МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ

Кафедра автоматизованого управління технологічними процесами
(повна назва кафедри)

Кваліфікаційна робота (проект)

магістр
(рівень вищої освіти)

на тему "Розробка інтелектуальної системи управління сортувальним комплексом"

Виконав: студент II курсу, групи ВІ-15-Ц
спеціальності 151 - автоматизація та комп'ютерно-інтегровані технології
(код і назва спеціальності)
спеціалізації _____
(код і назва спеціалізації)
освітньої програми автоматизація та комп'ютерно-інтегровані технології
(назва освітньої програми)
Пономаренко К. А.
(ініціали та прізвище)

Керівник доц., к.т.н., доц. Мінейко Н.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент зост. дир. СВ, Альтера Запоріжжя" Крот О.Г.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ

Кафедра автоматизованого управління технологічними процесами
Рівень вищої освіти магістр
Спеціальність 151 - автоматизація та комп'ютерно-інтегровані технології
(код та назва)
Спеціалізація _____
(код та назва)
Освітня програма автоматизація та комп'ютерно-інтегровані технології

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
«____» _____ 2020 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЄКТ) СТУДЕНТОВІ (СТУДЕНТЦІ)

Пономаренко Костянтин Андійович
(прізвище, ім'я, по батькові)

1 Тема роботи (проекту) „Розробка інтелектуальної системи управління сортувальним комплексом"
керівник роботи доц. к.т.н. Мінєйло К.О.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ЗНУ від «25» 05 20 20 року № 597-С
2 Строк подання студентом роботи 06.12.2020
3 Вихідні дані до роботи проаналізувати існуючи засновки сортувальних комплексів, дослідити принципи розпізнавання. Визначення закономірностей вибору пов. сер. по роботу комплексу. Наукова-технічна література.
4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Вступ, Розділ 1 Аналіз існуюч. установок. сортув., Розділ 2. Дослідж. принц. розпізнавання, Розділ 3 Вибір прилад та інженерної кінематики, Розділ 4 Практич. ре
5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Технічні засоби сортувального комплексу, дані отримані в ході дослідження прототипу „Очисті", конструкція робота з положеннями двигунів в. структурна схема функціональних частин лабораторного комплексу, результати розпізнавання за допомогою нейронної мережі YOLOv3, інтерфейс програми управління лабораторним комплексом „Очисті" в ручному та автоматичному режимах

6 Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата завдання видав	Підпис, дата завдання прийняв
Розділ 1	доц. к.т.н., доц. Мінейло К.О.	16.09.20	22.09.20
Розділ 2	доц. к.т.н., доц. Мінейло К.О.	29.09.20	22.10.20
Розділ 3	доц. к.т.н., доц. Мінейло К.О.	23.10.20	12.11.20
Розділ 4	доц. к.т.н., доц. Мінейло К.О.	13.11.20	30.11.20

7 Дата видачі завдання 01.09.2020

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Вступ	01.09.20-15.09.20	
2.	Технології сортування. Аналіз існуючих технологій сортувальних комплексів	16.09.20 - 28.09.20	
3.	Дослідження принципів розпізнавання за допомогою машинного зору	29.09.20 - 22.10.20	
4.	Вирішення прямої і інверсної задач кінематики	23.10.20 - 12.11.20	
5.	Практична реалізація рішень	13.11.20 - 30.11.20	
6.	Висновки	01.12.20 - 06.12.20	

Студент _____ Пономаренко К.А.
(підпис) (ініціали та прізвище)

Керівник роботи (проекту) _____ Мінейло К.О.
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ Овчинникова Т.А.
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

Пономаренко К.А. Розробка інтелектуальної системи управління сортувальним комплексом.

Кваліфікаційна випускна робота для здобуття ступеня вищої освіти магістра за спеціальністю 151 - Автоматизація та комп'ютерно-інтегровані технології, науковий керівник Н.О. Міняйло. Запорізький національний університет. Інженерний навчально науковий інститут. Кафедра автоматизованого управління технологічними процесами. Запоріжжя, 2020.

В роботі проаналізовано існуючі установки сортувальних комплексів. Досліджено роботу навчального комплексу «Orion-I». Розроблено методи вдосконалення навчального комплексу.

СОРТУВАННЯ, ВИРІБ, РОБОТ, УПРАВЛІННЯ, ПРОЕКТУВАННЯ, МОДЕЛЮВАННЯ, МОДЕРНІЗАЦІЯ, КОНТРОЛЕР, ПРОГРАМНЕ ЗАБЕЗПЕЧЕНЯ, РОЗПІЗНАВАННЯ, НЕЙРОННА МЕРЕЖА, КІНЕМАТИКА

1. Міняйло Н. О., Пономаренко К. А., Просвірнін А. Л., Востоцький С. М. Прототип системи управління сортувальним комплексом. Металургія : наукові праці Інженерного інституту Запорізького національного університету. . 2019. № 1. С. 61-66.

2. Міняйло Н. О., Пономаренко К. А. Інтелектуальна система управління роботою сортувального комплексу. Збірник наукових праць студентів, аспірантів і молодих вчених «Молода наука-2020» Запоріжжя: Запорізький національний університет, 2020. Т. 5 С. 125-126. URL: http://sites.znu.edu.ua/stud-sci-soc/tom_5.pdf.

3. Пономаренко К. А., Міняйло Н. О. Прототип роботизованого сортувального комплексу. Матеріали XXIV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Металургія як основа сучасної промисловості. Запоріжжя: ІІ ЗНУ, 2019. Т. 1 С. 70.

4.Пономаренко К. А., Міняйло Н. О. Удосконалення роботизованого комплексу за рахунок інверсної кінематики. Матеріали XXV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Запоріжжя: ІННІ ЗНУ, 2020. С. 70.

ABSTRACT

Ponomarenko K.A. Development of an intelligent system for managing the sorting complex.

Qualifying final work for obtaining a master's degree in higher education by specialty 151 - Automation and computer-integrated technologies, supervisor N.A. Minyaylo. Zaporizhzhia National University. Engineering Educational Institute. Department of Automated Process Control. Zaporizhzhya, 2020.

The work analyzed the existing installations of sorting complexes. The work of the Orion-I training complex was investigated. Methods for improving the educational complex have been developed.

SORTING, PRODUCT, ROBOT, CONTROL, DESIGN, MODELING, MODERNIZATION, CONTROLLER, SOFTWARE, RECOGNITION, NEURAL NETWORK, KINEMATICS

1. Minyaylo N. O., Ponomarenko K. A., Prosvirnin A. L., Vostotsky S. M. Prototype of the management system of the sorting complex. Metallurgy: scientific works of the Engineering Institute of Zaporizhzhya National University. 2019. № 1. C. 61-66.

2. Minyaylo N. O., Ponomarenko K. A. Intelligent system for controlling the operation of the sorting complex. Collection of scientific works of students, graduate students and young scientists "Young Science-2020" Zaporizhzhya: Zaporizhzhya National University, 2020. T. 5 C. 125-126. URL: http://sites.znu.edu.ua/stud-sci-soc/tom_5.pdf.

3. Ponomarenko K. A., Minyailo N.A. Prototype of a robotic sorting complex. Materials of the XXIV Scientific and Technical Conference of students, undergraduates, graduate students, young scientists and teachers. Metallurgy as the basis of modern industry. Zaporozhye: II ZNU, 2019. T. 1 S. 70.

4. Monomarenko K. A., Minyaylo N. A. Improvement of the robotic complex due to inverse kinematics. Materials of the XXV Scientific and Technical Conference of students, undergraduates, graduate students, young scientists and teachers. Zaporozhye: INNI ZNU, 2020. C. 70.

АНОТАЦИЯ

Пономаренко К.А. Разработка интеллектуальной системы управления сортировочным комплексом.

Квалификационная выпускная работа для получения степени высшего образования магистра по специальности 151 - Автоматизация и компьютерно-интегрированные технологии, научный руководитель Н.А. Миняйло. Запорожский национальный университет. Инженерный учебно научный институт. Кафедра автоматизированного управления технологическими процессами. Запороржье, 2020.

В работе проанализированы существующие установки сортировочных комплексов. Исследована работа учебного комплекса "Orion-I". Разработаны методы совершенствования учебного комплекса.

СОРТИРОВКА, ИЗДЕЛИЕ, РОБОТ, УПРАВЛЕНИЕ, ПРОЕКТИРОВАНИЕ, МОДЕЛИРОВАНИЕ, МОДЕРНИЗАЦИЯ, КОНТРОЛЛЕР, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, РАСПОЗНАВАНИЕ, НЕЙРОННАЯ СЕТЬ, КИНЕМАТИКА

1. Миняйло Н. О., Пономаренко К. А., Просвирнин А. Л., Востоцкий С. М. Прототип системы управления сортировочным комплексом. Металлургия: научные труды Инженерного института Запорожского национального университета. 2019. № 1. С. 61-66.

2. Миняйло Н. О., Пономаренко К. А. Интеллектуальная система управления работой сортировочного комплекса. Сборник научных работ студентов, аспирантов и молодых ученых "Молодая наука-2020" Запорожье: Запорожский национальный университет, 2020. Т. 5 С. 125-126. URL: http://sites.znu.edu.ua/stud-sci-soc/tom_5.pdf.

3. Пономаренко К. А., Миняйло Н.А. Прототип роботизированного сортировочного комплекса. Материалы XXIV научно-технической конференции студентов, магистрантов, аспирантов, молодых ученых и преподавателей. Металлургия как основа современной промышленности. Запорожье: II ЗНУ, 2019. Т. 1 С. 70.

4.Пономаренко К. А., Миняйло Н. А. Усовершенствование роботизированного комплекса за счет инверсной кинематики. Материалы XXV научно-технической конференции студентов, магистрантов, аспирантов, молодых ученых и преподавателей. Запорожье: ИННИ ЗНУ, 2020. С. 70.

ЗМІСТ

ВСТУП

Актуальність теми. Промислові роботи являються одним з компонентів автоматизованих виробничих систем, вживаних в гнучкому автоматизованому виробництві, які при незмінному рівні якості дозволяють збільшити продуктивність праці в цілому.

Мета і задачі роботи. Мета даної роботи – виконати аналіз роботи сортувального комплексу «Orion-I», визначити його недоліки які заважають коректному сортуванню об’єктів та виконати вдосконалення системи за рахунок введення кінематики та вдосконалення принципів розпізнавання.

Об’єкт дослідження. Процес сортування виробів що подаються на конвеєрній стрічці, за допомогою роботизованого комплексу.

Предмет дослідження. Інтелектуальна система управління роботою сортувального комплексу у складі автоматизованої системи управління.

Методи дослідження. Для опису позиції останньої ланки маніпулятора сортувального комплексу було використано аналітичний метод розрахунку прямої та інверсної завдань кінематики, що дозволило застосувати більш складні методи розпізнавання об’єктів сортування. Для детектування об’єктів, на конвеєрній стрічці, використано метод розпізнавання, на базі нейронної мережі YOLOv3 та фреймворку Darknet, що дозволяє не обмежуватися такими параметрами, як колір та електромагнітність.

Наукова новизна одержаних результатів. Полягає в дослідженні закономірностей впливу оточуючого середовища на роботу первинних перетворювачів у складі роботизованої системи сортування. Також навчання нейронної мережі YOLOv3 на базі фреймворку Darknet для розпізнавання об’єктів на конвеєрній стрічці, для подальшого їх сортування.

Практичне значення одержаних результатів. Підвищення якості розпізнавання об’єктів, використовуючи нейронну мережу, а також визначення їх положення на конвеєрній стрічці. На основі кінематичного розрахунку визначати кут оберту серводвигунів маніпулятора для досягнення

об’єкту сортування, використовуючи положення, отримані від нейронної мережі.

Особистий внесок дослідника. Проведено експериментальне дослідження роботи первинних перетворювачів роботизованого комплексу, аналіз статичних даних, дослідження методів розпізнавання, навчання нейронної мережі YOLOv3 на базі фреймворку Darknet, дослідження методів розрахунки кінематики для робота-маніпулятора, розрахунок прямої та інверсної кінематики за допомогою аналітичного методу, розробка програми взаємодії нейронної мережі та кінематики, програмування інтерфейсу оператора для управління роботою сортувального комплексу.

Апробація результатів роботи.

1. Міняйло Н. О., Пономаренко К. А. Інтелектуальна система управління роботою сортувального комплексу. Збірник наукових праць студентів, аспірантів і молодих вчених «Молода наука-2020» Запоріжжя: Запорізький національний університет, 2020. Т. 5 С. 125-126. URL: яhttp://sites.znu.edu.ua/stud-sci-soc/tom_5.pdf.

2. Пономаренко К. А., Міняйло Н. О. Прототип роботизованого сортувального комплексу. Матеріали XXIV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Металургія як основа сучасної промисловості. Запоріжжя: ІІ ЗНУ, 2019. Т. 1 С. 70.

3.Пономаренко К. А., Міняйло Н. О. Удосконалення роботизованого комплексу за рахунок інверсної кінематики. Матеріали XXV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Запоріжжя: ІННІ ЗНУ, 2020. С. 70.

Публікації.

1. Міняйло Н. О., Пономаренко К. А., Просвірнін А. Л., Востоцький С. М. Прототип системи управління сортувальним комплексом. Металургія : наукові праці Інженерного інституту Запорізького національного університету. . 2019. № 1. С. 61-66.

Структура та обсяг магістерської роботи. Дана магістерська робота, на тему «Інтелектуальна система управління роботою сортувальним комплексом», містить 94 сторінки, 4 додатки, 45 рисунки, 3 таблиці та 56 найменувань у переліку посилань.

РОЗІД 1
ТЕХНОЛОГІІ СОРТУВАННЯ. АНАЛІЗ ІСНУЮЧИХ УСТАНОВОК СОРТУВАЛЬНИХ КОМПЛЕКСІВ

1.1 Технологічний процес, як об’єкт автоматизації

Процедура сортування і укладання продукції в пакувальну тару існує практично на кожному виробничому підприємстві незалежно від обсягу виробництва. Процес завжди містить будь-які умови, які необхідно строго дотримуватися. Сортування товару по типом, матеріалом, кольору, формою, та іншими ознаками, подальша укладання його в різну тару, з огляду на ті ж ознаки, вибір обгорткового матеріалу прокладки матеріалів тощо. При цьому технологію упаковки / укладання необхідно завжди дотримуватися, тому що в процесі транспортування товару отримувачу можуть статися непередбачені випадки - від якості упаковки безпосередньо залежить цілісність вмісту тари. До того ж, якщо в сортуванні були допущені помилки, то повернення і заміна товару призведе до відчутної втрати часу і збоїв в логістичних процесах виробника і, можливо, покупця. Виходячи з вище сказаного, вкрай важливо підійти до цього завдання з усією відповідальністю [1].

Розрізняють ручне і автоматичне сортування. При ручному сортування розпізнавання потрібних матеріалів проводиться персоналом візуально, а відбір здійснюється вручну (при цьому окремі допоміжні операції можуть бути механізовані). На лініях автоматичне сортування ідентифікація матеріалів, що відбираються та їх виділення із загального потоку відбувається без участі персоналу.

При цьому в основі технологій автоматичного сортування зазвичай лежить використання сенсорів оптичного розпізнавання матеріалів -так зване оптичне сортування (optical sorting, sensor-based sorting).

Важливим напрямком підвищення ефективності автоматичного сортування відходів є збільшення ефективності відділення (сепарації) обраних

компонентів. В даний час в системах оптичного сортування матеріалів використовується три способи відділення ресурсних фракцій:

- відділення за допомогою повітряного струменя, що викидається з повітряного сопла (air nozzle, air jet) - так звана «повітряна» сепарація;

- відділення за допомогою системи відкидних заслінка (flap system);

- відділення за допомогою механічного маніпулятора за типом грейферного захоплення (mechanical fingers).

Автоматичне сортування і укладання допомагають уникнути людських помилок при здійсненні такої роботи. З урахуванням того, що даний процес пов'язаний з великою повторюваністю - для людини подібна діяльність є досить монотонним працею, який, на жаль, часто супроводжується помилками.

Обробка сміття - одна з областей, куди протягом останніх років активно приходять робототехнічні рішення. Число проектів з різних країн обчислюється десятками. Серед них, наприклад, промислові роботи-сортувальники від фінської компанії ZEN Robotics і американської AMP Robotics (і той і інший є маніпулятори, які вже працюють на кількох заводах), сортувальник від американської Waste Robotics, здатний вихоплювати з конвеєра пластикові пакети зі сміттям. Сортування відбувається за кольором, матеріалом, розміром та іншими ознаками [3].

Також, сортування широко використовується в харчовій промисловості. У кондитерському виробництві використовують машинний зір, який розпізнає вироби за необхідними критеріями, що дозволяє ефективно знаходити браковані вироби, які не підходять за кольором, формою або начинкою. У виробництві напоїв сортування відбувається за формою та кольором тари, за типом напоїв, а також їх мутністю.

Автоматизація сортування поштових посилок здатна зекономити багато часу та грошей. Наприклад, у центрі сортування відправлень Японії, сортування відбувається за допомогою QR – коду, в якому записана уся інформація про посилку, адресата та адресанта. Посилка переміщається по

конвеєру, а спеціальний датчик зчитує інформацію з QR - коду й заносить в базу, після чого відбувається автоматичне сортування по категоріям. Тобто, повністю виключається ручна робота [3].

Є ще багато сфер, де можна застосувати автоматичне сортування, наприклад, у медицині (сортування ліків за кольором препарату, за вагою, формою та змістом), у радіо-електронній промисловості (за розміром, номіналом, типом та кольором), у металургії (за якістю матеріалу, розміром та змістом та інші.

Сфери застосування технології сортування представлені на рисунку 1.1

Рисунок 1.1 – Сфери застосування технології сортування

1.2 Аналіз існуючих промислових роботів сортування

1.2.1 Робот для сортування таблеток Toshiba TH550

Робот функціонує під управлінням контролера Toshiba TS2000 з'єднаного з комп'ютером за інтерфейсом Ethernet. Для керування роботом використовується спеціалізована бібліотека DigiMetrix Robotics Library for LabVIEW, яка суттєво спрощує розробку подібних програм.

Захоплення і переміщення таблеток здійснюється маніпулятором робота за допомогою керованої пневматичної системи. Джерелом тиску

системи є повітряний компресор. Керування пневматичним каналом проводиться за допомогою цифрового виходу контролера робота.

Робот для сортування таблеток TH550 представлено на рисунку 1.2.

Рисунок 1.2 - Робот для сортування таблеток TH550

Пошук і класифікація таблеток здійснюється за допомогою мініатюрної камери Basler нової лінійки ace, закріпленої на корпусі 4-осьового робота-маніпулятора TH550. Зображення з камери передається в ноутбук за інтерфейсом Gigabit Ethernet, після чого здійснюється його обробка на верхньому рівні і формування команди контролеру робота на захоплення і пересування. За допомогою лінійки Basler ace технології Power over Ethernet, зовнішнє живлення камері не потрібно.

Камера для пошуку і класифікації таблеток Basler ace представлено на рисунку 1.3.

Рисунок 1.3 - Камера для пошуку і класифікації таблеток Basler ace

1.2.2 Робот для сортування відходів «Кларк»

У 2016 році компанія AMP Robotics навчила машину сортувати сміття. Для цього розробники на стару механізовану лінію встановили "очі" і "серце" - процесор з алгоритмом штучного інтелекту. Робота назвали Кларком. Його навички обмежені відбором картону, ламінованого пластиком - в такій упаковці продають йогурти, молоко, соки та інші продукти.

Кларк працює швидше людини. За хвилину в купі сміття він знаходить 60 предметів. Людина – 40 [4].

Робот використовує елементи штучного інтелекту для автоматичного розпізнавання різноманітних харчових упаковок і контейнерів. Комп'ютеру показали тисячі знімків найрізноманітніших пляшок, консервних банок, цеглин та іншого сміття і тепер система здатна самостійно розпізнавати їх на стрічці конвеєра [5].

