Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

перший (бакалаврський)
(рівень вищої освіти)

Розроблення системи автоматизації для сортування пластмасових виробів з використанням маніпулятора із 6 ступенями вільності
(тема)

Виконав:
здобувач 4 року навчання,
групи АКТАКІТ-21-3
Дмитро Савенко
(власне ім'я, прізвище)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва освітньої програми)

Керівник професор Юрій РОМАШОВ
(посада, власне ім'я, прізвище)

Допускається до захисту
Зав. кафедри КІТАР _____ Ігор НЕВЛЮДОВ
(підпис) (власне ім'я, прізвище)

2025 р.

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

"30" травня 2025 р.  Дмитро САВЕНКО

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва)
Тип програми Освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
« 07 » квітня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Савенку Дмитру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення системи автоматизації для сортування пластмасових виробів з використанням маніпулятора із 6 ступенями вільності

Затверджена наказом по університету від 19.05.2025 р. № 390 Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 21.06.2025 р.
3. Вихідні дані до роботи
3.1 Модель автоматизованої системи сортування за допомогою реалізації технічного зору
3.2 Реалізація симуляційного макета з використанням технічного зору
3.3 Реалізація модуля технічного зору
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Аналіз систем автоматизованого сортування
4.2 Аналіз роботи роботів-маніпуляторів
4.3 Аналіз методів та технологій для реалізації технічного зору
4.4 Розробка симуляційного макета для реалізації автоматичного сортування
4.5 Розробка та проєктування скрипту для отримання зображення з камер та управління роботом-маніпулятором
4.6 Використання OpenCV для реалізації зору та CoppeliaSim Education для реалізації симуляційного макета

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)
Демонстраційний матеріал у вигляді презентації (*.pptx) – 16 с.

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

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Аналіз публікацій та патентний пошук за темою роботи	07.04-13.04.2025	Виконано
2	Обгрунтування актуальності роботи, визначення мети, предмета та об'єкту розробки	14.04-20.04.2025	Виконано
3	Пошук розв'язання поставленої задачі в роботі за допомогою програмування	21.04-27.04.2025	Виконано
4	Аналіз аналогічних пристроїв. Аналіз недоліків наявних систем та пошуки усунення проблем	28.04-04.05.2025	Виконано
5	Розробка симуляційного макета в Coppelia Sim	05.05-11.05.2025	Виконано
6	Розробка кода для технічного зору	12.05-02.06.2025	Виконано
7	Подання роботи на перевірку Інтернет-сервісом StrikePlagiarism	10.06.2025	Виконано
8	Оформлення звіту	11.06-14.06.2025	Виконано
9	Оформлення пояснювальної записки	15.06.2025	Виконано
10	Подання роботи на рецензію	17.06.2025	Виконано
11	Подання роботи на підпис зав. кафедри	19.06.2025	Виконано
12	Подання кваліфікаційної роботи в ЕК	20.06.2025	Виконано

Дата видачі завдання 07 квітня 2025 р.

Здобувач _____ Дмитро САВЕНКО
(підпис) (власне ім'я, прізвище)

Керівник роботи _____ проф. Юрій РОМАШОВ
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 54 с., 2 табл., 36 рис., 2 додатків, 20 джерел.

ІНТЕРФЕЙС, ІНФОРМАЦІЙНА СИСТЕМА, ПРОГРАМНИЙ ЗАСІБ, АЛГОРИТМ, СИСТЕМА СОРТУВАННЯ, АВТОМАТИЗАЦІЯ СОРТУВАННЯ, РОБОТИ-МАНІПУЛЯТОРИ, ТЕХНОЛОГІЇ ТА МЕТОДИ ТЕХНІЧНОГО ЗОРУ, COPELLIASIM

Об’єкт розробки – система автоматизації для сортування пластмасових виробів з використанням маніпулятора із 6 ступенями вільності.

Предмет розробки – методи розпізнавання об’єктів за допомогою технічного зору.

Мета роботи – покращення швидкості розпізнавання за рахунок розробки симуляційного макету.

В першому розділі розглядано теоретичні відомості про роботів, системи автоматизації та методи реалізації технічного зору за допомогою різних допоміжних засобів.

В другому розділі розроблено допоміжні діаграми для створення симуляційного макета, логічна схема ідентифікації об’єкта за допомогою технічного зору.

В третьому розділі представлені результати виконаної роботи для реалізації системи автоматичного сортування за допомогою робота з шістьма восьми та технічного зору.

У результаті розроблена симуляційний макет у середовищі CoppeliaSim Education, окрема камера технічного зору на основі Yolo, що має власну базу даних.

В процесі виконання кваліфікаційної роботи потрібно вирішити теоретичні та практичні задачі:

– аналіз систем автоматизованого сортування;

– аналіз роботи роботів-маніпуляторів;

– аналіз методів та технологій для реалізації технічного зору;

– розробка симуляційного макета для автоматичного сортування;

– розробка та проєктування скрипта для отримання зображення з камер;

– використання OpenCV для реалізації зору.

Робота виконана згідно [1-3].

ABSTRACT

Explanatory note: 54 pages, 2 tables, 36 pictures, 2 addiction, 20 sources.

