D. (2018). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research, 37(4 – 5), 421 – 436 p. [Електронний ресурс] / – Режим доступу: https://doi.org/10.1177/0278364917710318

13. Afsoon Afzal, Deborah S. Katz, Claire Le Goues, and Christopher S. Timperley. 2021. Simulation for Robotics Test Automation: Developer Perspectives. In Conference on Software Testing, Verification and Validation (ICST '14). IEEE, 263-274 p.

14. Охорона праці та цивільний захист [Електронний ресурс] : навчальний посібник / КПІ ім. Ігоря Сікорського ; уклад.: О. Г. Левченко. Електронні текстові данні (1 файл: 7,22 Мбайт). Київ : КПІ ім. Ігоря Сікорського, 2024. – 362 с.

15. Petar Kormushev, Sylvain Calinon &Darwin G. Caldwell. Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input, 2012. 581 – 603 p. [Електронний ресурс] / – Режим доступу: https://www.tandfonline.com/doi/abs/10.1163/016918611X558261

16. І.І. Павленко, М.О. Годунко. Захватні пристрої роботів. Кропивницький, 2020. 390 с. [Електронний ресурс] / – Режим доступу: http://www.kgemt.org.ua/pdf/about_colege/library_fund/zakhvatni_prystroyi_robotiv_pavlenko_ii_hkg3mwxip8.pdf

17. Manav Raj, Robert Seamans (2019). Primer on artificial intelligence and robotics. Journal of Organization Design 8(1), 2019. 14 p.

18. Невлюдов, І.Ш. Теорія автоматичного управління (збірник задач) [Текст]: навчальний посібник / І.Ш.Невлюдов, О.В.Токарєва. – Харків: ХНУРЕ, 2020. – 240 с.