Робот сортування відходів «Кларк» представлено на рисунку 1.4.

Рисунок 1.4 – Робот сортування відходів «Кларк»

1.2.3 Маніпулятор IRB 360 Flex Picker для сортування сміття

IRB 360 Flex Picker (іноді просто IRB 360) від компанії ABB - промисловий дельта-робот для сортування і упаковки готової продукції. Також здатний виконувати розвантажувально-навантажувальні роботи.

Зовні пристрій нагадує довгу прямокутну шафу. Крізь прозорі вікна останнього можна побачити головні робочі елементи конструкції - рухоме підґрунтя з прикріпленими до неї осями. Ці осі утворюють своєрідну «руку», за допомогою якої IRB 360 FlexPicker виконує свої функції.

Робот випускається в шести версіях. Кожна з них відрізняється один від одного вантажопідйомністю і розміром робочого діаметра, в рамках якого може функціонувати "рука". Монтується IRB 360 Flex Picker над конвеєрною стрічкою.

Функції і можливості:

- демонструє високу швидкість роботи як у широкому, так і у вузькому просторі з жорсткими вимогами до показника допустимого відхилення;

- здійснює більшу кількість захоплень на хвилину завдяки вдосконаленому фланцевому з 'єднанню і, як наслідок, дозволяє швидше розфасовувати упаковану продукцію навіть при швидкому русі конвеєрної стрічки;

- завдяки вбудованій технології комп'ютерного зору здатний сортувати предмети за різними критеріями і параметрами, зберігаючи високу швидкість і точність роботи;

- володіє гігієнічним дизайном, який дозволяє містити робота в чистоті без особливих зусиль [6].

Робот для сортування сміття IRB 360 Flex Picker представлено на рисунку 1.5.

Рисунок 1.5 – Робот для сортування сміття IRB 360 Flex Picker

1.2.4 Фінський Робот – сортувальник ZenRobotics

Фінський ZenRobotics був заснований в 2007 році. Робот ZenRobotics Recycler здатний піднімати і переміщати об'єкти вагою до 20 кг з продуктивністю до 2 тисяч на годину. Точність системи складає 98%, одна "рука" з поточною версією програмного забезпечення здатна виділити з потоку сміття до 4 фракцій. У конвеєра кожен робот займає ділянку 2х2 метра. Кілька ZenRobotics Recycler були закуплені США в 2016 році [7].

Сенсори системи постійно спостерігають за потоком твердих відходів, а програма аналізує дані в реальному часі.

Система ZenRobotics Recycler (ZRR) здатна сортувати метал, різні сорти деревини, мінерали, тверді пластмаси, картон. Також програма навчена помічати нестандартні об'єкти або нові види сміття.

Навчання ZRR відбувається за рахунок того, що системі показують зразки об'єктів, які потрібно розсортувати. За допомогою простого управління користувач може сам налаштувати програму, а доступ до зберігаються на хмарі рапортів здійснюється через комп'ютери, смартфони або планшети [8].

Робот ZenRoborics представлено на рисунку 1.6.

Рисунок 1.6 – Робот – сортувальник ZenRobotics

1.3 Аналіз існуючих лабораторних стендів по управлінню маніпуляторами

Для підготовки фахівців, які могли б проектувати та обслуговувати сучасні роботизовані комплекси на виробництві, необхідно створювати відповідні лабораторні установки та відкривати нові напрямки підготовки.

1.3.1 Навчальний маніпулятор «Оптіма – 1»

Навчальний маніпулятор Оптіма-1 - освітній комплекс, призначений для навчання учнів та студентів програмуванню мікроконтролерів, алгоритмізації процесів і основ кінематики. Маніпулятор виконаний на базі мікроконтролера Arduino Uno - одному з найпопулярніших на сьогоднішній день. Робототехнічний комплекс оснащений маніпулятором з трьома ступенями свободи і механічним захватом, роздавачем шайб з датчиком кольору і великої

кількістю допоміжних датчиків і пристроїв. Також робот може бути оснащений вакуумним захопленням (присоскою). Комплекс змонтовано на спеціальному столі з нанесеною розміткою і колірним маркуванням зон для вирішення практичних завдань з переміщення заготовок-шайб [9].

Навчальний маніпулятор «Оптіма – 1» представлено на рисунку 1.7.

Рисунок 1.7 - Навчальний маніпулятор «Оптіма – 1»

1.3.2 Лабораторний стенд САУ-РОБОТ-НН

Стенд "САУ Робот-Маніпулятор" призначений для виконання низки лабораторних робіт електротехнічних і технологічних спеціальностей, що вивчають дисципліни за системами автоматизації різних галузей промисловості. За допомогою лабораторного стенду можна вивчати технічні характеристики, способи програмування ПЛК Omron, а також реалізувати системи автоматизації на, базі робота - маніпулятора. Сучасна елементна база стенду дозволяє використовувати його не тільки в рамках навчального процесу, а й при проведенні науково-дослідних робіт [10].

Лабораторний стенд САУ-РОБОТ-НН представлено на рисунку 1.8.

Рисунок 1.8 - Лабораторний стенд САУ-РОБОТ-НН

1.3.3 Система автоматизації та управління робота – маніпулятора компанії «Новий Стиль»

Українською компанією «Новий стиль» розроблений стенд «Система автоматизації і управління робота-маніпулятора».

Стенд дозволяє вивчити технічні характеристики і систему програмування промислового контролера Omron, а також виконати систему автоматизації робота-маніпулятора.

До складу стенду входить:

- робот-маніпулятор УР-3;

- модуль промислового контролера Omron CPM-2A/CPM-1L з блоком управління; персональний комп'ютер/ноутбук;

- програмне забезпечення (компакт-диск);

- методичні вказівки до виконання лабораторних робіт;

- технічний опис.

На розробленому стенді проводяться лабораторні роботи по вивченню ПЛК Omron Sysmac CPM2A/CPM/1L, автоматизації управління приводу повороту/підйому робота, автоматизації управління робота-маніпулятора [11].

Система автоматизації та управління робота – маніпулятора компанії «Новий Стиль» представлено на рисунку 1.9.

Рисунок 1.9 Система автоматизації та управління робота – маніпулятора компанії «Новий Стиль»

1.3.4. Інтелектуальний робототехнічний комплекс ARM – 5

На кафедрі Математичне моделювання МГТУ ім. Н.Э. Баумана розроблений «Інтелектуальний робототехнічний комплекс ARM – 5».

Робототехнічний комплекс складається з робота-маніпулятора ARM - 5 і комп'ютера, оснащеного Web- камерою. Маніпулятор є п’ятизвенним, в якості приводів використовуються сервоприводи. Сервоприводи допрацьовані і дозволяють отримувати в контурі зворотного зв'язку реальні координати приводу. Ця особливість дозволяє порівнювати задане в програмі положення ланки робота з його реальним положенням в просторі і коригувати траєкторії ланок. Виконавчим облаштуванням робота є плоский захват.

Програмне забезпечення створене студентами кафедри «Математичне моделювання» МГТУ ім. Н.Э. Баумана, що навчаються за фахом «Прикладна математика».

Робототехнічний комплекс вирішує задачу побудови заданого об'єкту (наприклад, вежі або піраміди) з кубиків, розташування яких в робочій зоні робота заздалегідь невідоме.

Для визначення положення кубиків і їх кількості використовується Web - камера і спеціальне програмне забезпечення. По знімку робочої зони, отриманої з Web - камери, визначається кількість кубиків в робочій зоні і їх розташування. Програма, що потім управляє, планує послідовність, в якій переміщатимуться кубики, і формує сигнали, що управляють, передаються на виконавчі механізми робота.

З використанням зворотного зв'язку по положенню частин захвату здійснюється контроль випадання кубика із захвату в процесі переміщення. Якщо кубик втрачений під час переміщення, комплекс визначає його нове положення в робочій зоні і наново планує траєкторію і порядок переміщення.

Оператор тільки задає об'єкт, який необхідно побудувати. Далі усі операції комплекс виконує повністю автономно, без втручання оператора [12].

«Інтелектуальний робототехнічний комплекс ARM – 5» представлено на рисунку 1.10.

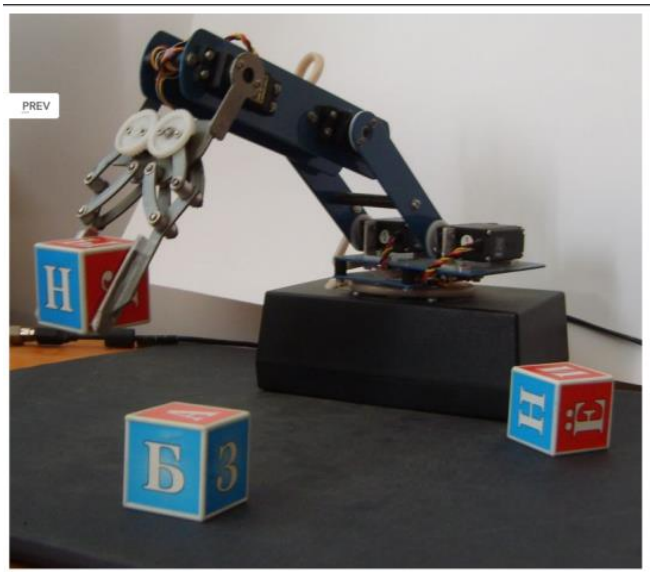

Рисунок 1.10 – «Інтелектуальний робототехнічний комплекс ARM – 5»

1.3.5 Коллоборативний робот-маніпулятор

У Білорусі компанією Rozum Robotics розроблений коллоборативний робот-маніпулятор.

Робот-маніпулятор схожий на людську руку, у нього є щось подібне до пальців, якими він захоплює об'єкт, може перекладати вантажі, наприклад з конвеєра в коробку, займатися тестуванням продукції. Йому під силу будь-які дії, що повторюються. Він може виконувати завдання в межах 70 см навколо себе і підіймати вантаж вагою 3 і більше кг Цей робот являється коллоборативним, оскільки він може працювати разом з людиною без захисних бар'єрів: у нього є інтелектуальна система відвертання зіткнення. Роботом можна управляти з планшета або смартфону по захищеній безпровідній мережі. Він може працювати 24 години на добу, 7 днів на тиждень. Проектування і складання робота робляться у Білорусі, а 95% компонентів - власна розробка компанії.

В основному застосування таких роботів планується в промисловості для оптимізації виробництва. Але зараз їх починають активно застосовувати в різних галузях - охороні здоров'я, сільському господарстві. Цього робота можна пристосувати практично для будь-якого виробництва: скрізь де є певні операції, дії, що повторюються, які можна автоматизувати. Робот може звільнити людину від нудної, монотонної роботи або роботи з небезпечними хімічними речовинами [13].

Коллоборативний робот – маніпулятор представлено на рисунку 1.11.

Рисунок 1.11 – Коллоборативний робот-маніпулятор

1.3.6 Робот-маніпулятор Parallax

На факультеті ПМ-ПУ (Прикладної математики – процесів управління) Санкт-петербурзький державного університету широко застосовуються сучасні комп'ютерні й інформаційні технології. До них можна віднести різні робототехнічні системи і об'єкти управління. Представлені подібні системи в двох лабораторіях:

1. Лабораторія комп'ютерного моделювання інформаційних систем :
   - людиноподібний робот «Вася»;
   - робот-маніпулятор Parallax;
   - установка кульки в магнітному підвісі.
2. Лабораторія мехатронних систем:
   - промисловий робот-маніпулятор FANUC M - 20.

Одна з основних дисциплін факультету - Теорія управління. Тому студентам надається реальна можливість перевірити отримані базові знання при програмуванні і формуванні законів управління спеціальних об'єктів управління.

Робот-маніпулятор канадської фірми Parallax являється 6-звенный механізмом, який представлено на рисунку 1.12.

Рисунок 1.12 – Робот-маніпулятор канадської фірми Parallax

Маніпулятор здійснює складний програмований рух. Для роботи з вказаним об'єктом необхідно знати основи програмування на високорівневих мовах, таких як C++. Об'єкт оснащений декількома енкодерами, що дозволяють отримувати положення кожної ланки. Що дає можливість побудувати управління із зворотним зв'язком. Застосування робота-маніпулятора обумовлене набором ситуацій, пов'язаних з підвищеною небезпекою для життя людини, а також трудомісткістю виконуваної роботи [14].

1.4 Аналіз лабораторного стенду «Orion-I»

1.4.1 Рівні системи автоматизації

У лабораторному комплексі наявні три рівні системи автоматизації.

На першому рівні знаходиться комплекс датчиків, який збирає інформацію з об’єкту, після чого, у вигляді цифрового сигналу, відправляє данні на другий рівень.

На другому рівні знаходиться контролер, який обробляє всю отриману інформацію з датчиків та генерує керуючий сигнал для робота - маніпулятора, відповідно характеристиці об’єкта сортування. Також, контролер посилає керуючий сигнал до конвеєру та обмінюється інформацією з третім рівнем.

На третьому рівні - АРМ оператора, що дозволяє оператору спостерігати за ходом процесу, що відбувається у лабораторному комплексі.

1.4.2 Визначення принципів управління по кожному технологічному параметру

Лабораторний роботизований комплекс призначений для сортування виробів за їх характеристиками. Робот – маніпулятор працює за двома режимами, в яких зберігаються встановлені характеристики для чотирьох зон сортування. В якості характеристик виступають два параметра: колір (червоний, жовтий, зелений і синій) та матеріал (метал і не метал). Об’єкт,

дерев’яний брусок з металевою або не металевою та кольоровою вставками, переміщається по стрічковому конвеєру в зону збору інформації з об’єкту, після зняття характеристик, він переміщується до зони забору об’єктів, де спрацьовує датчик наявності й керуючий сигнал, з контролеру, подається до робота-маніпулятора та конвеєра. Після зупинки конвеєра маніпулятор починає рух і переміщує об’єкт в одну із чотирьох зон, згідно з його характеристиками. В разі, якщо об’єкт не відповідає характеристикам будь якої із зон вибраного режиму, він продовжує рух повз зону забору об’єктів.

Для вибору режиму роботи лабораторного комплексу, а саме, вибору з наявних характеристик, які будуть притаманні чотирьом зонам, оператор переключає тумблер, тим самим змінює сигнал що поступає до контролеру.

1.4.3 Технічні засоби нижнього рівня СА

1.4.3.1 Первинні перетворювачі

Інфрачервоний датчик перешкоди YL-63 - невеликий датчик для визначення перешкод. Цифровий вихід датчика можна безпосередньо підключати до Arduino або іншим мікроконтролерам. Розмір 3,2x1,4 см, компаратор напруги LM393, відстань виявлення перешкод від 2 до 30 см. Напруга живлення від 3,3 В до 5 В. Для роботи у лабораторному комплексі, датчик було удосконалено, встановивши напроти нього інфрачервоний випромінювач, який до додав точності та відстані роботи датчика [15].

Датчик перешкоди YL-63 зображено на рисунку 1.13.

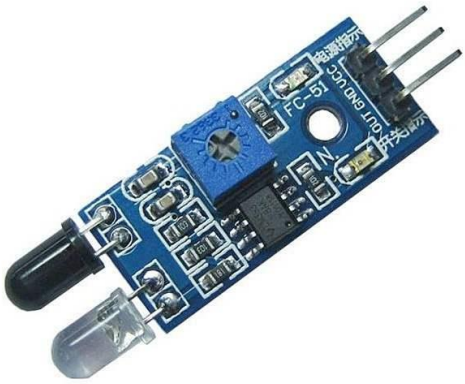

Рисунок 1.13 - Інфрачервоний датчик перешкоди YL-63

Індуктивний датчик APS-30-4N - 3-дротяний PNP датчик наближення, призначений для визначення наближення до датчика металевих поверхонь, магнітопровідних матеріалів, таких, як залізо та мідь. Датчик виявляє предмети без фізичного контакту в межах робочої зони. Часто такі датчики застосовують в системах автоматизації як кінцеві датчики положення. Вихід - нормально розімкнений. Відстань спрацьовування 4 мм. Напруга живлення від 6 В до 32 В [16].

Індуктивний датчик APS-30-4N зображено на рисунку 1.14.

Рисунок 1.14 - Індуктивний датчик APS-30-4N

Датчик кольору TCS230, розташований в центрі плати. Він складається з фотодіодів чотирьох типів: 16 фотодіодів з червоним фільтром, 16 фотодіодів з зеленим фільтром, 16 фотодіодів з синім фільтром і 16 фотодіодів без світлофільтра. До датчику підносять зразок одного кольорів, зразок висвітлюється світлодіодами на платі навколо датчика. Датчик має перетворювач струму в частоту, він перетворює свідчення фотодіодів в квадратну хвилю з частотою, пропорційною інтенсивності світла обраного кольору. Ця частота потім зчитується контролером Arduino. Вхідна напруга від 3 В до 5 В. Оптимальна відстань визначення кольору 5 мм [17].

Датчик кольору TCS230 зображено на рисунку 1.15.

Рисунок 1.15 – Датчик розпізнавання кольору TCS230

1.4.3.2 Пристрій моніторингу

LCD1602 - LCD дисплей для підключення до Arduino. Має два рядки по 16 символів в кожній. Працює зі стандартною бібліотекою LiquidCrystal з постачання Arduino IDE. Разом з ним використовується LCD1602 конвертор в IIC/I2C, який призначений для спрощення підключення дисплея до Arduino. LCD застосовується для індикація поточного стану лабораторного комплексу [18].

LCD1602 та конвертор в IIC/I2C зображені на рисунку 1.16.

Рисунок 1.16 – Дисплей LCD1602 та конвертор в IIC/I2C

1.4.3.3 Регулюючі органи

Сервопривід MG996R - це вдосконалена версія сервопривода MG995 TowerPro. Перепроектована друкована плата і система управління, що зробило сервопривід точнішим. Також вдосконалені редуктор і мотор, що зменшило мертву зону поліпшило центрування. Усі технічні характеристики точно такі, як у MG995, але новий сервопривід набагато точніше і безпечніше у використанні на об’єктах підвищеної точності. Напруга живлення від 4,8 В до 7,2 В. Вона має обертальний момент до 11 кг/см, що задовольняє висунутим вимогам. Так, як перша ланка вимагає обертальний момент більше 19,68 кг/см, встановлено два сервоприводи типу MG996R [19].

Загальний вигляд сервопривода MG996R представлені на рисунку 1.17.

Рисунок 1.17 – Сервопривід MG996R Tower Pro

1.4.3.4 Мікроконтролер

Мікроконтролер Arduino Mega вироблено в Італії. Програмування здійснюється на мові C++. Базується на потужному чипі ATmega2560, який працює на частоті 16 МГц. Має 54 цифрових вході/виходів, 14 з яких можуть працювати в режимі ШІМ (PWM), 16 аналогових входів, 4 апаратних послідовних портів UART для зв'язку з комп'ютером та іншими пристроями, роз'єм USB, роз'єм для зовнішнього живлення, ICSP хідер і кнопку скидання. Нова версія Rev3 включає в себе чіп Atmega16U2 з програмною прошивкою конвертера «USB-послідовний порт», замість використовуваних в більш ранніх версіях мікросхем FTDI, що дозволяє підвищити швидкість при передачі даних.

Управляючі сигнали, які формуються контролером, всього їх 7, являються дискретними, тому що управління сервоприводами MG995 і MG996R здійснюється за допомогою сигналу, який представляє собою імпульси постійної частоти та змінної ширини. Напруга живлення від 5 В до 20 В [20].

Мікроконтролер Arduino Mega 2560 Rev 3 зображено на рисунку 1.18.