INTERFACE, INFORMATION SYSTEM, SOFTWARE, ALGORITHM, SORTING SYSTEM, SORTING AUTOMATION, MANIPULATOR ROBOTS, TECHNOLOGIES AND METHODS OF TECHNICAL VISION, COPELLIASIM

The object of development is a system for automating the sorting of plastic products using a manipulator with 6 degrees of freedom.

The subject of development is methods for recognizing objects using machine vision.

The purpose of the work is to increase the recognition speed by developing a simulation model.

The first section discusses theoretical information about robots, automation systems and methods for implementing technical vision using various aids.

The second section develops auxiliary diagrams for creating a simulation layout, a logical scheme for identifying an object using technical vision.

The third section presents the results of the work performed to implement an automatic sorting system using a six-eight robot and technical vision.

As a result, a simulation layout was developed in the CoppeliaSim Education environment, a separate technical vision camera based on Yolo, which has its own database.

In the process of performing the qualification work, it is necessary to solve theoretical and practical problems

- analysis of automated sorting systems;
- analysis of the operation of manipulator robots;
- analysis of methods and technologies for implementing technical vision;
- development of a simulation layout for implementing automatic sorting;
- development and design of a script for obtaining images from cameras;

– use of OpenCV for implementing vision.

The work was performed according to [1-3].

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗ – програмне забезпечення

САПР – система автоматизованого проєктування

ЧПУ – числове програмне управління

BIM – Building Information Modeling

CAS – Computer Algebra System

GIS – Geographic Information System

HMI – Human Machine Interface

IDE – Integrated Development Environment

PC – Personal Computer

RemoteAPI – Remote Application Programming Interface

ROS – Robot Operating System

ВСТУП

Шлях автоматизації на сьогодні все стає більш популярним серед фірм та виробництв. Кожний куток, квадратний метр чи підвал намагаються використовувати максимально ефективно та з максимальною користю, тому що:

– безпека для робочого персоналу, а також можливість роботи з небезпечними елементами через роботи-маніпулятори;

– зменшена кількість персоналу, що забезпечує більшу безпеку та менші витрати людино-годин та людино ресурсу;

– менша оплата оренди за земельну ділянку та більша його рентабельність на $м^2$;

– розширювання виробництва за менший об'єм використаного місця.

Кожен намагається використовувати роботів, маніпуляторів, автоматизовані конвеєрні лінії, сенсорику та все інше, що належить до Industry 4.0 [4]. Сортування пластикових виробів є дуже важливою темою сьогоднішнього часу, бо його об’єм вироблення є більшою, ніж об’єм перероблення. Тому система автоматизованого сортування в жилих будівлях буде не лише забезпечувати сортування пластику для подальшого його перероблення, а також облегшить майбутнє сортування вже на більш професійних системах сортування пластикових виробів.

Таким чином метою моєї кваліфікаційної роботи є покращення швидкості розпізнавання за рахунок розробки симуляційного макету.

Об’єкт розробки – система автоматизації сортування пластмасових виробів з використанням маніпулятора із 6 ступенем свободи.

Предмет розробки – методи розпізнавання об’єктів за допомогою технічного зору.

Для вирішення цього завдання були поставлені наступні теоретичні та практичні задачі:

– аналіз систем автоматизованого сортування;

– аналіз роботи роботів-маніпуляторів;

– аналіз методів та технологій для реалізації технічного зору;

– розробка симуляційного макета для реалізації автоматичного сортування;

– розробка та проєктування скрипта для отримання зображення з камер та управління роботом-маніпулятором;

– використання OpenCV для реалізації зору та CoppeliaSim Education для реалізації симуляційного макета.

1 АНАЛІЗ НАЯВНИХ СИСТЕМ СОРТУВАННЯ ПЛАСТИКОВИХ ВИРОБІВ ЗА ДОПОМОГОЮ РОБОТІВ

Пластикові вироби як й багато інших виробів також поділяються на різні класи, які наведено на рис. 1.1.

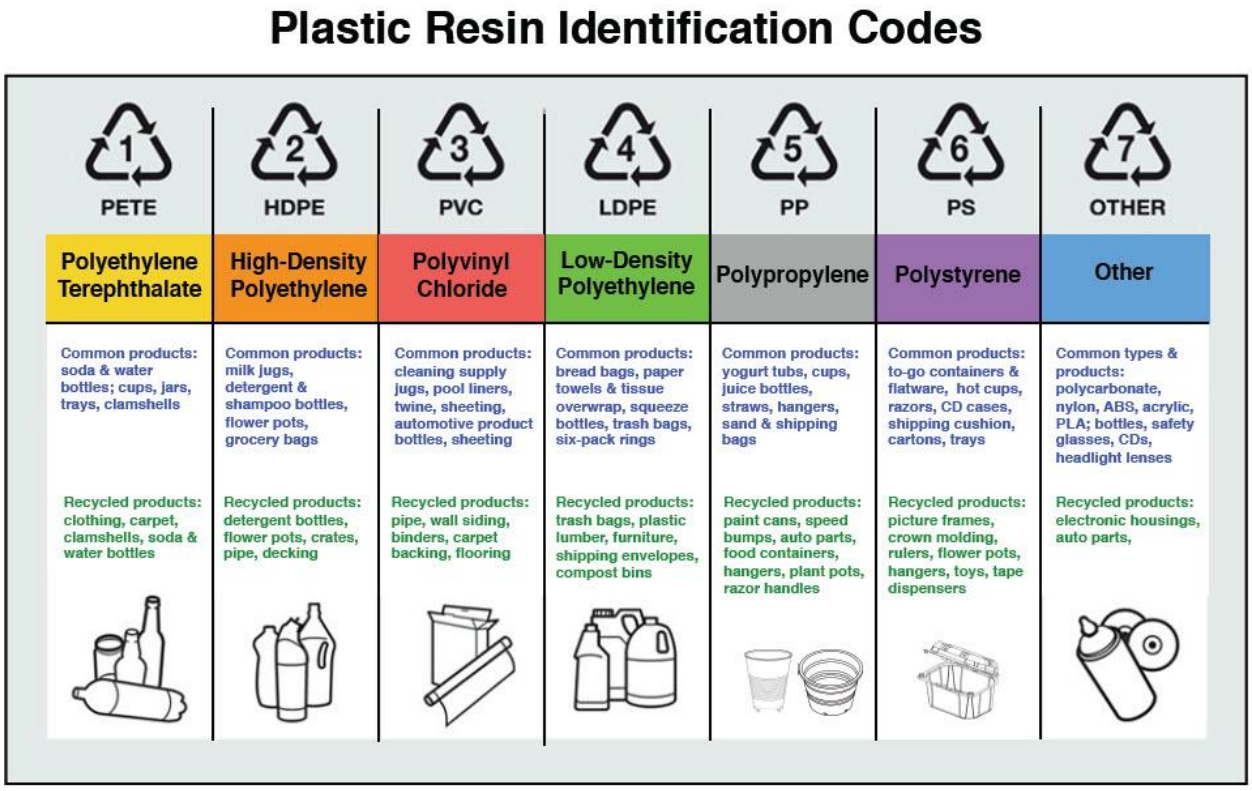


Рисунок 1.1 – Ідентифікаційний код для перероблювання пластикових виробів [5]

Кожен із цих кодів є на пластикових виробах, хай то кетчуп, вода чи мастило для двигуна автівки. Вони відіграють значну частку на етапах перероблювання сміття. Кожен із цих кодів можна також ще класифікувати за хімічним складом, властивостями та сферою застосування:

– PET або PETE (Поліетилентерефталат), використовують для пляшок для напоїв та олії або для одноразової харчової упаковки. Він є міцним, прозорим, вологозахисним та легким;

– HDPE (Поліетилен високої щільності), використовують для побутової хімії та молока, контейнерів, дитячих іграшок тощо. Так само є міцним, хімічно та вологостійким;

– PVC (Полівінілхлорид), може бути гнучким або твердим залежно від складу, не може бути використаним для зберігання харчових продуктів через хімічний склад та його взаємодію зі світлом. Використовують в парі з будівельними матеріалами або електрокабелями;

– LDPE (Поліетилен низької щільності) – м'який, еластичний та вологостійкий та може бути використаний для харчової промисловості, але має недолік у перероблюванні, через його складність розщеплювання;

– PP (Поліпропілен) – найпоширеніший клас у промисловості одноразових медичних виробах, швидкоїжа, кавових автоматів, бо є стійким до високих температур та хімічних речовин. До переваг можна віднести його легкість, міцність та використання у мікрохвильовій печі.;

– PS (Полістирол) – Буває у твердій формі або у вигляді пінопласту, небезпечний при нагріваннях, через виділяння стиролу (потенційно токсична сполука), також є крихким;

– OTHER (Інші пластики) – до цього класу відносяться всі види форм та хімічний склад пластику, що не ввійшли до перших шести класів, до прикладу ABS, PLA, PLA+ для 3D-друку, диски, багаторазові пляшки [6-7];

Тому для сортування всіх цих класів є дуже важливим їх хімічний склад та змога пропуску світла, бо найпоширеніші варіанти сортування пластикових виробів це:

- поляризаційний зір;
- комп'ютерний зір;
- можливість пропуску світла через певний виріб;

1.1 Поляризаційний зір

Поляризаційний зір (нанотехнологія) – астрономічне джерело випромінювання, що за допомогою процесу поляризації світла, при якому електричне поле світлової хвилі коливається в певному напрямку. Пластикові вироби поводяться по різному із поляризованим світлом (рис. 1.2), тобто кожне електричне поле пластикового виробу має свою силу, грубість, потужність спектра Фур'є.