Рисунок 1.18 – Мікроконтролер Arduino Mega 2560 Rev 3

1.4.3.5 Блок живлення та перетворювач напруги

Для функціонування лабораторного комплексу потрібне зовнішнє живлення, яке розраховується з ураховувати всіх компонентів: контролер Arduino Mega, 7 сервоприводів, LCD дисплей та три датчика, датчик положення, індуктивності та датчик кольору.

Інфрачервоний датчик споживає до 120 мА, індуктивний датчик до 300 мА, датчик кольору до 400 мА, сервоприводи в кількості 7-ми штук разом споживають до 4,2 А, а контролер 40 мА на один вхід/вихід.

З цим завдання справляється блок живлення MN-120-12 [21], який має вхідну напругу 220 В, 50 Гц змінного струму, а на виході напругу 12В постійного струму та 10 А. Так як, сервоприводи споживають напругу від 4,8 В до 7 В, необхідно використати регулюючий стабілізатор напруги XL4015 [22], який знижує напругу з 12 В до 6 В. Датчики та LCD дисплей живляться від контролера.

Блок живлення MN-120-12 та стабілізатор напруги XL4015 зображені на рисунку 1.19.

Рисунок 1.19 – Блок живлення MN-120-12 та стабілізатор напруги XL4015

1.5 Дослідження впливу оточуючого середовища на первинні перетворювачі

В ході роботи з лабораторним сортувальним комплексом «Orion-I» були виявлені помилки в системі зчитування параметрів з первинних перетворювачів. Для визначення причин негативного результату було вирішено виконати дослідження впливу оточуючого середовища, а саме рівня освітленості та температури приміщення, на роботу первинних перетворювачів. Для зняття параметру рівня освітленості було використано цифровий датчик BH1750, а для температури - DS18B20.

1.5.1 Датчик освітленості BH1750

Датчик освітленості BH1750 призначений для вимірювання фонового освітлення. Він являється 16-бітовим датчиком освітленості (люксметром) з інтерфейсом I2C. Ця мікросхема добре підходить для отримання даних про навколишнє освітлення. Фотодіод на BH1750 визначає інтенсивність світла, яка перетворюється на вихідну напругу за допомогою операційного підсилювача. Вбудований АЦП видає 16-бітові цифрові дані. Внутрішня

логіка BH1750 позбавляє від необхідності будь-яких складних обчислень, оскільки він безпосередньо виводить значущі цифрові дані в люксах (лк) [23].

Датчик освітленості BH1750 представлено на рисунку 1.20.

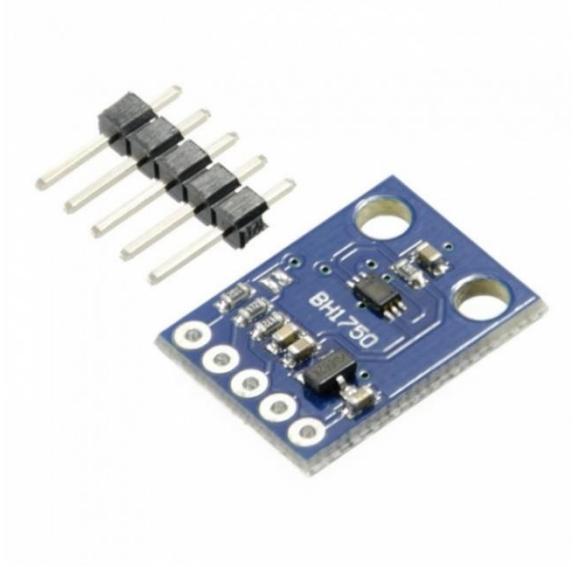

Рисунок 1.20 - Датчик освітленості BH1750

1.5.2 Датчик температури DS18B20

DS18B20 це цифровий вимірювач температури, з роздільною здатністю перетворення 9 - 12 розрядів і функцією тривожного сигналу контролю за температурою. Параметри контролю можуть бути задані користувачем і збережені в енергонезалежній пам'яті датчика.

DS18B20 обмінюється даними з мікроконтролером по однопровідній лінії зв'язку, використовуючи протокол інтерфейсу 1-Wire.

Живлення датчик може отримувати безпосередньо від лінії даних, без використання зовнішнього джерела. У цьому режимі живлення датчика походить від енергії, запасеної на паразитній ємності.

Діапазон вимірювання температури становить від -55 до + 125 ° С. Для діапазону від -10 до + 85 ° С похибка не перевищує 0,5 ° С [24].

Датчик температури DS18B20 представлено на рисунку 1.21.

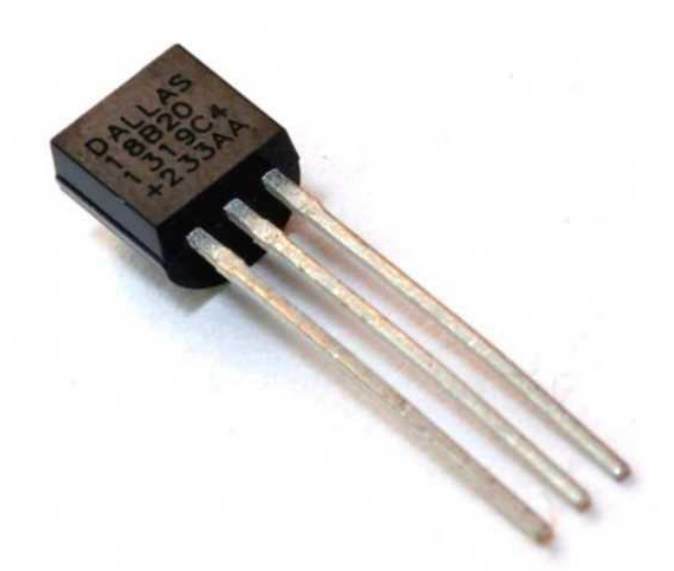

Рисунок 1.21 - Датчик температури DS18B20

1.6 Зняття параметрів датчиків освітленості й температури. Отримання результатів

Для дослідження роботи первинних датчиків, було проведено дослідження. Для цього обрана вибірка температури, від 15 до 25 градусів з кроком 2 градуси та рівнем освітленості від 17 тис. до 56 тис. люк. По десять перевірок роботи сортування на одне значення температури..

Результати роботи лабораторного комплексу «Orion-I» з використанням датчиків освітленості та температури представлено у таблиці 1.1

Таблиця 1.1 - Результати роботи лабораторного комплексу «Orion-I» з використанням датчиків освітленості та температури

№	Температура, °C	Рівень освітленості, люкс	Параметри датчиків					Відповідність завданню
			Датчик розпізнавання кольору			Індуктивний датчик	Інфрачервоний датчик	
			R	G	B			
1	15	17521	2	104	5	-	-	-

Продовження таблиці 1.1

2		21486	5	123	11	+	-	-
3		23597	7	130	17	+	+	+
4		25178	9	133	23	-	+	-
5		29537	13	151	32	-	+	-
6		32846	16	163	41	+	+	-
7		35167	20	198	49	-	+	-
8		39491	24	186	60	-	+	-
9		42437	28	195	69	-	-	-
10		46780	33	209	81	+	-	-
Виконаних завдань, %:								**10**
11	17	17247	2	103	5	+	-	-
12		21584	5	124	12	-	-	-
13		23681	7	131	18	-	+	+
14		24987	9	133	24	+	+	+
15		29428	13	152	34	-	+	-
16		33108	17	166	43	+	+	-
17		35123	20	171	50	-	+	-
18		39897	24	189	62	+	-	-
19		42421	28	196	71	-	-	-
20		46468	32	210	83	+	-	-
Виконаних завдань, %:								**10**
21	19	17234	3	104	6	+	-	-
22		21571	6	125	13	+	-	-
23		23668	8	132	19	+	+	+
24		24974	10	134	25	+	+	+
25		29415	14	153	35	+	+	+
26		33095	17	167	44	+	+	-
27		35110	21	172	51	+	+	-

Продовження таблиці 1.1

28		39884	25	190	63	+	-	-
29		42408	29	197	72	+	-	-
30		46455	33	210	83	+	-	-
Виконаних завдань, %:								**30**
31	21	17357	4	103	4	+	-	-
32		21694	7	122	10	+	-	-
33		23791	9	129	16	+	+	+
34		25097	11	132	22	+	+	+
35		29538	15	150	32	+	+	+
36		33218	18	162	41	+	+	-
37		35233	22	197	48	+	+	-
38		40007	26	185	60	+	-	-
39		42531	30	194	69	+	-	-
40		46578	33	207	79	+	-	-
Виконаних завдань, %:								**30**
41	23	17144	3	103	5	+	-	-
42		21481	5	121	11	+	-	-
43		23578	7	128	17	+	+	+
44		24884	10	131	24	+	+	+
45		29325	13	149	33	+	+	-
46		33005	16	161	42	+	+	-
47		35020	20	196	49	+	+	-
48		39794	24	184	61	+	-	-
49		42318	28	193	70	+	-	-
50		46365	30	205	79	+	-	-
Виконаних завдань, %:								**20**
51	25	17230	2	103	5	+	-	-
52		21567	6	129	17	+	-	+

Продовження таблиці 1.1

53		23664	7	131	18	+	+	+
54		24970	9	133	24	+	+	+
55		29411	11	148	28	+	+	+
56		33091	17	166	43	+	+	-
57		35106	20	171	50	+	+	-
58		39880	24	189	62	+	-	-
59		42404	28	196	71	+	-	-
60		46451	31	209	82	+	-	-
Виконаних завдань, %:								**40**

Виконавши дослідження роботи сортувального комплексу «Orion-I» з використанням датчиків освітленості та температури було виявлено, що використання поточних датчиків являється не доцільним, адже вони вразливі до зміни зовнішніх факторів, таких як: освітлення та температура приміщення.

При кожному дослідженні, значення виконаних завдань не перевищувало 40 %. Данні, отримані в ході експерименту зображені на рисунку 1.22.

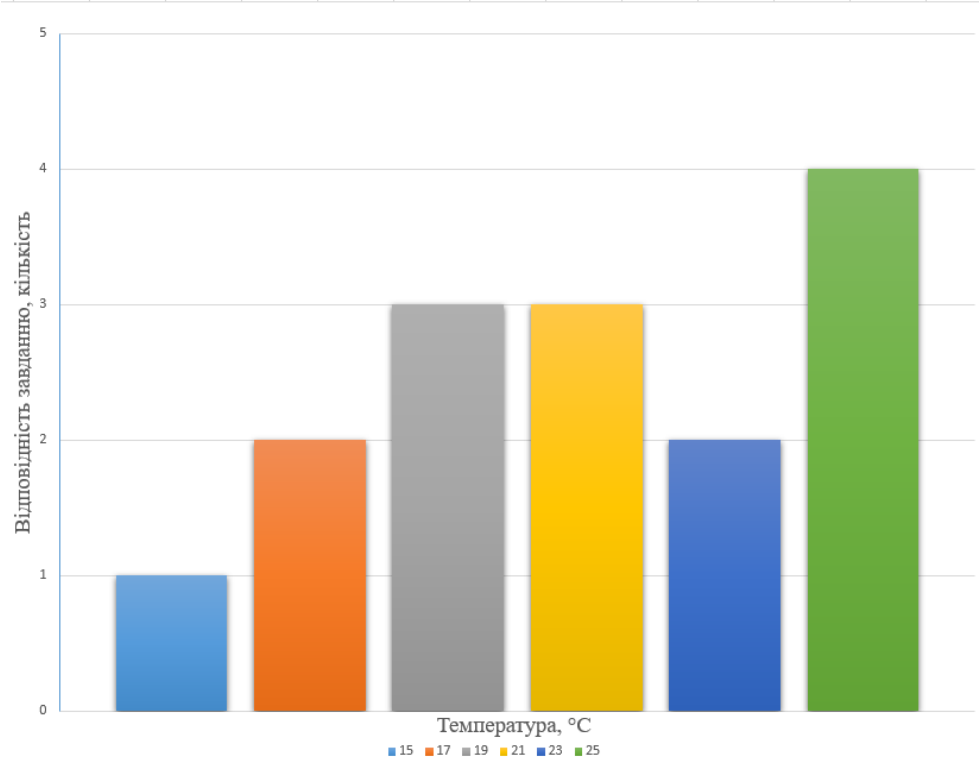

Рисунок 1.22 - Данні, отримані в ході експерименту

Також робота прототипу «Orion-I» сильно залежить від позиціювання об’єктів сортування, адже, на даний момент, для справної роботи сортувального комплексу, об’єкт повинен знаходитись строго в зоні забору маніпулятором.

Для збільшення спектру завдань, які зможе виконувати робот «Orion-I», а також видалення недоліків, необхідно провести ряд поліпшень, як конструкційних так і математичних.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ ПРИНЦИПІВ РОЗПІЗНАВАННЯ ЗА ДОПОМОГОЮ МАШИННОГО ЗОРУ

2.1 Основні поняття машинного зору

Машинний зір (Machine Vision, MV) - це автоматична фіксація та обробка зображень, як нерухомих, так і рухомих об'єктів за допомогою комп'ютерних засобів. В Україні також використовується термін «технічний зір».

Перші спроби змусити комп'ютер «бачити» відносяться до початку 60-х років 20 століття. Однак лише в останні роки у зв'язку з підвищенням обчислювальних потужностей і швидкодії процесорів, обсягів пам'яті, підвищенням роздільної здатності та інших параметрів камер, розвитком смуги пропускання каналів зв'язку, а також з появою таких технологій, як машинне і глибоке навчання (Machine/Deep Learning), штучний інтелект AI (Artificial Intelligence) технології CV/MV стали знаходити все більше застосувань в різних галузях індустрії і повсякденного життя людей [25].

Машинний зір зосереджується на застосуванні, в основному промисловому, наприклад, автономні роботи і системи візуальної перевірки та вимірювань. Це означає, що технології датчиків зображення і теорії управління пов'язані з обробкою відеоданих для управління роботом і обробка отриманих даних в реальному часі здійснюється програмно або апаратно.

Обробка зображень і аналіз зображень зосереджуються на роботі з 2D зображеннями, тобто як перетворити одне зображення на інше. Наприклад, попіксельні операції збільшення контрастності, дії з виділення країв, усунення шумів або геометричні перетворення, такі як обертання зображення. Дані операції передбачають, що обробка/аналіз зображення діють незалежно від змісту самих зображень.

Комп'ютерний зір зосереджується на обробці тривимірних сцен, спроектованих на одне або кілька зображень. Наприклад, відновленням

структури або іншої інформації про 3D сцену за одним або кількома зображеннями. Комп'ютерний зір часто залежить від більш-менш складних припущень щодо того, що представлено на зображеннях.

Також існує область, названа візуалізація, яка спочатку була пов'язана з процесом створення зображень, але іноді мала справу з обробкою та аналізом. Наприклад, рентгенографія працює з аналізом відеоданих медичного застосування.

Нарешті, розпізнавання образів є областю, яка використовує різні методи для отримання інформації з відеоданих, в основному, засновані на статистичному підході. Значна частина цієї області присвячена практичному застосуванню цих методів.

Таким чином, можна зробити висновок, що поняття «машинний зір» на сьогоднішній день включає в себе: комп'ютерний зір, розпізнавання зорових образів, аналіз та обробка зображень тощо.

До завдань, що розглядаються в рамках машинного зору, зокрема, входять:

1. Виявлення/розпізнавання/відстеження об'єктів, що мають певні властивості (в найширшому сенсі) на статичному зображенні та/або у відеопотоці.

Розпізнавання/виявлення. Класичне завдання в комп'ютерному зору, обробці зображень і машинному зору - це визначення чи містять відеодані певний характерний об'єкт, особливість або активність. Це завдання може бути достовірно і легко вирішене людиною, але досі не вирішене задовільно в комп'ютерному зору в загальному випадку: випадкові об'єкти у випадкових ситуаціях;

Існуючі методи для вирішення таких проблем можуть бути достовірно вирішені тільки для окремих об'єктів, таких як прості геометричні об'єкти, людські особи, друковані або рукописні символи, автомобілі і тільки в певних умовах, зазвичай це певне освітлення, фон і положення об'єкта щодо камери. У літературі описано різну безліч проблем розпізнавання:

- Розпізнавання: один або кілька попередньо заданих або вивчених об'єктів або класів об'єктів що можуть бути розпізнані, зазвичай разом з їх двомірним положенням на зображенні або тривимірним положенням у сцені.

- Ідентифікація: розпізнається індивідуальний екземпляр об'єкта належного до будь-якого класу.

- Виявлення: відеодані перевіряються на наявність певної умови. Виявлення, засноване на відносно простих і швидких обчисленнях, іноді використовується для знаходження невеликих ділянок в аналізованому зображенні, які потім аналізуються за допомогою прийомів, більш вимогливих до ресурсів, для отримання правильної інтерпретації.

Існує кілька спеціалізованих завдань, заснованих на розпізнаванні текстів, наприклад:

- Пошук зображень за змістом: знаходження всіх зображень у великому наборі зображень, які мають певний різними шляхами зміст.

- Оцінка положення: визначення положення або орієнтації певного об'єкта щодо камери.

- Оптичне розпізнавання знаків: розпізнавання символів на зображеннях друкованого або рукописного тексту, зазвичай для перекладу в текстовий формат, найбільш зручний для редагування або індексації (наприклад, ASCII).

- Відновлення 3D форми за 2D зображеннями за допомогою: стерео реконструкції карти глибини; реконструкції поля нормалів і карти глибини із зафарбовування півтонового зображення; реконструкції карти глибини за текстурою; визначення форми з переміщення.

Відновлення сцени. Дані два або більше зображення сцени, або відеодані. Відновлення сцени має завданням відтворити тривимірну модель сцени. У найпростішому випадку моделлю може бути набір точок тривимірного простору. Більш складні методи відтворюють повну тривимірну модель.

Відновлення зображень. Завдання відновлення зображень - це видалення шуму (шум датчика, розмитість рухомого об'єкта тощо). Найбільш простим підходом до вирішення цього завдання є різні типи фільтрів, таких як фільтри

нижніх або середніх частот. Більш високий рівень видалення шумів досягається в ході початкового аналізу відеоданих на наявність різних структур, таких як лінії або межі, а потім управління процесом фільтрації на основі цих даних.

- Виділення на зображеннях структур певного виду, сегментація зображень.

- Аналіз оптичного потоку (розташування пікселів між двома зображеннями). Кілька завдань, пов'язаних з оцінкою руху, в яких послідовність зображень (відеодані) обробляються для знаходження оцінки швидкості кожної точки зображення або 3D сцени. Прикладами таких завдань є: визначення тривимірного руху камери, стеження, тобто слідування за переміщеннями об'єкта (наприклад, машин або людей) [27].

2.2 Основні компоненти технічного зору

Основними чотирма складовими системи зору є об'єктив і система освітлення (підсвічування об'єкта), датчик зображення або камера, процесор і спосіб передачі результатів, будь то за допомогою фізичних входів/виходів (I/O) або за допомогою інших засобів комунікації на основі, як правило, стандартних протоколів і загальноприйнятих інтерфейсів.

Об'єктив захоплює зображення і передає його сенсору у вигляді світлової проекції. Щоб оптимізувати систему зору, відеокамера повинна мати відповідний об'єктив. Хоча існує багато типів об'єктивів, у програмах машинного зору для простоти управління зазвичай використовуються об'єктиви з фіксованою фокусною відстанню.

У процесі розпізнавання враховуються три визначальні фактори:

- поле зору;

- робоча відстань до об'єкта;

- розмір сенсора (матриці) камери.

Для отримання зображення, достатнього для його подальшої обробки та досягнення потрібної якості, існує багато різних способів передачі до цільового об'єкта необхідного рівня освітлення. Напрямок, з якого виходить світло, його яскравість, колір або довжина хвилі порівняно з кольором об'єкта - все це важливі елементи, які слід враховувати при проектуванні системи машинного зору для конкретного середовища застосування [28].