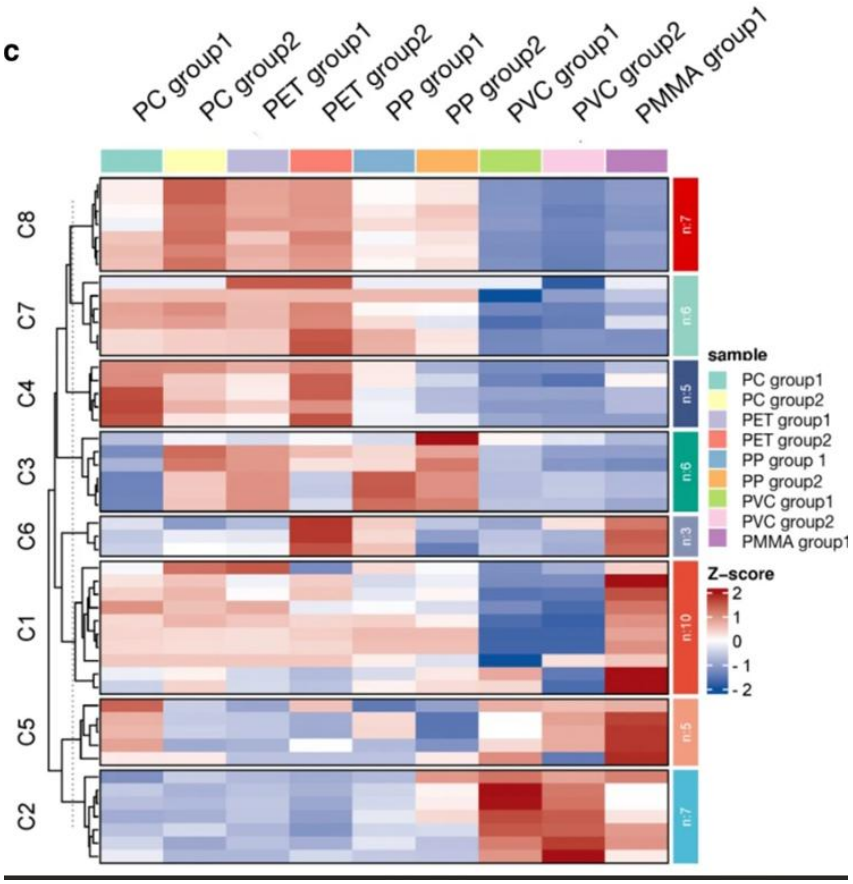


Рисунок 1.2 – Діаграма залежностей виду пластику до різних характеристик при поляризаційному світлі [8].

Аналізуючи дану діаграму, можемо зробити висновок, що кожен із матеріалів має різні характеристичні властивості на кожному рівні світла, але

якщо беремо певну характеристику, як силу чи об’ємну ентропію, то можемо бачити на рис. 1.3, що вони мають велику розбіжність на нанорівні, що допомогою нам краще його сортувати.

а) б)

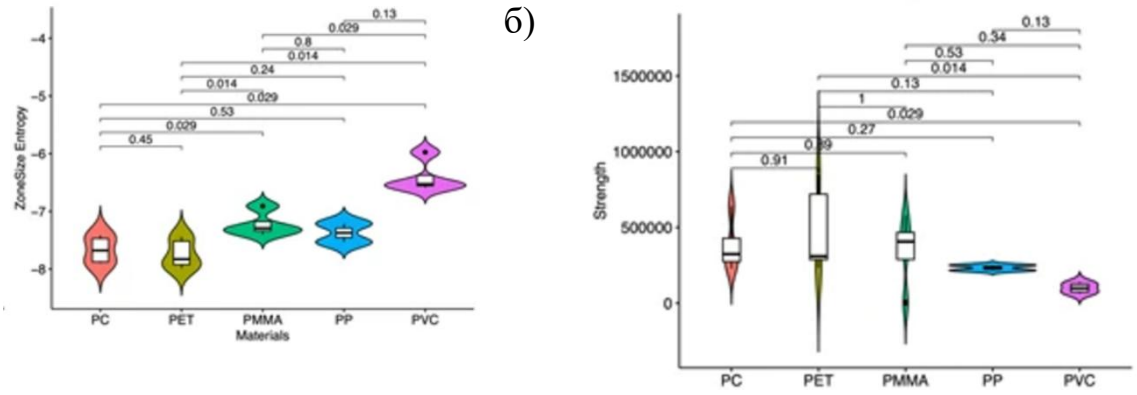


Рисунок 1.3 – Силова (а) та об’ємна ентропія (б) характеристика пластикових виробів [8].

Технологія все ще є новою, тому матеріали для використання її коштують дорого, але лідерами за продажами та точністю є камери-об’єктиви від компанії Sony (рис. 1.4), характеристики яких приведено в табл. 1.1.

Рисунок 1.4 – LUCID Vision Labs Triton™ TRI050S1-PC monochrome Polarisationskamera, Sony IMX264MZR, 5,0 MP

Таблиця 1.1 – Технічні характеристики камери Sony IMX264MZR

Параметри	Значення
Розміри	29x29x45мм
Пам’ять	128 МБ
Сенсор	Progressive Scan CMOS
Формат сенсору	2/3 "
Піксельне полотно	2,488x2,048px
Розмір сенсору	8,45x7,07мм
Чіткість пікселів	12bit
Звуковий діапазон	71,34 Дб
Тип живлення	2.5 (Зовнішнє живлення), або 3.1(Power over Ethernet)
Підключення до контроллеру	8-pin M8
Робоча температура	-20 до +55 °C
Синхронізація	GPIO, Software або PTP (IEEE 1588)
Відеовихід	GigE (PoE)
Тип захисту	IP67

1.2 Комп’ютерний зір

Комп’ютерний зір – це галузь штучного інтелекту, що дозволяє розпізнавати, аналізувати та інпретувати комп’ютеру візуальну інформацію з навколишнього середовища. Завдяки цій технології були розроблені багато технологій для автоматизації, безпеки, медицині, транспортній галузі та багато інших галузей.

Якщо брати галузь автоматизації на виробництві, то дана технологія використовується для «зору» робота, контролю якості та дефектоскопії. Завдяки мікроконтролеру підключеному до камери та написаному коду для розпізнавання певного об’єкта можна:

– зменшити використання людиногодин;
– зменшити ризики небезпеки при роботі з промисловими машинами;
– підвищити продуктивність;
– збільшити якість продукції;
– збільшити кількість виготовлення продукції у відношенні шт/год.

Технологія використовується вже не перше десятиріччя, бо вже у 1972 році був виготовлений перший інтелектуальний, автономний робот із комп'ютерним зором під назвою «SHAKEY» (рис. 1.5).

Рисунок 1.5 – SHAKEY, перший автономний, інтелектуальний робот

Його інтелектуальність полягала у тому, що він завдяки написаній програмі міг аналізувати простір навколо себе та малювати перешкоди на комп'ютері PDP-10 та аналізуючи їх, він міг прокладати свій майбутній шлях.

На сьогодні технології стали набагато компактнішими та дешевшими у виробництві, прикладом може слугувати популярна плата ESP-32 з різними додатковими модулями, такі як OV2640 Kamera (рис. 1.6).

Рисунок 1.6 – ESP-32 Cam Wi-Fi/Bluetooth Board with OV2640 Kamera(compatible with Arduino)

Де у своєму поєднанні можливо розробити комп’ютерний зір, що буде розпізнавати об’єкти за їх характеристиками або певними розмірами.

До навчання комп’ютерного зору причасна кожна людина, що реєструвалась на будь-якому сайті в мережі інтернет, бо технологія Captcha була розроблена як додатковий Firewall від спам-атак роботів та вже на результатах пройдених Captcha були створені перші штучні інтелекти для розпізнавання об’єктів на зображеннях, завдяки цьому було покращення функції радарів на дорозі, підвищено якість знімків у медицині, збільшилась якість продукції, що проходила перевірку завдяки комп’ютерному зору.

Сам процес розпізнавання комп’ютером те, що він бачить проходить наступним чином (рис. 1.7)

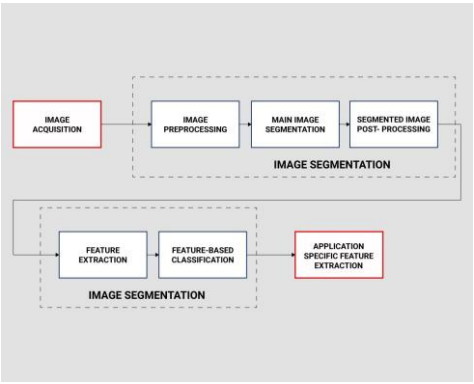


Рисунок 1.7 – Діаграма розпізнавання комп’ютером об’єктів на зображенні [9].

Напочатку комп’ютер отримує інформацію про об’єкти на зображенні (якщо це відбувається у реальному часі) чи аналізує зображення, де потім починаються два основних етапи, що відбуваються лише у певній послідовності й ніяк інше, а саме:

– сегментація зображення;

– сегментація зображення за запитом.

Тобто зображення може бути розділеним на різні методи аналізу:

– аналіз за допомогою машинного навчання;

– аналіз за допомогою глибокого навчання;

– аналіз за допомогою OpenCV (Open Source Computer Vision Library).

При використанні машинного навчання, початкові результати будуть незадовільними через те, що у базі даних немає будь-якої інформації щодо об’єктів, що мають бути проаналізовані

При використанні глибокого навчання, початкові результати будуть кращі, бо перед експлуатацією буде завантажено багато даних інформації до бази даних для того, щоб система одразу могла працювати на задовільний результат.

При використанні OpenCV результати будуть наближені до ідеального, бо OpenCV розташовує дуже великою базою даних з об’єктами для комп’ютерного зору [12].

Якщо коротко підбивати підсумки, то серед цих трьох методів аналізу, кращий є OpenCV через його велику базу даних та підтримкою багатьома мовами програмування (C++, C, C#, Python), якщо мова йде про безпеку даних, то лишаються лише методи машинного та глибокого навчання, але недоліком є те, що затрачаються великі людино та машинні ресурси на їх навчання та додавання багатьох систем безпеки для запобігання стороннього доступу з зовні інтернету, тобто встановлення локального сервера.

Популярними бібліотеками на базі мови Python є MobileNetV2 (рис. 1.8), Convolutional Neural Networks(CNNs), що являють собою майже готову систему розпізнавання об’єктів.