Для роботи в умовах недостатнього освітлення можуть використовуватися камери з підсвічуванням, в яких кільцеве джерело світла забезпечує яскраве рівномірне освітлення об'єкта, коли необхідно висвітити фактуру матеріалу, дрібні деталі тощо. Також освітлення допомагає позбутися від бліків, засвічування об'єкта, використовується в складних умовах, наприклад, в тумані [25].

Інтегроване джерело світла с дифузійним кільцем зображено на рисунку 2.1

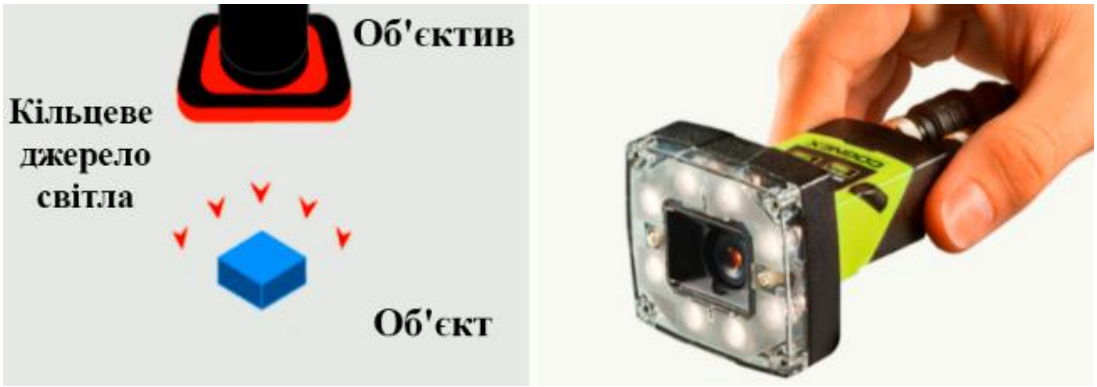

Рисунок 2.1 - Інтегроване джерело світла с дифузійним кільцем

2.3 Підходи рішення задач технічного зору

Основні підходи до вирішення завдань CV:

- контурний аналіз;
- пошук за шаблоном (template matching);
- пошук поза шаблонами, порівняння ключових точок (feature detection, description matching);
- суміщення даних (Data Fusion).

Технічний зір не обмежується тільки цими основними методами, наприклад, можна виділити так звані генетичні алгоритми, що застосовуються, зокрема, для розпізнавання осіб [25].

2.3.1 Контурний аналіз

Контурний аналіз - це один з важливих і дуже корисних методів опису, зберігання, розпізнавання, порівняння і пошуку графічних образів/об'єктів.

Контур - це зовнішні обриси (обведення) предмета/об'єкта.

При проведенні контурного аналізу:

- передбачається, що контур містить достатню інформацію про форму об'єкта;

- внутрішні точки об'єкта до уваги не приймаються.

Вищенаведені положення, зрозуміло, накладають суттєві обмеження на область застосування контурного аналізу, які, в основному, пов'язані з проблемами виділення контуру на зображеннях:

- через однакову яскравість з фоном об'єкт може не мати чіткої межі, або може бути зашумлений перешкодами, що призводить до неможливості виділення контуру;

- перекриття об'єктів або їх угруповання призводить до того, що контур виділяється неправильно і не відповідає межі об'єкта.

Однак перехід до розгляду лише контурів об'єктів дозволяє уникнути простору зображення - простору контурів, що значно зменшує складність алгоритмів та обчислень [29].

2.3.2 Пошук за шаблоном

Найпоширеніший метод розпізнавання об'єктів у CV - пошук відповідності шаблонам зображень (template matching), щоб визначити, чи є вказаний об'єкт на зображенні, і, якщо є, де він знаходиться на зображенні. Програми методу: розпізнавання транспортних засобів, прокладання

маршрутів для мобільних роботів, виробництво і додатки в медицині та інші [25]. Основні види пошуку за шаблоном представлено далі у розділі.

2.3.2.1. Відповідність на базі характерних ознак

Формування опису зображення у вигляді безлічі ХО - чи не найскладніше завдання в процесі побудови системи структурного розпізнавання. Вибір ознак часто залишається процедурою евристичною, ХО виступають базою для прийняття глобальних рішень про клас об'єкта. Головними вимогами є інваріантність до геометричних перетворень і перешкоджання. Ці властивості в сучасних системах комп'ютерного зору досягаються комплексним використанням інтегро-диференційного аналізу зображення на основі гаусівського згладжування. Диференціювання дозволяє виділити локальні особливості, а за рахунок функціонального інтегрування досягається інваріантність і перешкоджання. Принципи отримання інваріантного уявлення зводяться або до аналізу моментів після диференційної обробки (локальні потоки), або до представлення в розширених просторах значень перетворень, побудованих за аналогом кореляційної обробки (SIFT, SURF) [30].

2.3.2.2. Проста відповідність

Проста відповідність - один з основних методів знаходження об'єкта на зображенні під час пошуку за шаблоном. Метод полягає в покроковому скануванні шаблоном вихідного зображення, коли кожен крок вимірюється або обчислює ступінь відповідності ділянки зображення шаблону. У кінці сканування на зображенні виділяється область, яка найбільше відповідає шаблону [25].

2.3.2.3. Відповідність на базі областей

Методи знаходження відповідності на базі областей (Area-based), які також називаються кореляційними методами, засновані на комбінованому алгоритмі знаходження характерних рис, ХС (feature detection), і відповідності

шаблону (template matching). Такий метод добре працює, якщо шаблони не мають помітних загальних ХС із зображенням, оскільки порівняння відбувається на піксельному рівні. Відповідно вимірюються за показниками інтенсивності шаблону та зображення [31].

У деяких випадках знаходження прямої відповідності між шаблоном і зображенням неможливо (див. малюнок нижче). Отже, якщо ви знаходите відповідність, використовуйте власне значення (eigenvalue) та власний простір (eigenspace). Ці величини містять інформацію, необхідну для порівняння образів за різних умов освітленості, контрастності контурів або збігу за положенням об'єктів [25].

2.3.2.4. Нейромережі

ШНМ (штучні нейромережі) - це математична модель функціонування традиційних для живих організмів нейромереж, які являють собою мережі нервових клітин. Як і в біологічному аналозі, в штучних мережах основним елементом виступають нейрони, поєднані між собою і утворюючі шари, число яких може бути різним залежно від складності нейромережі та її призначення (вирішуваних завдань) [32].

Мінімальні обчислювальні елементи штучної нейронної мережі теж називаються нейронами. Нейронні мережі зазвичай складаються з трьох або більше шарів: вхідного, прихованого шару (або шарів) і вихідного шару, у деяких випадках вхідний та вихідний шари не враховуються, і тоді їх кількість у мережі вважається за кількістю прихованих шарів. Такий тип нейронної мережі називається перцептроном [33].

Найпростіший перцептрон зображено на рисунку 2.2.

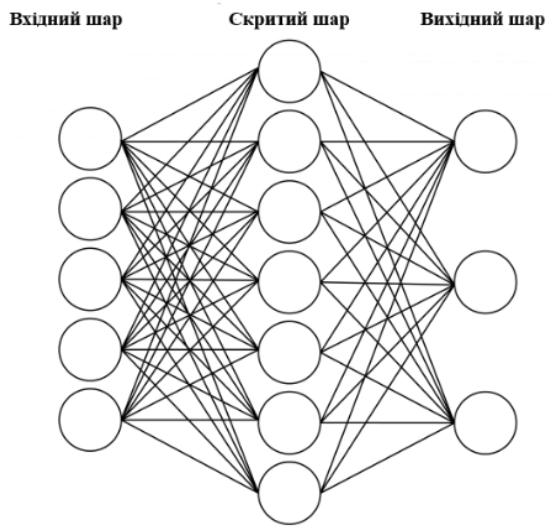

Рисунок 2.2 - Найпростіший перцептрон

2.3.2.5. Глибоке навчання

Глибоке навчання - це набір алгоритмів машинного навчання, які моделюють абстракції високого рівня в даних з використанням архітектур, що складаються з декількох нелінійних перетворень.

Технологія глибокого навчання заснована на штучних нейронних мережах (ШНМ). Ці ШНМ отримують алгоритми навчання і постійно зростаючі обсяги даних для підвищення ефективності процесів навчання. Чим більший обсяг даних, тим ефективніший цей процес. Процес навчання називається «глибоким», тому що з часом нейронна мережа покриває все більшу кількість рівнів. Чим глибше ця мережа проникає, тим вища її продуктивність [34].

Процес глибокого машинного навчання складається з двох основних етапів: навчання та формування висновків. Фазу навчання - це метод маркування великих обсягів даних та визначення їх відповідних характеристик. Система порівнює ці характеристики і запам'ятовує їх, щоб зробити правильні висновки, коли вона зіткнеться з подібними даними наступного разу.

Процес глибокого навчання включає наступні етапи:

- ШНМ ставлять набір двійкових запитань у вигляді да/ні;

- вилучення числових значень з блоків даних;

- класифікація даних відповідно до отриманих відповідей;

- маркування даних.

Під час фази формування висновків, система робить певні висновки, а потім маркує нові невивчені дані, використовуючи її попередні знання [34].

2.3.3 Комп'ютерний зір поза шаблонами

Набори ХС для обробки зображень у комп'ютерному зору можуть представляти елементи зображення, такі як точки, краї, лінії або межі об'єктів. Інші приклади ХС відносяться до руху в послідовності зображень, до форм, представлених у вигляді кривих між областями зображення, або до властивостей цих областей.

2.3.3.1 Фотограмметрія

Фотограмметрія дозволяє визначити по знімках досліджуваного об’єкта його форму розміри та просторове положення в заданій системі координат. А також його площину та об’єм.

Фотограмметрію використовують в різноманітних областях науки, техніки та виробництва. Наприклад:

- для визначення деформації споруд і їх окремих частин, які відбуваються в ході експлуатації та з плином часу. Наприклад, зрівняння вимірів, проведених по стереопарам моста чи підйомного крану, отриманих до їх навантаження, під час навантаження та після, дозволяє визначити їх деформацію в залежності від ваги навантаження;

- - для визначення характеристик рухомих об'єктів: транспортних засобів, ковша екскаватора, ракет, снарядів, елементарних часток при проведенні ядерних досліджень і тому подібне;

- при гідротехнічних, гляціологічних, геологічних, географічних пошуків і дослідженнях;

- при реставрації пам'ятників архітектури, скульптурних монументів, унікальних предметів;

- для фіксації і складання плану дорожньо-транспортної події або місця злочину;

- для визначення по знімках, отриманих в електронному мікроскопі, характеристик мікрорельєфу, наприклад, полірованих поверхонь [35].

Тривимірна модель будівлі, отримана методом фотограмметрії по фотографіям БПЛА представлено на рисунку 2.3.

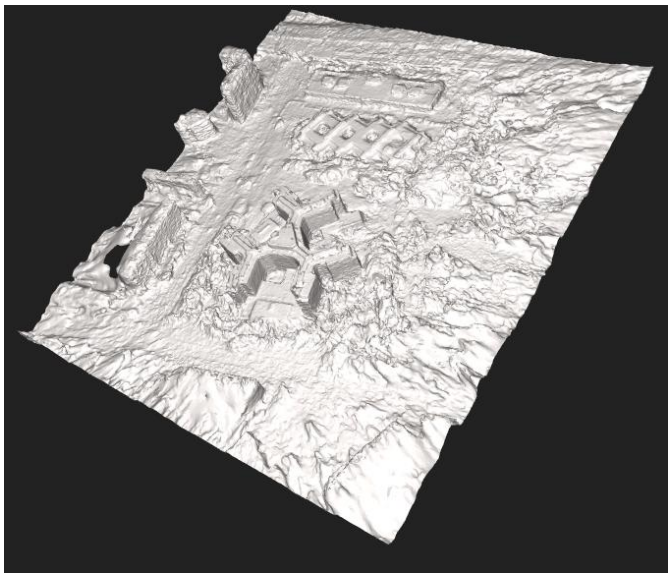

Рисунок 2.3 - Тривимірна модель будівлі, отримана методом фотограмметрії по фотографіям БПЛА

2.3.3.2 Виявлення перешкод

Виявлення перешкод використовується, наприклад, в системах допомоги водієві ADAS (Adnvanced Driver Assistance System), в системах управління безпілотними ЛА та ін.

До числа алгоритмів ADAS входять наступні:

- контроль смуги руху;

- виявлення об'єктів на шляху руху і по сторонах;

- розпізнавання дорожніх об'єктів;

- адаптивний круїз контроль;

- круговий огляд [25].

2.3.3.3 SLAM

SLAM (Simultaneous Localization And Mapping) - метод одночасної навігації і побудови карти, що використовуваний роботами і автономними транспортними засобами для побудови карти в невідомому просторі або для оновлення карти в заздалегідь відомому просторі з одночасним контролем поточного місця розташування і пройденого шляху.

Метод одночасної навігації і побудови карти (SLAM) - це концепція, яка зв'язує два незалежні процеси у безперервний цикл послідовних обчислень, при якому результати одного процесу беруть участь в обчисленнях іншого процесу [36].

Також даний метод використовується для просторової реконструкції (Stereo - SLAM) під час руху транспортних засобів для створення об'ємних карт об'єктів по знімках з однієї або декількох CV- камер.

Приклад просторової реконструкції за допомогою CV- камери представлено на рисунку 2.4

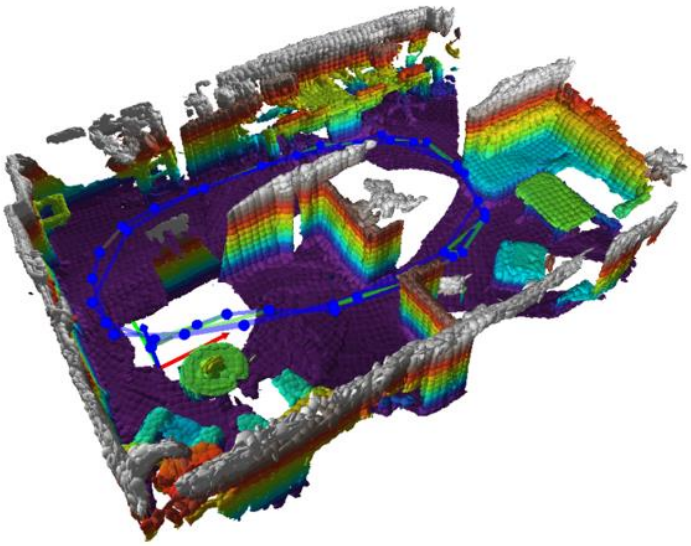

Рисунок 2.4- Приклад просторової реконструкції за допомогою CV-камери

2.3.3.4 Детектування

Детектування об'єктів - це знаходження екземплярів об'єктів на зображенні. Воно передбачає співпостовлення двох і більше зображень при

пошуку зображень унікальних об'єктів, наприклад, архітектурних споруд, скульптур, картин тощо, виявлення на зображеннях класів об'єктів різного ступеня спільності (автомобілів, тварин, меблів, осіб людей тощо, а також їх підкласів), категоризація сцен (місто, ліс, гори, узбережжя тощо) [37].

Програми для детектування об'єктів також досить різноманітні: сортування зображень у домашніх цифрових фотоальбомах, пошук товарів за їхніми зображеннями в інтернет-магазинах, вилучення зображень у геоінформаційних системах, біометрична ідентифікація особистості, цільовий пошук зображень у соціальних мережах і багато іншого.

Розпізнавання такого розмаїття об'єктів і додатків обумовлює необхідність використання методів машинного і глибокого навчання.

Деякі інші приклади застосування методу розпізнавання поза шаблонами: фотограмметрія, виявлення перешкод, одночасна локалізації об'єктів і побудова карти в невідомому просторі (SLAM), дефектоскопія [25].

Приклад детектування у програмі VoTT представлено на рисунку 2.5.

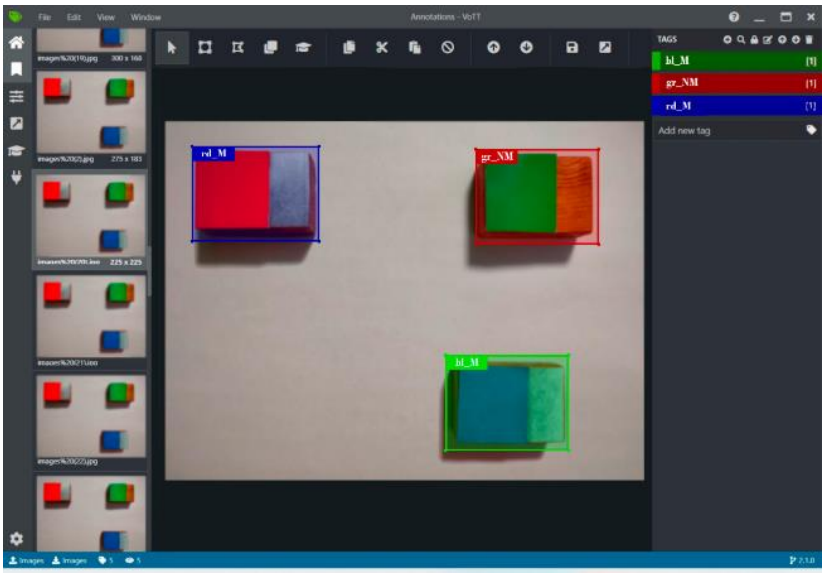

Рисунок 2.5 - Приклад детектування у програмі VoTT

Вибір методики розпізнавання об’єктів відіграє важливу роль у сортуванні, адже, при неправильному визначенні параметрів може порушитись уся технологія сортування. Також, основним параметром являється швидкість обробки зображення.

РОЗДІЛ 3
ВИРІШЕННЯ ПРЯМОГО ТА ІНВЕРСНОГО ЗАВДАННЯ КІНЕМАТИКИ

3.1 Основні поняття

3.1.1 Кінематика точки

Матеріальною точкою називається тіло, що володіє масою, розмірами, формою, обертанням і внутрішньою структурою якою можна знехтувати в умовах досліджуваної задачі. Являється найпростішою фізичною моделлю в механіці [38].

Розташування точки у момент часу t задається радіус-вектором $\bar{r}$, проведеним до цієї точки з початку координат, що зображено на рисунку 3.1.

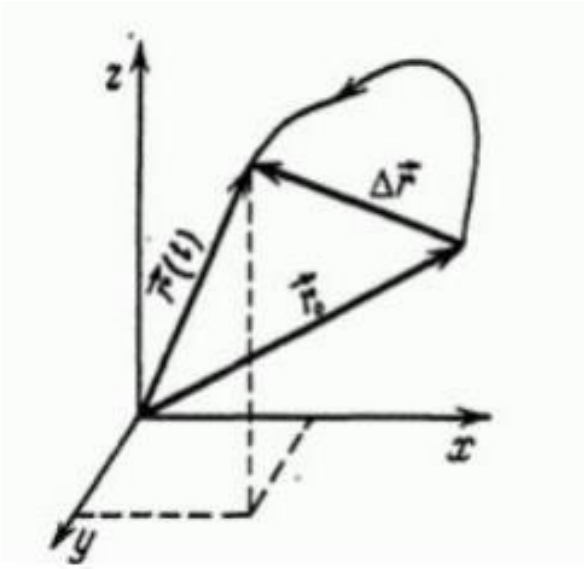

Рисунок 3.1 - Розташування точки у момент часу t, що задається радіус-вектором $\bar{r}$, проведеним до цієї точки з початку координат

Під час руху кінець радіус-вектора описує просторову криву - траєкторію. У прямокутній декартовій системі координат положення радіус-вектора задається трьома його проекціями на осі - координатами x, y, z.

Шлях – це скалярна величина, що являється довжиною траєкторії, невід'ємна та не убуває з часом. Пересуванням точки називається вектор $\Delta\bar{r}$ що

з'єднує початкове положення точки з кінцевим і рівний різниці радіус-векторів в кінцевий і початковий моменти часу [39].