Рисунок 1.8 – Робота MobileNetV2 у розпізнаванні об’єктів (людей)

Система не є дуже точною, через її малу базу даних, але за допомогою додаткового скрипту, можливо додати індекс розпізнавання (i), де i > 0.5, то об’єкт буде розпізнаний та обведений, що можна бачити на рис. 1.8. Додатковими проблемами комп’ютерного зору є велика скупність об’єктів на зображені або недосвідченість зору у задані певного класу до об’єкта (рис. 1.9). Дана проблема може свідчити про те, що задані шаблони у системі комп'ютерного зору задані за розміром на зображені, та не мають додаткового скрипту для більш чіткого розпізнавання об’єкта. Таке саме працює й зі статуями та людьми (рис. 1.10), тому що, на першому етапі обробки зображення відбувається корекція експозицію, та забирання шумів із зображення, що виводить зображення на оптимальну яскравість.

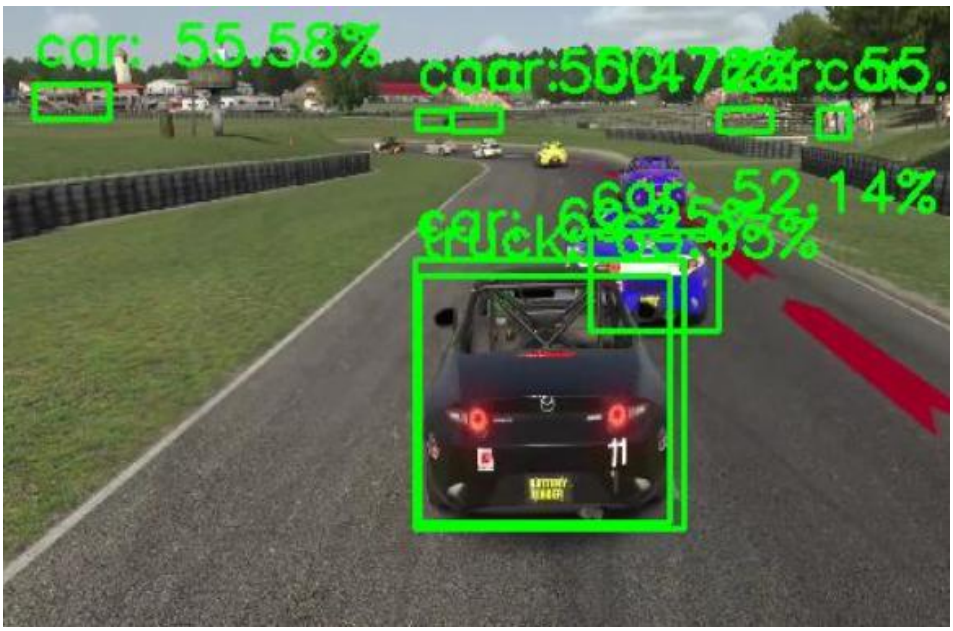


Рисунок 1.9 – Розпізнавання одного об’єкту як «Car», та «Truck»

Рисунок 1.10 – Неправильне слідкування заданого об’єкта (людини)

1.3 Сортування за допомогою пропуску світла

Сортування за допомогою пропуску світла є одною з основних технологій для сортування пластикових виробів, бо дана технологія є

дешевою, легкою у навчанні та має велику рентабельність. Для реалізації треба невеликий набір сенсорів, а саме:

– джерело ІЧ-світла (LED або галогенова лампа);

– іч-детектор (InGaAs або CMOS з фільтрацією);

– лінейний сканер (для конвеєрної лінії);

– сжате повітря для видалення потрібного пластику після розпізнавання.

Основним та керуючим законом для цього є закон Бугера-Ламберта-Бера [10], котрий описує, як світло стухає при проходженні через матеріал

$$I = I_0 \cdot e^{-\alpha \cdot d} \tag{1.1}$$

де I – інтенсивність пройденого світла,

I_0 – інтенсивність падаючого світла,

a - коефіцієнт поглинання,

d – товщина матеріалу.

Аналізуючи рис. 1.11 можемо вирахувати коефіцієнт поглинання пластикових виробів, дані наведені у таблиці 1.2.

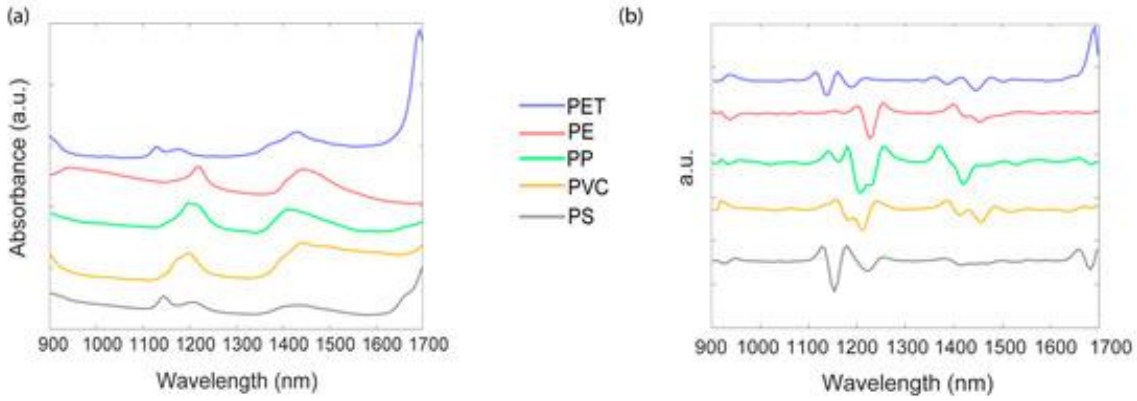


Рисунок 1.11 – Коефіцієнти поглинання пластикових виробів [11]

Таблиця 1.2 – Коефіцієнт поглинання для пластикових виробів [11].