Рух точки відносно вибраної системи відліку вважається заданим, якщо відомий спосіб, за допомогою якого можна визначити положення точки в будь-який момент часу. Встановити рух точки - це означає вказати спосіб, який дозволяє в будь-який момент часу визначити її положення по відношенню до обраної системи відліку [40].

3.1.2 Координатний спосіб завдання руху

Координатним способом завдання руху у вибраній системі координат є координати рухомої точки як функції від часу. Тобто задані три функції від часу (при тривимірному русі): $\begin{cases} x = x(t) \\ y = y(t). \\ z = z(t) \end{cases}$

Ці рівняння є рівняннями траєкторії у параметричній формі. Виключаючи з цих рівнянь параметр t, можна отримати три пари систем двох рівнянь, кожна з яких представляє траєкторію точки, як перетин поверхонь [41].

Нехай існує нерухома прямокутна система координат з центром у нерухомій точці O. При цьому розташування точки M однозначне визначаються її координатами x, y, z. Координатний спосіб завдання руху представлено на рисунку 3.2.

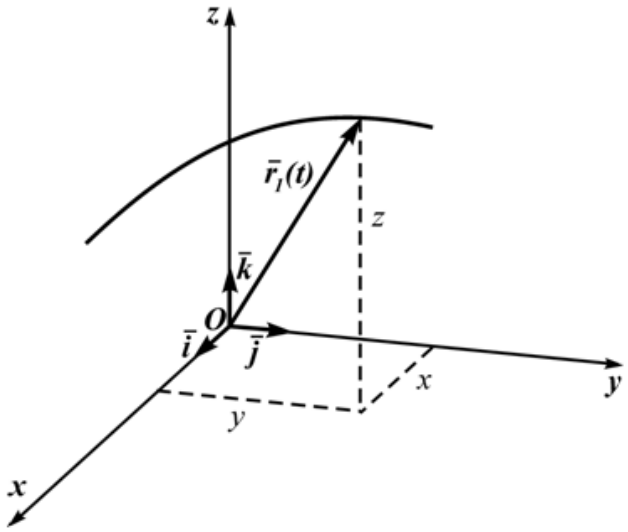

Рисунок 3.2 - Координатний спосіб завдання руху

3.2 Пряме та інверсне завдання кінематики

Для керування рухом кінцевої ланки маніпулятора (захвату) необхідно з певним інтервалом часу вирішувати пряме та інверсне завдання кінематики.

Пряме завдання кінематики - знаходження координат кінця кінематичного ланцюга відносно базової системи координат при заданих довжинах ланок і кутах між ними.

Інверсне завдання кінематики - знаходження кутів між ланками при заданих координатах кінця кінематичного ланцюга і довжинах ланок [42].

На відміну від прямого завдання, основні проблеми при цьому пов'язані з рішенням інверсної. Актуальність вирішення інверсної задачі кінематики (ІЗК) особливо зростає при керуванні маніпуляторами з надлишковою кількістю ланок у режимі малих тимчасових інтервалів. У наразі розроблено безліч методів, здатних вирішити ІЗК для зазначених маніпуляторів [43].

Конфігурація робота з положенням двигунів представлена на рисунку 3.3.

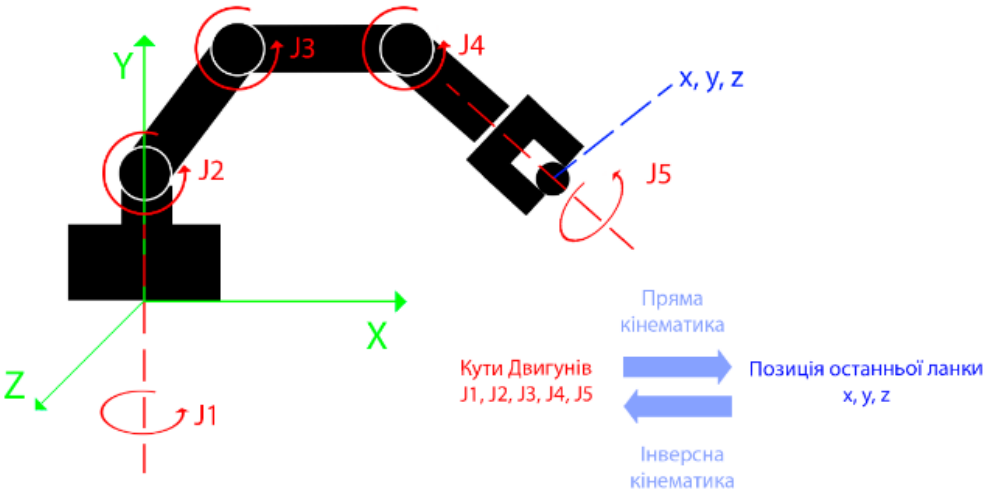

Рисунок 3.3 - Конфігурація робота з положенням двигунів

3.2.1 Аналітичні методи вирішення інверсного завдання кінематики

До переваг аналітичного методу вирішення інверсного завдання кінематики відносять отримання довільної точності рішення. Однак, знаходження точного рішення у вигляді аналітичних залежностей для

узагальнених координат від конструктивних параметрів і заданого вектора положення маніпулятора представляється можливим не для всіх маніпуляторів. Аналітичне рішення, таким чином, існує тільки для роботів з певною конструкцією. Наприклад, осі частини суміжних зчленувань повинні перетинатися в одній точці або повинні бути паралельні або перпендикулярні між собою.

Знаходження узагальнених координат в явному вигляді досить складне завдання, оскільки рівняння є нелінійними. Для спрощення завдання існує ряд методів, призначених для більш простого отримання аналітичних виразів. Метод зворотних перетворень дозволяє вирішувати інверсну задачу кінематики простих маніпуляторів. Суть методу полягає у визначенні кутів поворотів ланок з рівнянь для окремих елементів наявного матричного рівняння [44].

3.2.2 Чисельні методи вирішення інверсного завдання кінематики

У разі якщо вирішення інверсного завдання у вигляді аналітичних виразів неможливо, використовуються численні методи.

Чисельний метод дозволяє вирішити ІЗК для тих конструкцій маніпуляторів, для яких отримання точного рішення в аналітичних вирази не є можливими або досить складними. Це є основною гідністю при використанні чисельного методу стосовно поставленого завдання.

Крім того, численні методи вирішення інверсного завдання дозволяють досягти необхідної точності рішення. Однак час схожості того чи іншого методу заздалегідь невідомо і багато в чому залежить від початкового наближення. У результаті чутливість до кількості змінних цільової функції, виду гіперповерхності, а також різна швидкість схожості та втрати на пошук вимагають значних обчислювальних потужностей, що ускладнює їх використання в реальному часі [45].

3.3 Вирішення прямого завдання кінематики

Положення і орієнтація твердого тіла в просторі однозначно визначається шістьма координатами: трьома лінійними і трьома кутовими. Використання методу, запропонованого в 1955 р. вченими Жаком Денавітом і Річардом Хартенбергом, дозволяє скоротити це число до чотирьох параметрів, званими параметрами Денавіта-Хартенберга:

- Θ_i – кут навколо осі z_{i-1} від x_{i-1} до x_i;
- α_i – кут навколо осі x_i від z_{i-1} до z_i;
- d_i – відстань уздовж осі z_{i-1} від x_{i-1} до x_i;
- a_i – відстань уздовж осі x_i від z_{i-1} до z_i.

Таке спрощення досягається за допомогою стандартизованого алгоритму прив'язки систем координат до ланок маніпулятора. Параметри Денавіта-Хартенберга представлено у таблиці 3.1.

Таблиця 3.1 – Параметри Денавіта-Хартенберга

Плече	Θ_i	α_i	d_i	a_i
1	Θ_1	$-\frac{\pi}{2}$	d_1	a_1
2	Θ_2	0	0	a_2
3	$\Theta_3 - \frac{\pi}{2}$	$\frac{\pi}{2}$	0	0
4	Θ_4	$\frac{\pi}{2}$	d_4	0
5	$\Theta_5 + \pi$	0	d_5	0

Для обертальних ланок параметри α_i, d_i і a_i являються характеристиками ланок, постійними для даного маніпулятора. Однак параметр Θ_i – змінна величина, яка змінюється при русі i-ї ланки відносно (i-1)-ї.

Використовуючи данні таблиці 3.1 будуються матриці однорідного перетворення:

$$J_1 = \begin{vmatrix} cos(\Theta_1) & -sin\,(\Theta_1) \times cos\,(-\frac{\pi}{2}) & sin\,(\Theta_1) \times sin\,(-\frac{\pi}{2}) & a_1 \times cos\,(\Theta_1) \\ sin(\Theta_1) & cos\,(\Theta_1) \times cos\,(-\frac{\pi}{2}) & -cos\,(\Theta_1) \times sin(-\frac{\pi}{2}) & a_1 \times sin\,(\Theta_1) \\ 0 & \sin\,(-\frac{\pi}{2}) & cos(-\frac{\pi}{2}) & d_1 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_2 = \begin{vmatrix} cos(\Theta_2) & -sin\,(\Theta_2) \times cos\,(0) & sin\,(\Theta_2) \times sin\,(0) & a_2 \times cos\,(\Theta_2) \\ sin(\Theta_2) & cos\,(\Theta_2) \times cos\,(0) & -cos\,(\Theta_2) \times sin(0) & a_2 \times sin\,(\Theta_2) \\ 0 & \sin\,(0) & cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_3 = \begin{vmatrix} cos\left(\Theta_3 - \frac{\pi}{2}\right) & -sin\,(\Theta_3 - \frac{\pi}{2}) \times cos\,(\frac{\pi}{2}) & sin\,(\Theta_3 - \frac{\pi}{2}) \times sin\,(\frac{\pi}{2}) & cos\,(\Theta_3 - \frac{\pi}{2}) \\ sin\left(\Theta_3 - \frac{\pi}{2}\right) & cos\,(\Theta_3 - \frac{\pi}{2}) \times cos\,(\frac{\pi}{2}) & -cos\,(\Theta_3 - \frac{\pi}{2}) \times sin(\frac{\pi}{2}) & sin\,(\Theta_3 - \frac{\pi}{2}) \\ 0 & \sin\,(\frac{\pi}{2}) & cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_4 = \begin{vmatrix} cos(\Theta_4) & -sin\,(\Theta_4) \times cos\,(\frac{\pi}{2}) & sin\,(\Theta_4) \times sin\,(\frac{\pi}{2}) & cos\,(\Theta_4) \\ sin(\Theta_4) & cos\,(\Theta_4) \times cos\,(\frac{\pi}{2}) & -cos\,(\Theta_4) \times sin(\frac{\pi}{2}) & sin\,(\Theta_4) \\ 0 & \sin\,(\frac{\pi}{2}) & cos(\frac{\pi}{2}) & d_4 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_5 = \begin{vmatrix} cos(\Theta_5 + \pi) & -sin\,(\Theta_5 + \pi) \times cos\,(0) & sin\,(\Theta_5 + \pi) \times sin\,(0) & cos\,(\Theta_5 + \pi) \\ sin(\Theta_5 + \pi) & cos\,(\Theta_5 + \pi) \times cos\,(0) & -cos\,(\Theta_5 + \pi) \times sin(0) & sin\,(\Theta_5 + \pi) \\ 0 & \sin\,(0) & cos(0) & d_5 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Для отримання координат кожної ланки відносно початкової системи координат, будуються матриці, які являються добутком матриць однорідного перетворення, формули (3.2), … (3.7). Для першого та останнього добутку застосовується тотожне перетворення (яке не призводить до зміни об'єкта), що здійснюється одиничною матрицею (матрицею тотожного перетворення), формула (3.1).

$$T = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (3.1)$$

$$R_1 = T \times {J_1}^T \quad (3.2)$$

$$R_2 = J_1 \times {R_1}^T \quad (3.3)$$

$$R_3 = J_2 \times {R_2}^T \quad (3.4)$$

$$R_4 = J_3 \times {R_3}^T \quad (3.5)$$

$$R_5 = J_4 \times {R_4}^T \quad (3.6)$$

$$R_6 = R_5 \times T^T \quad (3.7)$$

Для знаходження положення останньої ланки маніпулятора, відносно початкової системи координат використовуються значення отриманої матриці з формули (3.7). Далі приведені формули з параметрами положення X, Y, Z:

$$X = R_6[1][4]$$
$$Y = R_6[2][4]$$
$$Z = R_6[3][4]$$

Щоб визначити напрямок, в якому повернута остання ланка маніпулятора, відносно початкової системи координат, необхідно виконати розрахунок RotX, RotY та RotZ, що представлені далі:

$$RotX(\psi) = \Psi = arctan2(\frac{R_6[1][3]}{RotY(\phi)}, \frac{R_6[2][3]}{RotY(\phi)}$$

$$RotY(\phi) = \phi = arctan2(-R_6[1][3], \sqrt{R_6[1][1]^2 + R_6[1][2]^2}$$

$$RotZ(\theta) = \theta = arctan2(\frac{R_6[1][2]}{RotY(\phi)}, \frac{R_6[1][1]}{RotY(\phi)}$$

3.4 Вирішення інверсного завдання кінематики

Для розрахунку ІК можливе використання одного з трьох методів: числовий, закритий або геометричний (аналітичний). Геометричний метод має пріоритет через його відносну простоту розрахунків для обладнання. Він полягає в знаходженні аналітичних виразів у явному вигляді з використанням апарату тригонометричних функцій з урахуванням кінематичної схеми маніпулятора.

Для визначення кутів, необхідних для досягнення бажаної точки, необхідно вивести рівняння для кожної ланки .Для цього необхідно побудувати матрицю, використовуючи метод Ейлера:

$$R_{0\_T} = \begin{vmatrix} R_{0\_T_1} & R_{0\_T_2} & R_{0\_T_3} & R_{0\_T_4} \\ R_{0\_T_5} & R_{0\_T_6} & R_{0\_T_7} & R_{0\_T_8} \\ R_{0\_T_9} & R_{0\_T_{10}} & R_{0\_T_{11}} & R_{0\_T_{12}} \\ R_{0\_T_{13}} & R_{0\_T_{14}} & R_{0\_T_{15}} & R_{0\_T_{16}} \end{vmatrix}$$

Значення $R_{0\_T_1} \ldots R_{0\_T_{16}}$ представлено у формулах (3.8),…(3.20):

$$R_{0\_T_1} = \cos(RotZ) \times \cos(RotX) - \cos(RotY) \times \sin(RotZ) \times \times \sin(RotX), \quad (3.8)$$

$$R_{0\_T_2} = \cos(RotY) \times \sin(\mathrm{RotZ}) + \cos(RotZ) \times \cos(RotY) \times \times \sin(RotX), \quad (3.9)$$

$$R_{0\_T_3} = \sin(RotY) \times \sin(RotX), \quad (3.10)$$

$$R_{0\_T_4} = X, \quad (3.11)$$

$$R_{0\_T_5} = \cos(RotY) \times \cos(RotX) \times \sin(RotZ) + \cos(RotZ) \times \sin(RotX), \quad (3.12)$$

$$R_{0\_T_6} = \cos(RotZ) \times \cos(RotY) \times \cos(RotX) - \sin(RotZ) \times \sin(RotX), \quad (3.13)$$

$$R_{0\_T_7} = \cos(RotX) \times \sin(RotY), \quad (3.14)$$

$$R_{0\_T_8} = Y, \tag{3.15}$$

$$R_{0\_T_9} = \sin(RotZ) \times \sin(RotY), \tag{3.16}$$

$$R_{0\_T_{10}} = \cos(RotZ) \times \sin(RotY), \tag{3.17}$$

$$R_{0\_T_{11}} = -\cos(RotY), \tag{3.18}$$

$$R_{0\_T_{12}} = Z, \tag{3.19}$$

$$R_{0\_T_{13}} = 0,\ R_{0\_T_{14}} = 0, R_{0\_T_{15}} = 0, R_{0\_T_{16}} = 1, \tag{3.20}$$

Далі застосовується тотожне перетворення та знаходження матриці положення перших трьох ланок (3.21):

$$RT_{0\_T} = R_{0\_T} \times T$$

$$Rem_{0\_3} = \begin{vmatrix} \cos(180) & \sin(180) & 0 & 0 \\ -\sin(180) \times \cos(\alpha_5) & \cos(180) \times \cos(\alpha_5) & \sin(\alpha_5) & 0 \\ \sin(180) \times \sin(\alpha_5) & -\cos(180) \times \sin(\alpha_5) & \cos(\alpha_5) & -d_5 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$R_{0\_3} = RT_{0\_T} \times Rem_{0\_3}, \tag{3.21}$$

З отриманих результатів розраховується значення кута першого двигуна, по формулі:

$$J_1 = \operatorname{atan}\left(\frac{R_{0\_3}[1][3]}{R_{0\_3}[0][3]}\right)$$

Однак, значення J1 може приймати чотири різних параметрів, в залежності від положення решти ланок. Для виключення невизначеності виконується перевірка:

- якщо виконується наступна тотожність: X>0 та Y>0, то значення J_1 остається незмінною, відповідно до формули;

- якщо виконується наступна тотожність: X>0 та Y>0, то значення J_1 остається незмінною, відповідно до формули;

- якщо виконується наступна тотожність: X<0 та Y<0, то від значення J_1 вираховується 180;

- якщо виконується наступна тотожність: X<0 та Y>0, то до значення J_1 додається 180.

Для знаходження значень J2 та J3 потрібно прибрати невизначеність, адже для досягнення необхідної позиції, ланки маніпулятора можуть прийняти два положення (FWD та MID). Розрахунок градусів двигунів J2 та J3 представлено у таблиці 3.2.

Таблиця 3.2 - Розрахунок градусів двигунів J2 та J3

	FWD	MID
pX	$\sqrt{\lvert R_{0_3}[1][3]\rvert^2 + \lvert R_{0\_4}[0][3]\rvert^2}$	$\sqrt{\lvert R_{0_3}[1][3]\rvert^2 + \lvert R_{0\_4}[0][3]\rvert^2}$
pY	$R_{0\_3}[2][3] - d_1$	$R_{0\_3}[2][3] - d_1$
pX-a1	$px - a1$	$-(px - a1)$
pa2H	$\sqrt{pY^2 + (pX - a1)^2}$	$\sqrt{pY^2 + (pX - a1)^2}$
pa3H	$\sqrt{{d_4}^2 + a_3^2}$	$\sqrt{{d_4}^2 + a_3^2}$
ΘA	$\frac{\pi}{180} \times \text{atan}\left(\frac{pY}{pX - a1}\right)$	$\frac{\pi}{180} \times$ $\times \text{acos}\left(\frac{a_2^2 + pa2H^2 - d_4^2}{2 \times a_2 \times pa2H}\right)$
ΘB	$\frac{\pi}{180} \times \text{acos}\left(\frac{\lvert pa3H\rvert^2 + a_2^2 - pa2H^2}{2 \times \lvert pa3H\rvert \times a_2}\right)$	$\frac{\pi}{180} \times \text{atan}\left(\frac{px - a1}{pY}\right.$
ΘC	$180 - \frac{\pi}{180} \times$ $\times \text{acos}\left(\frac{\lvert pa3H\rvert^2 + a_2^2}{2 \times \lvert pa3H\rvert \times a_2}\right)$	$180 - \frac{\pi}{180} \times$ $\times \text{acos}\left(\frac{\lvert pa3H\rvert^2 + a_2^2}{2 \times \lvert pa3H\rvert \times a_2}\right)$
ΘD	-	$90 - (\Theta A + \Theta B)$
ΘE	90	90

Продовження таблиці 3.2

J2 ∠	$-(\Theta A + \Theta B)$	$-180 + \Theta D$
J3 ∠	ΘC	ΘC

Опираючись на таблицю 3.2 визначено, що в залежності від знаку, який приймає pX-a1, значення J2 та J3 вираховується згідно до колонки FMD та MID (позитивне та негативне відповідно).

Значення двох останніх двигунів J4 та J5 визначаються аналогічно з матрицями однорідного перетворення з попереднього пункту розділу, однак, замість параметрів Θ_i використовуються значення J1, J2 та J3:

$$J_{i\_1} = \begin{vmatrix} Jr1 & -sin\,(Jr1) \times \cos\,(\alpha_1) & sin\,(Jr1) \times sin\,(\ \alpha_1) & a_1 \times cos\,(Jr1) \\ sin(Jr1) & cos\,(Jr1) \times cos\,(\alpha_1) & -cos\,(Jr1) \times sin(\alpha_1 & a_1 \times sin\,(Jr1) \\ 0 & \sin\,(\alpha_1) & cos(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_{i\_2} = \begin{vmatrix} cos(Jr2) & -sin\,(Jr2) \times cos\,(\alpha_2) & sin\,(Jr2) \times sin\,(\alpha_2) & a_2 \times cos\,(Jr2) \\ sin(Jr2) & cos\,(Jr2) \times cos\,(\alpha_2) & -cos\,(Jr2) \times sin(\alpha_2) & a_2 \times sin\,(Jr2) \\ 0 & \sin\,(\alpha_2) & cos(\alpha_2) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

$$J_{i\_3} = \begin{vmatrix} cos(Jr3) & -sin\,(Jr3) \times cos\,(\alpha_3) & sin\,(Jr3) \times sin\,(\ \alpha_3) & cos\,(Jr3) \\ sin(Jr3) & cos\,(Jr3) \times cos\,(\alpha_3) & -cos\,(Jr3) \times sin(\alpha_3) & sin\,(Jr3) \\ 0 & \sin\,(\alpha_3) & cos(\alpha_3) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix},$$

де $Jr1 = J1 \times 0{,}0174533;$

$Jr2 = J2 \times 0{,}0174533;$

$Jr3 = J3 \times 0{,}0174533.$

Відповідно проводиться добуток отриманих матриць з тотожним перетворенням у першому рівнянні:

$$R_{i\_1} = T \times {J_{i\_1}}^T$$

$$R_{i\_2} = J_{i\_2} \times {R_{i\_1}}^T$$

$$R_{i\_3} = J_3 \times {R_{i\_2}}^T$$

Після чого виконується транспонування матриці R_{i_3}, однак з виключенням останніх ряду та стовпця, а також її добуток з матрицею (3.21):

$$R_{3\_5} = {R_{i\_3}}^T \times R_{0\_3}$$

З отриманої матриці вираховується значення градусів двигунів J4 та J5:

$$J4 = atan2(R_{3\_5}[2][2], \sqrt{1 - R_{3\_5}[2][2]^2})$$

Обчислення J5 відбувається опираючись на J4:

Якщо J4>0 та рівняння $atan2(-R_{3\_5}[2][1], R_{3\_5}[2][0]) > 0$ задовольняють нерівність, тоді:

$$J5 = atan2(-R_{3\_5}[2][1], R_{3\_5}[2][0])$$

Якщо J4>0 та рівняння $atan2(-R_{3\_5}[2][1], R_{3\_5}[2][0]) > 0$ не задовольняють нерівність, тоді:

$$J5 = atan2(R_{3\_5}[2][1], -R_{3\_5}[2][0])$$

Виконавши розрахунки прямої та інверсної завдань кінематики були отримані параметри, які дозволяють виконувати, як ручне, так і автоматичне керування маніпулятором роботизованого комплексу «Orion-I». Для перевірки результатів достатньо співставити результати прямої та інверсної завдань кінематики. Адже вхідні параметри в прямій кінематиці являються вихідними в інверсній і навпаки.

РОЗДІЛ 4
ПРАКТИЧНА РЕАЛІЗАЦІЯ ПОЛІПШЕНЬ ЛАБОРАТОРНОГО СТЕНДУ «ORION-I»

4.1 Структурна схема лабораторного стенду

Для визначення основних функціональних частин лабораторного стенду «Orion-I», для подальшого його вдосконалення, була побудована структурна схема, що зображено на рисунку 4.1.

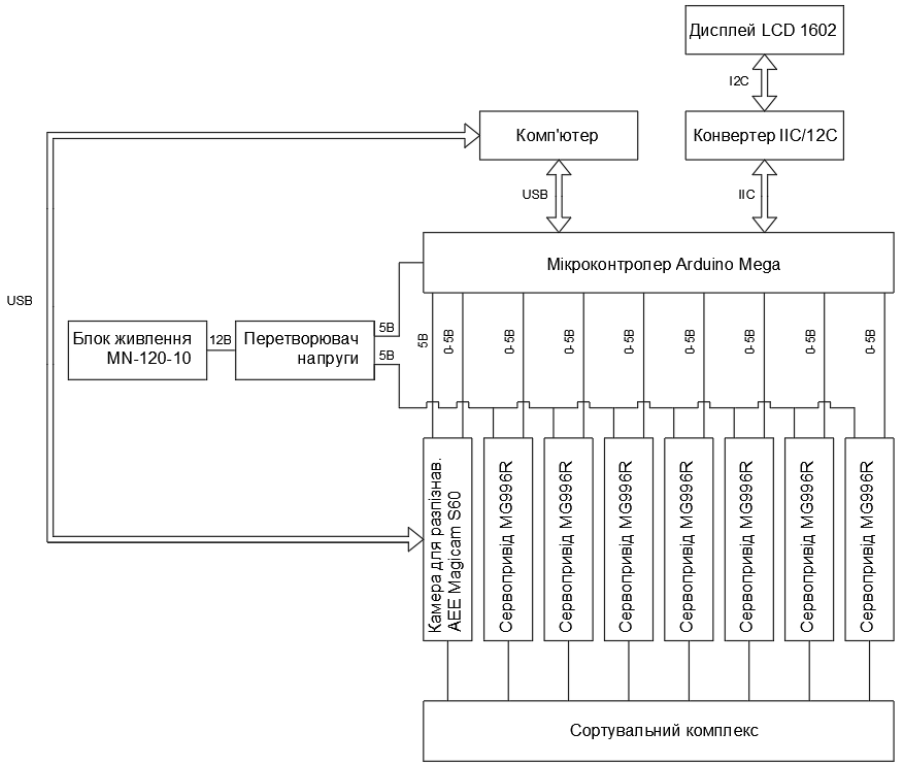

Рисунок 4.1 – Структурна схема функціональних частин лабораторного стенду «Orion-I»

4.2 Розпізнавання об’єктів на базі YOLOv3 та Darknet

Для вдосконалення зчитування параметрів об’єктів сортування, виходячи з дослідів роботи лабораторного стенду «Orion-I» та дослідження принципів розпізнавання за допомогою машинного зору, з розділів один та два

відповідно, було обрано систему виявлення об'єктів YOLOv3 на базі нейронної мережі DarkNet.

YOLOv3-це вдосконалена версія архітектури YOLO. Вона складається з 106-ти згорткових шарів і краще детектує невеликі об'єкти в порівнянні з її попередницею YOLOv2. Основна особливість YOLOv3 полягає в тому, що на виході є три шари кожен з яких розрахований на виявлення об'єктів різного розміру.

DarkNet-це фреймворк нейронної мережі з відкритим вихідним кодом, написаний на мовах C та CUDA.

В якості пристрою для розпізнавання об’єктів була обрана відеокамера AEE S60 MagiCam, яка має наступні характеристики:

- максимальна роздільна здатність відеозйомки: 1920x1080;
- тип матриці: CMOS;
- матриця: 16 Мп;
- максимальна частота кадрів: 60 кадрів/с.

Відеокамера AEE S60 MagiCam представлена на рисунку 4.2.

Рисунок 4.2 - Відеокамера AEE S60 MagiCam

4.2.1 Тренування нейронної мережі YOLOv3

Для знаходження об’єктів, на відеозображенні, YOLOv3 використовує конфігураційні файли з натренованими моделями. Однак, для відстеження об’єктів, які відсутні в базі даних необхідно натренувати власну модель.

Для навчання YOLOv3 було використано операційну систему Ubuntu.

Ubuntu 18.04 - це операційна система, яка розроблюється спільнотою, що заснована на ядрі Linux. Вона ідеально підходить для використання на персональних комп'ютерах, ноутбуках і серверах. В ній містяться всі необхідні компоненти, які потрібні для тренування нейромережі YOLOv3.

Оболонка операційної системи Ubuntu 18.04 представлено на рисунку 4.3

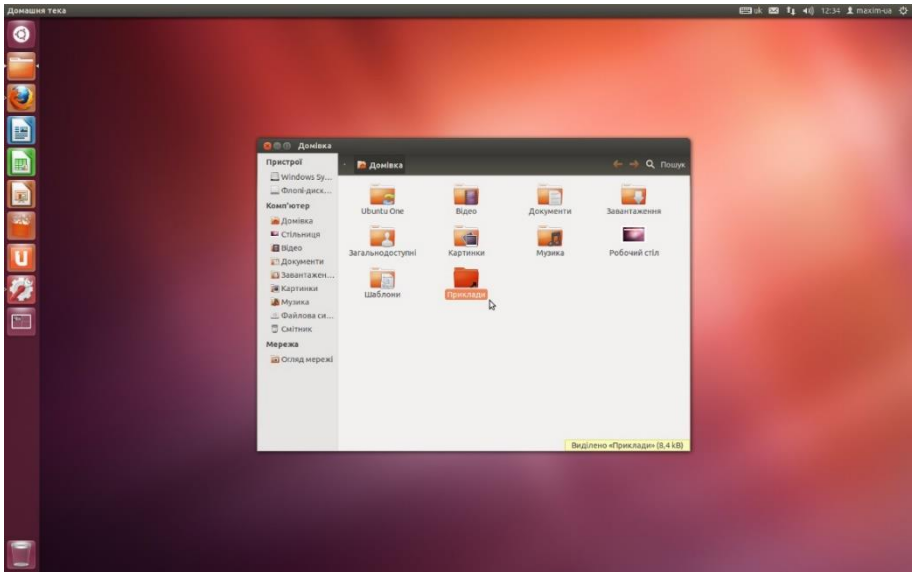

Рисунок 4.3 - Оболонка операційної системи Ubuntu 18.04

Досліджуваний лабораторний стенд «Orion-I» виконує сортування дерев’яних брусків з металевими та кольоровими вставками. Для занесення даних об’єкту сортування було виконано ручне тегування за допомогою Visual Object Tagging Tool.

Цей інструмент дозволяє занести всю інформацію про положення об’єкту на фотографії, для подальшої її використання у навчанні нейронної мережі YOLOv3.

Щоб отримати бажаний результат застосовано близько ста фотографій необхідного об’єкту. Для спрощення отримання фотографій, було знято відео дерев’яних брусків з різних ракурсів, після чого, за допомогою програми Visual Studio, конвертовано відеофайл у набір зображень формату .jpg.

Ручне тегування в Visual Object Tagging Tool представлено на рисунку 4.4.

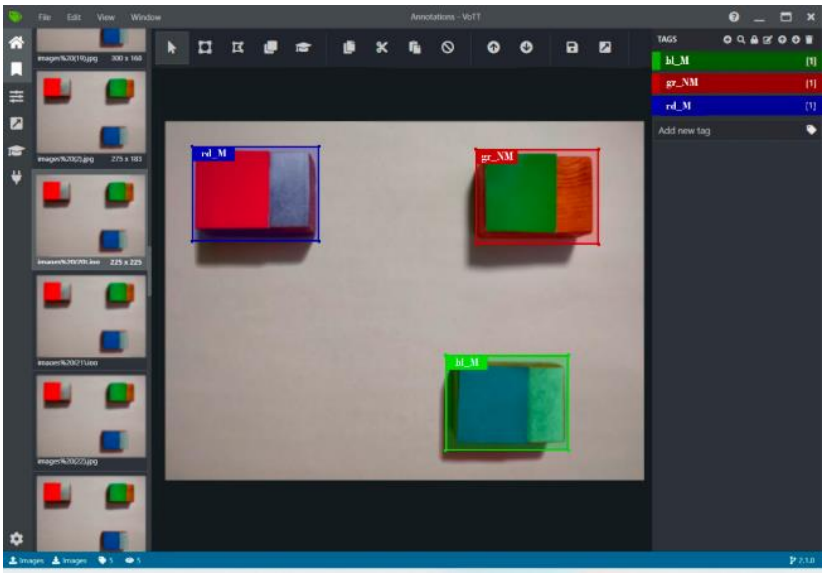

Рисунок 4.4 - Ручне тегування в Visual Object Tagging Tool

Завершавши тегування, отриманий файл з координатами виділених об’єктів та їх найменуваннями конвертується в формат YOLOv3.

Для початку навчання отриманої моделі використовувався файл «pre-trained dark-net weights», також попередньо конвертований в формат YOLOv3.

Перед початком тренування необхідно редагувати параметри конфігураційних файлів: занести шукані об’єкті в базу, підібрати кількість ітерацій для кожної фотографії та виконати підрахунок необхідного часу для навчання. Тренування нейронної мережі відбувається декілька годин, якщо застосовувати CUDA ядра і правильно підібрати параметри конфігураційних файлів.

CUDA - це абревіатура технології запатентованої Nvidia, означає Compute Unified Device Architecture (обчислювальна уніфікована Архітектура пристрою). Тобто це архітектура - така форма організації внутрішнього устрою ядер в відеокарті. Мета цієї технології - ефективні паралельні (одночасні) обчислення.

Одне ядро CUDA аналогічно процесорному, з тією лише різницею, що воно простіше за своєю структурою, проте їх кількість дуже велика. Типовий процесор має від 2 до 16 ядер, в той час, як CUDA у відеокарті - сотні, навіть

у бюджетних відеокартах Nvidia. А високопродуктивні рішення і зовсім налічують тисячі [50].

Після закінчення тренування нейронної мережі YOLOv3 необхідно підключити отримані файли, замість «pre-trained dark-net weights».

Результати розпізнавання, за допомогою нейронної мережі YOLOv3 представлено на рисунку 4.5.

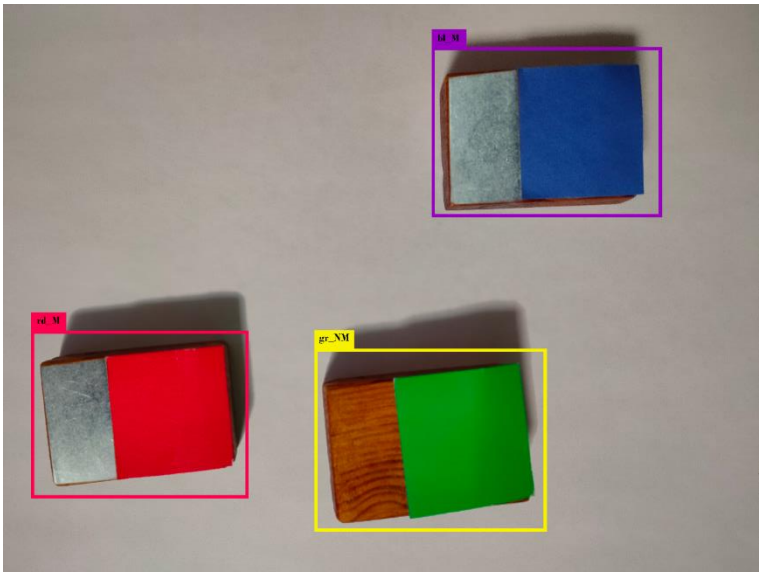

Рисунок 4.5 - Результати розпізнавання, за допомогою нейронної мережі YOLOv3

Для визначення координат об’єктів було змінено файл image.c в репозиторії папки darknet. Для цього, у файл було додано програмний код, який виводить інформації об обмежувальній рамки.

Частина коду зміненого файлу image.c представлено у додатку А.

4.3 Побудова програмного коду на базі розрахунків кінематики

Для розрахунку переміщення маніпулятора лабораторного стенду «Orion-I» була розроблена програма на основі прямої та інверсної завдань кінематики, з розділу 3. Програма виконується на комп’ютері, адже контролер Arduino Mega не в змозі впоратися з розрахунками кінематики. Програму написано на мові Python у програмі Visual Studio Code.

Python— високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника і читаності коду. Синтаксис ядра Python мінімалістичний. У той же час стандартна бібліотека включає великий набір корисних функцій.

Python підтримує структурне, узагальнене, об'єктно-орієнтоване, функціональне та аспектно-орієнтоване програмування. Основні архітектурні риси-динамічна типізація, автоматичне управління пам'яттю, повна інтроспекція, механізм обробки винятків, підтримка багатопотокових обчислень, високорівневі структури даних. Підтримується розбиття програм на модулі, які, в свою чергу, можуть об'єднуватися в пакети [51].

Visual Studio Code-редактор вихідного коду, розроблений Microsoft для Windows, Linux і macOS. Позиціонується як «легкий» редактор коду для кроссплатформенной розробки веб додатків. Включає в себе відладчик, інструменти для роботи з Git, підсвічування синтаксису, IntelliSense та засоби для рефакторингу. Має широкі можливості для кастомізації: призначені для користувача теми, поєднання клавіш і файли конфігурації. Поширюється безкоштовно, розробляється як програмне забезпечення з відкритим вихідним кодом, але готові збірки поширюються за пропрієтарною ліцензією [52].

Інтерфейс програми Visual Studio Code представлено на рисунку 4.6.

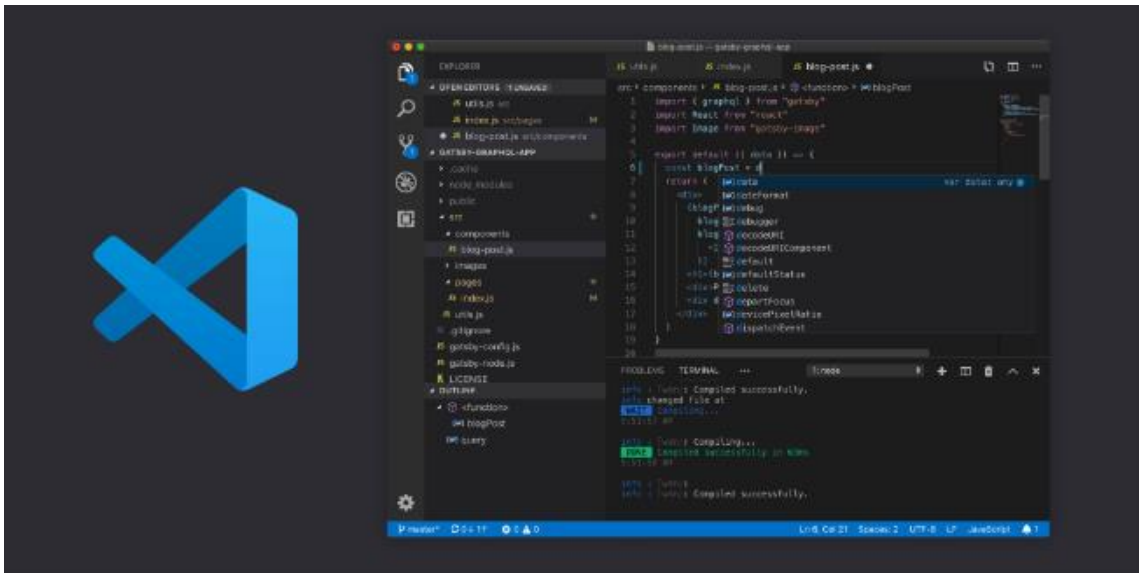

Рисунок 4.6 - Інтерфейс програми Visual Studio Code

Програмний код для розрахунку інверсної кінематики на мові Python представлено у додатку Б.