Пластиковий виріб	Коефіцієнт поглинання, нм
PET	1420, 1660, 1910
PVC	1418, 1715, 1740
PE	1200, 1730
PP	1150, 1380
PS	1490, 1600
ABS	1500-1700
PC	1700-1800

Якщо ж значення із таблиці 1.2 нам невідомі та не знаємо товщину матеріалу, то за допомогою оптичної місткості (абсорбції) можемо знайти наш коефіцієнт поглинання:

$$A = \log_{10}\left(\frac{I_0}{I}\right) \tag{1.2}$$

Якщо товщина матеріалу відома:

$$\alpha = \frac{1}{d} \cdot \ln\left(\frac{I_0}{I}\right) \tag{1.3}$$

Але в автоматичних системах не вирішують рівняння вручну, а використовують машине навчання або алгоритми сопоставлення спектрів. Однією з популярних формул для кореляції є

$$R = \frac{\sum(S_i \cdot R_i)}{\sum S_i^2 \cdot \sum R_i^2} \tag{1.4}$$

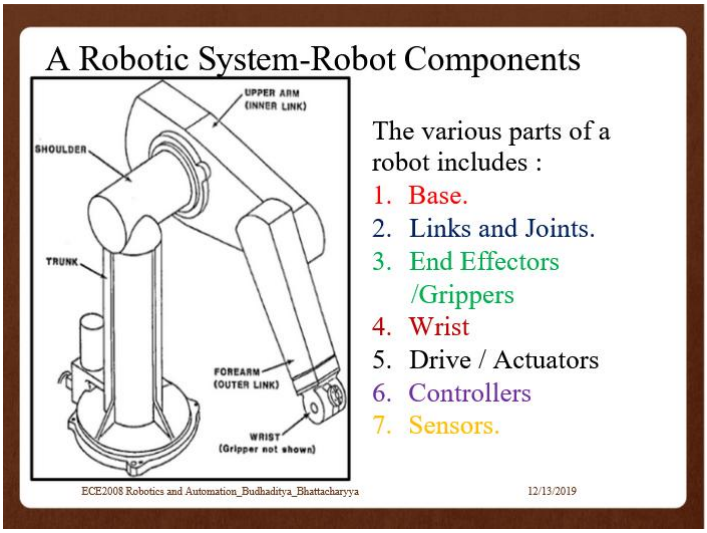
де S_i – значення спектру дослідчуйого образця на довжині хвилі i, R_i - значення референсного образця на тій самій же довжині хвилі i. Тобто, якщо

відповідь R ~ 1, то спектри майже ідентичні, R ~ 0 – спектри ортогональні(різні) R < 0, спектри протилежні, що свідчить про помилку у системі.

1.5 Аналіз роботів-маніпуляторів

У зв’язку зі швидким розвитком Industry 4.0 з'являється більше й більше автоматизованих та роботизованих маніпуляторів для специфічних завдань: пакетування, зварювання, пресування, для тестування, та інші.

Для початку треба з’ясувати, що таке індустріальні роботи маніпулятори. За дефініцію від IFR: індустріальний робот - автоматично контрольований, перепрограмований, багатофункціональний маніпулятор, програмується в трьох або більше осей, які можуть бути закріпленим на місці або мобільним для використання в програмах промислової автоматизації.



1 – Base (Основа) 2. Links and Joints (З’єднання та суглоби) 3. End Effectors/Grippers (Кінцеві ефектори/захвати) 4. Wrist (Зап’ястя) 5. Drive / Actuators (Привід/Актуатори) 6. Controllers (контролери) 7.Sensors (сенсори)

Рисунок 1.12 – Компоненти робота маніпулятора

1.6 Історія розвитку робототехніки

Початок розробки ще у XX сторіччі, коли Джордж Девол та Джозеф Енгельберт зробили першого програмованного робота з гідравлічними акураторами у 1959 році під назвою «UNIMATION, INC.». Після всіх бюрократичних процесів вже у 1961 цей робот був встановлено на лініях GM'у, вони виготовляли всеможливі деталі для автіков, двері, встановлення світла, збірка коробки передач тощо. Хоч цей робот (рис. 1.13) й був без мови програмування, але його «пам’ять» складала 200 різних кроків, що він міг запам’ятати.

Рисунок 1.13 – Перший робот «UNIMATION, INC.»

Щоб він запам’ятав кроки, треба було лише його переміщати в потрібні точки та позіційній детектор буде запам’ятовувати ці точки.

Другим проривним відкриттям в робототехниці був цилідричний робот «Versatran» (рис. 1.14), що був у 1962 році створений. Це був перший робот, що міг пересуватись у цилідричних координатах. Він так само був без мови програмування та був з мануальним налштауванням кроків його автоматизації.