4.4 Створення програми для мікроконтролера

Так як усі розрахунки кінематики та розпізнавання об’єктів виконується на комп’ютері, на контролер необхідно лише передавати значення положення серводвигунів.

Для програмування мікроконтролера Arduino Mega використовується програмний пакет Arduino IDE.

Arduino IDE – це додаток, призначений для створення, настройки, тестування та обслуговування програмного забезпечення.

Інтегрована середа розробки характеризується наявністю складної функціональності, включаючи редагування та компіляцію вихідного коду, створення програмних ресурсів, створення баз даних і т.д. [53].

Інтерфейс програми Arduino IDE представлено на рисунку 4.7.

Рисунок 4.7 - Інтерфейс програми Arduino IDE

Передача даних відбувається за допомогою USB порту по протоколу UART зі швидкістю 115200 бод.

Для передачі усієї інформації про положення двигунів, данні було зашифровано в єдине повідомлення, формату «"J1" + J1Angle + "J2" + J2Angle + "J3" + J3Angle + "J4" + J4Angle + "J5" + J5Angle + "\n"».

Програмний код для керування серводвигунами на мові C++ представлено у додатку В.

4.5 Створення інтерфейсу оператора

Для створення інтерфейсу оператора лабораторного стенду «Orion-I» було використано Qt Designer.

Qt Designer – кроссплатформлене вільне середовище для розробки графічних інтерфейсів (GUI) програм використовують бібліотеку Qt.

Він дозволяє створювати графічні інтерфейси користувача за допомогою ряду інструментів. Існує панель інструментів «панель віджетів», в якій доступні для використання елементи інтерфейсу — віджети, такі як, наприклад, «випадаючий список» ComboBox, «поле введення» LineEdit, «кнопка» PushButton і багато інших. Кожен віджет має свій набір властивостей, який визначається відповідним йому класом бібліотеки Qt. Властивості віджета можуть бути змінені за допомогою «Редактора властивостей». Для кожного класу властивостей віджета існує свій спеціалізований редактор. Характерною особливістю Qt Designer є підтримка візуального редагування сигналів і слотів. Так, наприклад, можна пов'язати сигнал, що генерується по переключенню стану віджета CheckBox зі слотом відповідає за доступність іншого віджета [54].

Інтерфейс програми Qt Designer представлено на рисунку 4.8.

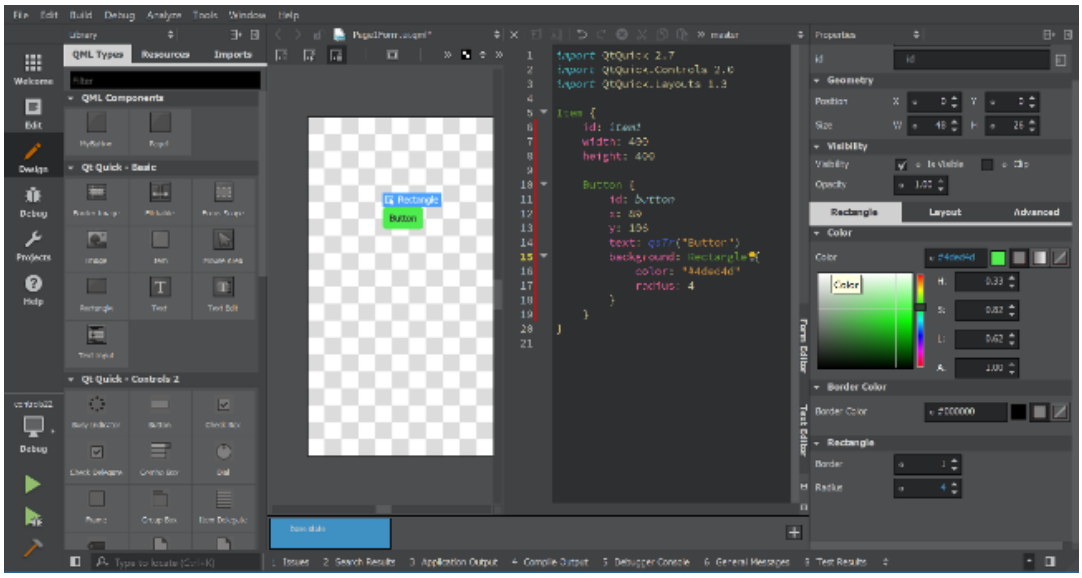

Рисунок 4.8 - Інтерфейс програми Qt Designer

Інтерфейс оператора дозволяє отримувати данні с камери, поточні координати останньої ланки маніпулятора та кут повороту двигунів.

В наявності два режими управління: ручний (Manual control) та автоматичний (Auto Control).

В автоматичному режимі програма лабораторного стенду відбувається автоматично.

В ручному режимі можливе керування як усіма двигунами одночасно, використовуючи пульт управління у вигляді двох хрестовин, застосовуючи інверсну кінематику, так і задавати положення окремих двигунів для налагодження.

За для підвищення безпеки, були прописані обмеження, які захищають від випадкових натискань. При виконанні маніпулятором програми в автоматичному режимі, усі кнопки управління блокуються та загораються червоні індикатори. В цей час неможливо відразу ж перейти до ручного режиму, спочатку необхідно зупинити виконання програми, після чого розблокується відповідна кнопка переходу. У ручному режимі виконання програми запустити неможливо, для цього необхідно змінити режим та натиснути кнопку «Start».

Код інтерфейсу програми управління лабораторним стендом «Orion-I» представлено у додатку Г.

Інтерфейс програми управління лабораторним стендом «Orion-I» у ручному режимі представлено на рисунку 4.9.

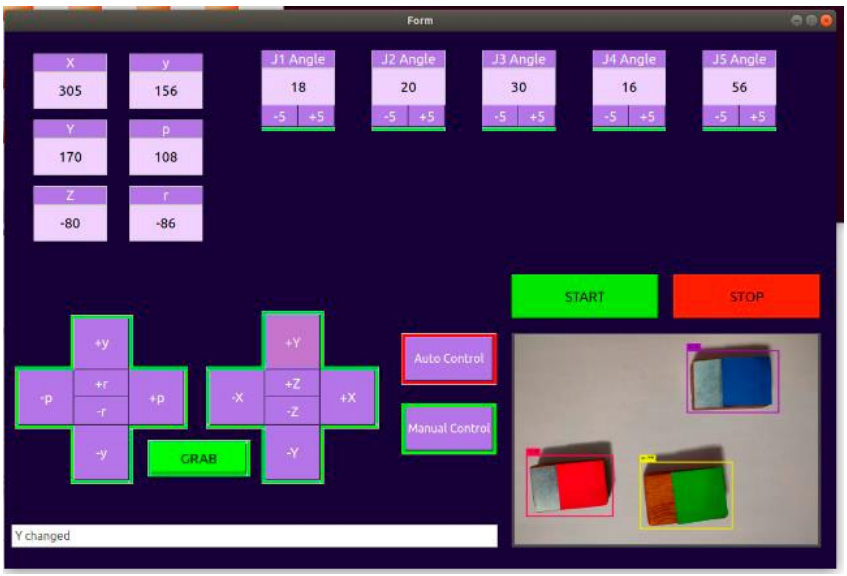

Рисунок 4.9 - Інтерфейс програми управління лабораторним стендом «Orion-I» у ручному режимі

Інтерфейс програми управління лабораторним стендом «Orion-I» у автоматичному режимі представлено на рисунку 4.10.

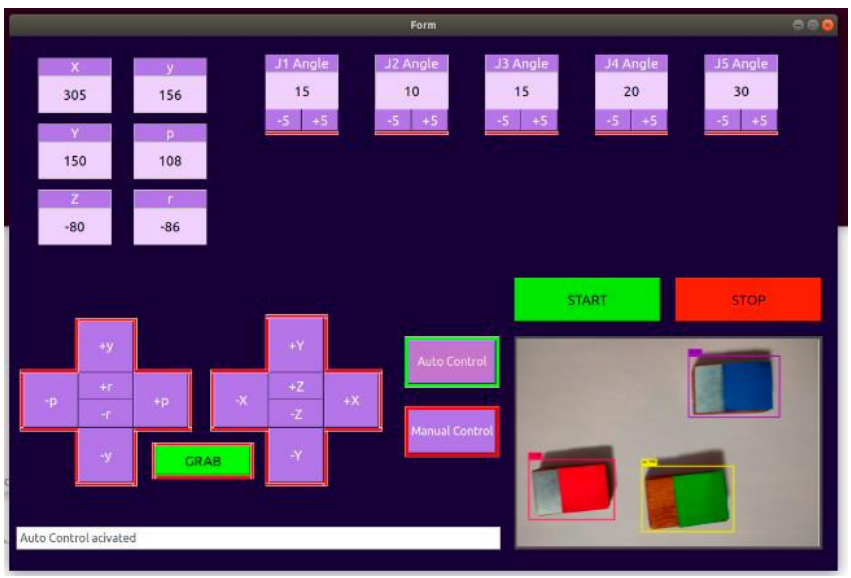

Рисунок 4.10 - Інтерфейс програми управління лабораторним стендом «Orion-I» у ручному режимі

Створення усіх компонентів програмного комплексу відбувалося у різних середовищах, з використанням різноманітних інструментів, що дозволило не обмежувати процес розробки конкретними методами або мовами програмування. У роботі використовувались такі мови, як: Python, C++ та CSS.

ВИСНОВКИ

На базі літературних джерел були розглянуті особливості функціювання лабораторного стенду по дослідженню промислових роботів – маніпуляторів. Виконано аналіз існуючих установок на підприємствах та принципів сортування об’єктів в таких сферах, як: переробна, харчова промисловості, сфера послуг, медицина та радіо-електронна промисловість. Також, виконано аналіз існуючих лабораторних стендів.

Було проведено дослідження роботи роботизованого комплексу для лабораторного практикуму «Orion-I» та виявлені недоліки, пов’язані з роботою первинних перетворювачів, а саме: вплив оточуючого середовища на параметри об’єктів сортування (температура та освітлення приміщення), а також обмеження, пов’язані з позиціюванням об’єктів.

Для збільшення спектру завдань, які може виконувати роботизований комплекс «Orion-I» виконано дослідження методів розпізнавання за допомогою відеокамери, та виділено такі основні вимоги до системи розпізнавання, як: швидкість обробки зображення та точність виявлення об’єктів.

Розраховані пряма та інверсна завдання кінематики для виявлення необхідного обороту двигунів, що дозволяє застосовувати інформацію, отриману від системи розпізнавання (координати x, y, z) для досягнення об’єкту сортування.

Для визначення основних функціональних частин лабораторного стенду «Orion-I», для подальшого його вдосконалення, була побудована структурна схема. Для вдосконалення роботизованого комплексу проведена заміна первинних перетворювачів: датчики кольору, положення та індуктивний датчик, на камеру AEE S60 MagiCam. Для використання якої виконано навчання нейронної мережі YOLOv3 на базі фреймворку Darknet, для розпізнавання об’єктів сортування на відеозображенні.

Також, у розділі практична реалізація поліпшень лабораторного стенду «Orion-I», було виконано описання написаної програми, яка виконує розрахунки прямої та інверсної кінематики на мові Python, для подальшого зв’язку з нейронною мережею YOLOv3.

Усі розрахунки кутів повороту двигунів та розпізнавання об’єктів виконується на комп’ютері, адже контролер Arduino Mega не призначено для таких операцій. Данні кутів серводвигунів, по USB, передаються з комп’ютера до контролеру у вигляді одного закодованого повідомлення. Де воно обробляється згідно з програмою, закладеною у Arduino Mega, написаною на Сі подібній мові та перетворюється у сигнал, який надходить до серводвигунів.

Уся інформація, щодо положення останньої ланки маніпулятора, данні кутів двигунів та кнопки для ручного управління маніпулятором, зображені на інтерфейсі оператора, розробленого за допомогою програмного пакету Qt Designer. Також, на екран, виводиться відеопотік з камери, попередньо оброблений нейромережею YOLOv3.

Застосовані технології дозволяють видалити недоліки, які виникли через недосконаліть первинних перетворювачів, а також збільшити спектр завдань, які може виконувати роботизований комплекс для лабораторного практикуму «Orion-I».

ПЕРЕЛІК ПОСИЛАНЬ

1. Котанович М. Міжнародна федерація робототехніки представила топ-10 найбільш роботизованих країн світу. URL: http:// robotforum.ru/novosti-texnogologij/mezhdunarodnaya-federacziya-robototexniki-predstavila-top-10-samyix-robotizirovannyix-stran-mira.html (дата звернення: 18.11.2020).

2. Борисов Д.Л., Фількін Т.Г., Норотаєв В.Н. Технології оптичного сортування відходів: аналіз факторів, які впливають на ефективність повітряної сепарації. Пермський національний дослідницький політехнічний університет. Перьм. 2014. С. 47.

3. Yuralovely. Автоматичне сортування пошти в Японії. URL: https://www.youtube.com/watch?v=3be2RThB2hg (дата звернення: 18.11.2020).

4. Робот Clarke зі штучним інтелектом допомагає сортувати сміття. URL: https://robo-hunter.com/news/robot-clarke-s-iskusstvennim-intellektom-pomogaet-sortirovat-musor7327 (дата звернення: 18.11.2020).

5. Handsome_robot. Робот Clarke займеться ресайклінгом. URL: http:// robotrends.ru/pub/1731/robot-clarke-zaymetsya-resayklingom (дата звернення: 18.11.2020).

6. Промисловий робот ABB FlexPicker 360. URL: https://www.robot96.ru/catalog/promyshlennye-roboty/roboty_abb/abb-flexpicker-360/ (дата звернення: 19.11.2020).

7. Прибирання та сортування сміття. Зарубіжні системи роботизованого сортування сміття та відходів. URL: http://robotrends.ru/robopedia/1711-uborka-i-sortirovka-musora. (дата звернення: 19.11.2020).

8. Фінський робот сортує всі види твердого сміття. URL: https://yandex.ru/turbo/hightech.fm/s/2016/07/29/zen-robotics (дата звернення: 19.11.2020).

9. Установка з вивчення роботизованих систем на базі робота-маніпулятора «Optima» 1.01. URL: https://zarnitza.ru/catalog/mekhatronika-i-robototekhnika/robototehnika/vysshie-uchebnye-zavedeniya/ustanovka-po-

izucheniiu-robotizirovannykh-sistem-na-baze-robota-manipuliatora-robotekh-101/ (дата звернення: 20.11.2020).

10. PRO-1.7.1 Типовий комплект навчального обладнання «Засоби автоматизації та управління робота маніпулятора», виконання настільне з ноутбуком, САУ-РОБОТ-НН. URL: http://flagmanpro.ru/cats/sredstva_ avtomatizacii_i_upravleniya_robota-manipulyatora.html (дата звернення: 20.11.2020).

11. Компанія «Новий стиль» Для вузів, технікумів та ПУ. URL: HTTP://NEWSTYLE-Y.RU/HIGH-SCHOOL/ (дата звернення: 21.11.2020).

12. Виставка робо технічних пристроїв. Політехнічний музей. URL: http://www.railab.ru/m-excib.html (дата звернення: 21.11.2020).

13. Коллоборативний робот-маніпулятор розроблений в Білорусі. URL: https://itkvariat.by/news/218-kollaborativnyy-robot-manipulyator-razrabotan-v-belarusi.html (дата звернення: 21.11.2020).

14. Робототехніка та об’єкти управління факультету ПМ-ПУ. URL: http://www.apmath.spbu.ru/ru/admission/ admission8.html (дата звернення: 21.11.2020).

15. Датчик перешкоди (модуль YL-63). URL: https://arduino.ua/prod1212-datchik-prepyatsviya (дата звернення: 22.11.2020).

16. Індуктивний датчик наближення LJ12A3-4-Z-BX. URL: https://arduino.ua/prod1426-indyktivnii-datchik-priblijeniya-lj12a3-4-zbx (дата звернення: 22.11.2020).

17. Датчик кольору TCS230. URL: http://arduino.ua/ prod252-datchik-cveta-tcs230 (дата звернення: 22.11.2020).

18. LCD дисплей 16x2 з синьою підсвіткою. URL: http://arduino.ua/prod169-lcd-displei-16x2-s-sinei-podsvetkoi (дата звернення: 22.11.2020).

19. Сервопривід MG996R 15 кг. URL: http://arduino.ua/prod272-servoprivod-mg996r-15-kg (дата звернення: 22.11.2020).

20. Arduino Mega 2560 R3 (CH340). URL: http://arduino.ua/prod2611-arduino-mega2560-r3-ch340 (дата звернення: 22.11.2020).

21. Блок живлення MN-120-12 12В 10А 120ВТ. URL: https://led-lampa.com.ua/p36378975-blok-pitaniya-120.html (дата звернення: 22.11.2020).

22. Понижуючий перетворювач 5А на XL4015 з регулюванням струму і напруги. URL: https://arduino.ua/prod2163-ponijaushhii-preobrazovatel-5a-na-xl4015-s-regylirovkoi-toka-i-napryajeniya (дата звернення: 22.11.2020).

23. BH1750 - цифровий датчик освітлення/люксметр (модуль GY-302). URL: https://micro-pi.ru/bh1750-gy-302 (дата звернення: 23.11.2020).

24. Датчик температури Arduino DS18B20. URL: https://arduinomaster.ru/ datchiki- arduino/ arduino-ds18b20/ (дата звернення: 23.11.2020).

25. Комп'ютерний зір: технології, ринок, перспективи. URL: https://www.tadviser.ru/index.php/Статья:Компьютерное_зрение:_технологии,_рынок,_перспективы (дата звернення: 25.11.2020).

26. Абрамс А.Д., Вебкамс Р.Б. Web interfaces to create live 3D environments. In Proceedings of the 18th ACM International Conference on Multimedia, Toronto. Canada. 2010. С. 331–340.

27. Мухамедіяров Р.М. Машинний зір: поняття, завдання та області застосування. Казахський національний технічний університет К.І.Сатпаєва, Алмати. Казахстан. URL: http://www.rusnauka.com/25_SSN_2009/Informatica/51050.doc.htm (дата звернення: 26.11.2020).

28. Френг Л. Основи машинного зору. Control Engineering. Росія. 2019. № 1. С. 59-61.

29. Noonv. OpenCV крок за кроком. Знаходження контурів та дії з ними. URL: http://robocraft.ru/blog/computervision/640.html (дата звернення: 27.11.2020).

30. Гороховатський В.А. Структурний аналіз та інтелектуальна обробка даних в комп’ютерному зорі: монографія. Харьків: «Компанія СМІТ». 2014. 317 с.

31. Махадаксщми Т., Муехаиах Р., Сваминатхан П. Review article: an overview of template matching technique in image processing. Research Journal of Applied Sciences, Engineering and Technology. vol. 4. № 24. 2012. С. 5469–5473.

32. Нейронні мережі: розпізнавання образів і зображень за допомогою ШІ. URL: https://center2m.ru/ai-recognition#:~:text=Распознавание%20образов%20нейронными%20сетями,представляют%20собой%20сети%20нервных%20клеток.&text=Их%20применяют%20для%20прогнозирования%2C%20распознавания,д (дата звернення: 28.11.2020).

34. Глибоке навчання: що це таке? URL: https://stfalcon.com/ru/blog/post/deep-learning-what-it-is#:~:text=Глубокое%20обучение%20—%20это%20набор%20алгоритмов,состоящих%20из%20нескольких%20нелинейных%20преобразований.&text=Эти%20ИНС%20получают%20алгоритмы%20обучения,для%20повышения%20эффективности%20процессов%20обучения (дата звернення: 28.11.2020).

35. Краснопевцев Б.В. Фотограмметрія: навч. Посіб. / Моск. держ. ун-т геодезії та картографії. Москва. 2008. 160 с.

36. SLAM. URL: http://robocraft.ru/blog/technology/724.html (дата звернення: 29.11.2020).

37. Азаренко Д.С. Детектування об'єкта на зображенні та визначення його зміщення на двох різних зображеннях. Інститут проблем штучного інтелекту МОН України та НАН України, м. Донецьк. Україна. №3. 2013. С. 90 - 97.

38. Матеріальна точка. URL: https://ru.wikipedia.org/wiki/Материальная_точка#:~:text=Материальная%20точка%2C%20свобода%20перемещения%20которой,пренебречь%20его%20формой%20и%20размерами) (дата звернення: 04.11.2020).

39. Механіка. URL: http://fizmat.by/kursy/kinematika/ravnomernoe_(дата звернення: 05.11.2020).

40. Вебер М.Я. Система автоматичного керування механічною рукою. Національний дослідницький Томський політехнічний університет. Томськ. 2018. 79 с.

41. Координатний спосіб завдання рухомої точки. URL: https://isopromat.ru/teormeh/obzornyj-kurs/koordinatnyj-sposob-zadania-dvizhenia-tochki (дата звернення: 07.11.2020).

42. Крейг Д. Введення у робототехніку. Механіка та управління. Інститут комп'ютерних досліджень. 2013. 564 с.

43. Данілов О.В., Кропотов О.М., Трифонов О.В. Загальний підхід до вирішення зворотного завдання кінематики для маніпулятора послідовної структури за допомогою кінцевого повороту і зсуву. Інститут прикладної математики імені М.В. Келдиша. Москва. 2018. 15 с.

44. Огляд методів вирішення зворотного завдання кінематики.URL: http://www.aptechsar.com/hp/task_ozk.shtml (дата звернення: 09.11.2020).

45. Масягін В.Б. Математичне моделювання та інформаційні технології при проектуванні: конспект лекцій. Омськ. 2015. 136 с.

46. Борисов О.І., Громов В.С., Пиркін А.А. Методи управління робототехнічними програмами: учбовий посібник. Університет ІТМО. Санкт-Петербург. 2016. 105с.

47. Фомін Т.А. Розробка та дослідження системи управління маніпулятором: бакалаврська робота. Сибірський федеральний університет. Політехнічний інститут. Красноярськ. 2019. 78 с.

48. Inverse Kinematics. URL: https://www.eecs.yorku.ca/course_archive/2017-18/W/4421/lectures/Inverse%20kinematics%20-%20annotated.pdf (дата звернення: 11.11.2020).

49. Ch. 3: Inverse Kinematics. Ch. 4: Velocity Kinematics. Oslo University Hospital. URL: https://www.uio.no/studier/emner/matnat/ifi/INF3480/v14/undervisningsmatriale/lec05-Inverse-VelocityKinematicsI.pdf (дата звернення: 12.11.2020).

50. Що таке CUDA ядра у відеокарті. URL: https://ru.pickgamer.com/post/chto-takoe-cuda-yadra-v-videokarte-432 (дата звернення: 29.11.2020).

51. Python. URL: https://ru.wikipedia.org/wiki/Python (дата звернення: 30.11.2020).

52. Visual Studio Code. URL: https://ru.wikipedia.org/wiki/Visual_Studio_Code (дата звернення: 30.11.2020).

53. Arduino IDE: програмне середовище для розробки під Ардуіно. URL: https://arduinoplus.ru/arduino-ide-opisanie-gde-skachat/ (дата звернення: 30.11.2020).

54. Qt Designer. URL: https://ru.wikipedia.org/wiki/Qt_Designer (дата звернення: 30.11.2020).

55. Міняйло Н. О., Пономаренко К. А., Просвірнін А. Л., Востоцький С. М. Прототип системи управління сортувальним комплексом. Металургія : наукові праці Інженерного інституту Запорізького національного університету. . 2019. № 1. С. 61-66.

56. Міняйло Н. О., Пономаренко К. А. Інтелектуальна система управління роботою сортувального комплексу. Збірник наукових праць студентів, аспірантів і молодих вчених «Молода наука-2020» Запоріжжя: Запорізький національний університет, 2020. Т. 5 С. 125-126. URL: http://sites.znu.edu.ua/stud-sci-soc/tom_5.pdf.

57. Пономаренко К. А., Міняйло Н. О. Прототип роботизованого сортувального комплексу. Матеріали XXIV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Металургія як основа сучасної промисловості. Запоріжжя: ІІ ЗНУ, 2019. Т. 1 С. 70.

58. Пономаренко К. А., Міняйло Н. О. Удосконалення роботизованого комплексу за рахунок інверсної кінематики. Матеріали XXV науково-технічної конференції студентів, магістрантів, аспірантів, молодих вчених та викладачів. Запоріжжя: ІННІ ЗНУ, 2020. С. 70.

Додаток А

Частина коду зміненого файлу image.c

```
#include "image.h"
#include "utils.h"
#include "blas.h"
#include "cuda.h"
#include <stdio.h>
#include <math.h>

#define STB_IMAGE_IMPLEMENTATION
#include "stb_image.h"
#define STB_IMAGE_WRITE_IMPLEMENTATION
#include "stb_image_write.h"

int windows = 0;

float colors[6][3] = { {1,0,1}, {0,0,1},{0,1,1},{0,1,0},{1,1,0},{1,0,0} };

float get_color(int c, int x, int max)
{
   float ratio = ((float)x/max)*5;
   int i = floor(ratio);
   int j = ceil(ratio);
   ratio -= i;
   float r = (1-ratio) * colors[i][c] + ratio*colors[j][c];
   //printf("%f\n", r);
   return r;
}

image mask_to_rgb(image mask)
{
   int n = mask.c;
   image im = make_image(mask.w, mask.h, 3);
   int i, j;
   for(j = 0; j < n; ++j){
      int offset = j*123457 % n;
      float red = get_color(2,offset,n);
      float green = get_color(1,offset,n);
      float blue = get_color(0,offset,n);
      for(i = 0; i < im.w*im.h; ++i){
         im.data[i + 0*im.w*im.h] += mask.data[j*im.h*im.w + i]*red;
         im.data[i + 1*im.w*im.h] += mask.data[j*im.h*im.w + i]*green;
         im.data[i + 2*im.w*im.h] += mask.data[j*im.h*im.w + i]*blue;
      }
   }
```

```
    return im;
}

static float get_pixel(image m, int x, int y, int c)
{
    assert(x < m.w && y < m.h && c < m.c);
    return m.data[c*m.h*m.w + y*m.w + x];
}
static float get_pixel_extend(image m, int x, int y, int c)
{
    if(x < 0 || x >= m.w || y < 0 || y >= m.h) return 0;
    /*
    if(x < 0) x = 0;
    if(x >= m.w) x = m.w-1;
    if(y < 0) y = 0;
    if(y >= m.h) y = m.h-1;
    */
    if(c < 0 || c >= m.c) return 0;
    return get_pixel(m, x, y, c);
}
static void set_pixel(image m, int x, int y, int c, float val)
{
    if (x < 0 || y < 0 || c < 0 || x >= m.w || y >= m.h || c >= m.c) return;
    assert(x < m.w && y < m.h && c < m.c);
    m.data[c*m.h*m.w + y*m.w + x] = val;
}
static void add_pixel(image m, int x, int y, int c, float val)
{
    assert(x < m.w && y < m.h && c < m.c);
    m.data[c*m.h*m.w + y*m.w + x] += val;
}

static float bilinear_interpolate(image im, float x, float y, int c)
{
    int ix = (int) floorf(x);
    int iy = (int) floorf(y);

    float dx = x - ix;
    float dy = y - iy;

    float val = (1-dy) * (1-dx) * get_pixel_extend(im, ix, iy, c) +
        dy     * (1-dx) * get_pixel_extend(im, ix, iy+1, c) +
        (1-dy) *   dx   * get_pixel_extend(im, ix+1, iy, c) +
        dy     *   dx   * get_pixel_extend(im, ix+1, iy+1, c);
    return val;
}
```

```
void composite_image(image source, image dest, int dx, int dy)
{
   int x,y,k;
   for(k = 0; k < source.c; ++k){
      for(y = 0; y < source.h; ++y){
         for(x = 0; x < source.w; ++x){
            float val = get_pixel(source, x, y, k);
            float val2 = get_pixel_extend(dest, dx+x, dy+y, k);
            set_pixel(dest, dx+x, dy+y, k, val * val2);
         }
      }
   }
}

image border_image(image a, int border)
{
   image b = make_image(a.w + 2*border, a.h + 2*border, a.c);
   int x,y,k;
   for(k = 0; k < b.c; ++k){
      for(y = 0; y < b.h; ++y){
         for(x = 0; x < b.w; ++x){
            float val = get_pixel_extend(a, x - border, y - border, k);
            if(x - border < 0 || x - border >= a.w || y - border < 0 || y - border >= a.h) val
= 1;
            set_pixel(b, x, y, k, val);
         }
      }
   }
   return b;
}
```

Додаток Б

Програмний код для розрахунку інверсної кінематики на мові Python

```
#Розрахунок інверсної кінематики
def inverse_kinematics():
    def inverseMatrix():
        #Об'явленя глобальних змінних
        global R0_4_center_spherical_wrist, R0_4_center_spherical_wrist_short
        global Work_Frame

        #Матриця тотожного перетворення
        Work_Frame = np.array([[1, 0, 0, 0],
                               [0, 1, 0, 0],
                               [0, 0, 1, 0],
                               [0, 0, 0, 1]])

        Tool_Frame = np.array([[1, 0, 0, 0],
                               [0, 1, 0, 0],
                               [0, 0, 1, 0],
                               [0, 0, 0, 1]])

        #Матриця на основі метода Ейлера
        R0_T = np.array([[math.cos(math.radians(Rotation_PositionR)) *
math.cos(math.radians(Rotation_PositionY)) - math.cos(math.radians(Rotation_PositionP)) *
math.sin(math.radians(Rotation_PositionR)) * math.sin(math.radians(Rotation_PositionY)),
math.cos(math.radians(Rotation_PositionY)) * math.sin(math.radians(Rotation_PositionR)) +
math.cos(math.radians(Rotation_PositionR)) * math.cos(math.radians(Rotation_PositionP)) *
math.sin(math.radians(Rotation_PositionY)), math.sin(math.radians(Rotation_PositionP)) *
math.sin(math.radians(Rotation_PositionY)), PositionX],
                         [math.cos(math.radians(Rotation_PositionP)) *
math.cos(math.radians(Rotation_PositionY)) * math.sin(math.radians(Rotation_PositionR)) +
math.cos(math.radians(Rotation_PositionR)) * math.sin(math.radians(Rotation_PositionY)),
math.cos(math.radians(Rotation_PositionR)) * math.cos(math.radians(Rotation_PositionP)) *
math.cos(math.radians(Rotation_PositionY)) - math.sin(math.radians(Rotation_PositionR)) *
```

```
math.sin(math.radians(Rotation_PositionY)), math.cos(math.radians(Rotation_PositionY)) *
math.sin(math.radians(Rotation_PositionP)), PositionY],
                    [math.sin(math.radians(Rotation_PositionR)) *
math.sin(math.radians(Rotation_PositionP)), math.cos(math.radians(Rotation_PositionR)) *
math.sin(math.radians(Rotation_PositionP)), -math.cos(math.radians(Rotation_PositionP)),
PositionZ],
                    [0, 0, 0, 1]])

        #Тотожне перетворення та знаходження матриці положення перших трьох
ланок
        R0_T_offset_by_Work_Frame = R0_T.dot(Work_Frame)
        R0_T_offset_by_Work_Frame[0][0] = R0_T_offset_by_Work_Frame[0][0] * -1
        R0_5 = R0_T_offset_by_Work_Frame.dot(Tool_Frame)

        R0_5_Remover = np.array([[math.cos(math.radians(180)),
math.sin(math.radians(180)), 0, 0],
                    [-math.sin(math.radians(180)) * math.cos(alpha5),
math.cos(math.radians(180)) * math.cos(alpha5), math.sin(alpha5), 0],
                    [math.sin(math.radians(180)) * math.sin(alpha5), -
math.cos(math.radians(180)) * math.sin(alpha5), math.cos(alpha5), -d5],
                    [0, 0, 0, 1]])

        R0_4_center_spherical_wrist = R0_5.dot(R0_5_Remover)

        R0_4_center_spherical_wrist_short = np.array([[ R0_4_center_spherical_wrist[0][0],
R0_4_center_spherical_wrist[0][1], R0_4_center_spherical_wrist[0][2]],
                                    [ R0_4_center_spherical_wrist[1][0],
R0_4_center_spherical_wrist[1][1], R0_4_center_spherical_wrist[1][2]],
                                    [ R0_4_center_spherical_wrist[2][0],
R0_4_center_spherical_wrist[2][1], R0_4_center_spherical_wrist[2][2]]])

    #Розрахунок при позитивному положенню J2 та J3
    def FWD():
        global pX_a1_FWD, J2_angle_FWD, J3_angle_FWD
        inverseMatrix()
```

```
            pX_FWD = math.sqrt((R0_4_center_spherical_wrist[1][3] ** 2) +
(R0_4_center_spherical_wrist[0][3] ** 2))
            pY_FWD = R0_4_center_spherical_wrist[2][3] - d1
            pX_a1_FWD = pX_FWD - a1
            pa2H_FWD = math.sqrt((pY_FWD ** 2) + ((pX_a1_FWD) ** 2))
            pa3H_FWD = math.sqrt((d4 ** 2) + (a3 ** 2))
            thetaA_FWD = math.degrees(math.atan(pY_FWD / (pX_FWD - a1)))

            blockThetaB = ((a2 ** 2) + (pa2H_FWD ** 2) - (pa3H_FWD ** 2))/(2 * a2 *
pa2H_FWD)
            if blockThetaB <= 1:
               thetaB_FWD = math.degrees(math.acos(((a2 ** 2) + (pa2H_FWD ** 2) -
(pa3H_FWD ** 2))/(2 * a2 * pa2H_FWD)))
               thetaC_FWD = 180 - math.degrees(math.acos((math.fabs(pa3H_FWD ** 2) +  (a2
** 2) - (pa2H_FWD ** 2))/(2 * math.fabs(pa3H_FWD) * a2)))
               thetaE_FWD = 90
               J2_angle_FWD = -(thetaA_FWD + thetaB_FWD)
               J3_angle_FWD = thetaC_FWD
               block = 0

            else:
               return

            thetaE_FWD = 90
            J2_angle_FWD = -(thetaA_FWD + thetaB_FWD)
            J3_angle_FWD = thetaC_FWD

         #Розрахунок при негативному положенню J2 та J3
         #MID
         def MID():
            inverseMatrix()
```

Додаток В

Програмний код для керування серводвигунами на мові С++

Підпрограма підключення сервоприводів:

```
#define servo0Pin 2
#define servo1Pin 3
#define servo3Pin 4
#define servo4Pin 5
#define servo5Pin 6
#define servo6Pin 7
```

Підпрограма прив'язування сервоприводів до виходів:

```
servo0.attach(servo0Pin);
 servo1.attach(servo1Pin);
 servo3.attach(servo3Pin);
 servo4.attach(servo4Pin);
 servo5.attach(servo5Pin);
 servo6.attach(servo6Pin);
```

Підпрограма руху маніпулятора в стартову позицію:

```
servo0.write(start_angle0);
 servo1.write(start_angle1);
 servo3.write(start_angle3);
 servo4.write(start_angle4);
 servo5.write(start_angle5);
 servo6.write(start_angle6);
```

Підпрограма налаштування виводу інформації на дісплей:

```
lcd.setCursor(0, 0);  // устанавливаем курсор в нужном положении на LCD
    lcd.clear(); // Очищяем дисплей
    lcd.print("Mat:");

    if (Sensor_Induct == 0)
    {
      lcd.print ("NMetal");
    }
    else if (Sensor_Induct == 1)
    {
      lcd.print("Metal");
      }
    lcd.setCursor(10,0);
    lcd.print("| M:"); // Выводим на дисплей "M"
    lcd.print(Counter_Metal); // Выводим на дисплей количество металических объектов
    lcd.setCursor(10,1);
    lcd.print("|NM:"); // Выводим на дисплей "NM"
    lcd.print(Counter_Not_Metal); // Выводим на дисплей количество неметалических
объектов
```

```
lcd.setCursor(0,1); // Переходим на следующую строку
lcd.print("Col:");
if (Color == 0)
{
lcd.print("-");
}
```

Підпрограма руху одної ланки маніпулятора:

```
for(angle4=start_angle4; angle4>take_position_angle4; angle4--) //  Звено 3
    {
     servo4.write(angle4);
     delay(Speed);
    }
```

Підпрограма розрахунку швидкості двигунів

```
float ACCStep = (HighStep * (ACCdur / 100));
float DCCStep = HighStep - (HighStep * (DCCdur / 100));
float AdjSpeed = (SpeedIn / 100);
float CalcRegSpeed = (SpeedMult / AdjSpeed);
int REGSpeed = int(CalcRegSpeed);
float ACCspdT = (ACCspd / 100);
float CalcACCSpeed = ((SpeedMult + (SpeedMult / ACCspdT)) / AdjSpeed);
float ACCSpeed = (CalcACCSpeed);
float ACCinc = (REGSpeed - ACCSpeed) / ACCStep;
float DCCspdT = (DCCspd / 100);
float CalcDCCSpeed = ((SpeedMult + (SpeedMult / DCCspdT)) / AdjSpeed);
float DCCSpeed = (CalcDCCSpeed);
float DCCinc = (REGSpeed + DCCSpeed) / DCCStep;
DCCSpeed = REGSpeed;
```

Додаток Г

Код інтерфейсу програми управління лабораторним стендом «Orion-I»

```
#Кнопка яка управляє положенням маніпулятора по осі X
def butXPlus():
    global PositionX
    global angleChange, positionChange
    global J1Angle, J2Angle, J3Angle, J4Angle, J5Angle
    J1Angle = int(J1Angle)
    J2Angle = int(J2Angle)
    J3Angle = int(J3Angle)
    J4Angle = int(J4Angle)
    J5Angle = int(J5Angle)
    if manualControl == True:
        if J1Angle < J1PosLim and J1Angle > J1NegLim and J2Angle < J2PosLim and J2Angle > J2NegLim and J3Angle < J3PosLim and J3Angle > J3NegLim and J4Angle < J4PosLim and J4Angle > J4NegLim and J5Angle < J5PosLim and J5Angle > J5NegLim:
            if block1 == 0:
                angleChange = 0
                positionChange = 1
                PositionX = PositionX + 10
                inverse_kinematics()
                txt()
                comOut()
                ui.info_panel.setText("X changed")
            else:
                ui.info_panel.setText("WARNING: Out of limit")
        else:
            ui.info_panel.setText("WARNING: Angle out of range")

#Кнопка що додає до значення J1 5 градусів
def butJ1AnglePlus():
    global J1Angle
    global angleChange, positionChange
    J1Angle = int(J1Angle)
```

```
        if manualControl == True:
            if J1Angle < J1PosLim:
                angleChange = 1
                positionChange = 0
                J1Angle = J1Angle + 5
                forward_kinematics()
                txt()
                comOut()
                ui.info_panel.setText("Angle J1 changed")
            else:
                ui.info_panel.setText("WARNING: Angle out of range")
        else:
            ui.info_panel.setText("WARNING: Manual control is blocked")


    #Кнопка Старту
    def butStart():
        global startBlock, stopBlock, robotStart, robotStop, manualBlock, autoBlock,
manualControl, autoControl
        startBlock = startBlock + 1
        if startBlock == 1 and manualControl != 1:
            robotStart = 1
            robotStop = 0
            stopBlock = 0
            manualBlock = 0
            autoBlock = 0
            ui.info_panel.setText("Robot started")
        elif manualControl:
            startBlock = 0
            ui.info_panel.setText("WARNING: Robot in Manual Control")
        elif startBlock != 1 and robotStart and manualControl != 1:
            ui.info_panel.setText("WARNING: Robot is already started")
```