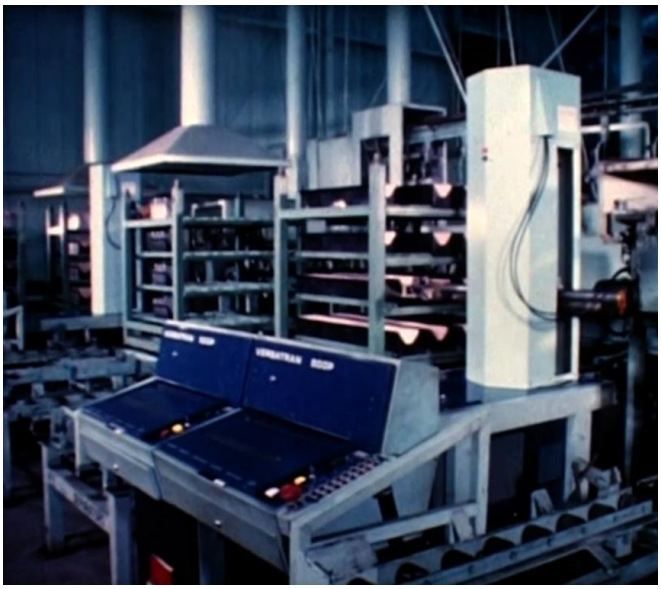


Рисунок 1.14 – Перший циліндричний робот «Versatran» з дистанційною панеллю управління

В порівнянні з першим роботом UNIMATION він мав більше точок у просторі, міг швидше та ефективніше виконувати різні завдання, мав пневматичні клешні та була можливість записувати до 4 программ управління.

Після всього цього, були побудований пробатько сьогоденних роботів маніпуляторів, так звана «Tentacle Arm» («Осьминожжа рука») (рис. 1.15) у 1968 році.

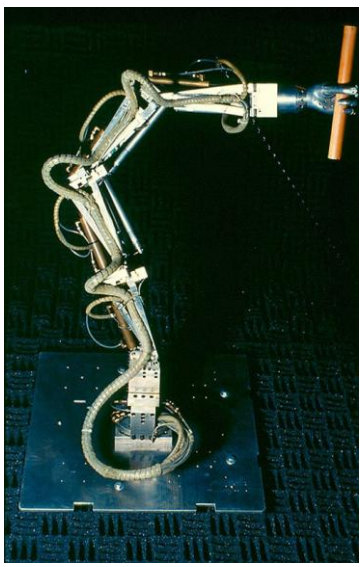


Рисунок 1.15 – Minsky Tentacle Arm (Осьминожа рука Минського)

Перші версії керувались за допомогою джойстика, але з часом був доданий комп’ютер «PDP-6» за для можливості автоматизування процесів руху та виконання задач де програмування маніпулятора задавалось за допомогою мови програмування Fortan (пізніше за допомогою LISP). Це все завдало те, що можливо було створювати прості HMI-інтерфейси, швидке перепрограмування програми та більш точне використання маніпулятора.

Після всіх вдалих тестів текущих роботів у 1969 році у Норвегії був створений перший комерційний маніпулятор-робот (рис. 1.16) для покрасу виробів для того, щоб був зменшений людський фактор та можливість зараження людей токсичними хімікаліями від фарби. Можна сказати, що цей робот був збіркою всіх існуючих на тих час роботів у єдиному екземлярі.

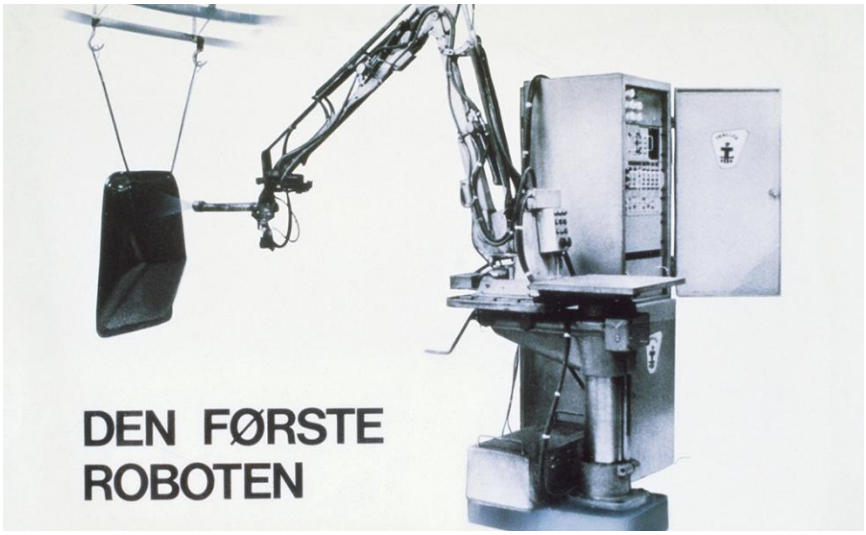


Рисунок 1.16 – Перший покрасний робот від фірми «Trallfa»

У кінці 60х років під керівництвом Daimler-Benz компанія KUKA почала розробку своїх роботів для зварювальних процесів. Ці роботи почали набирати шалену популярність після вдалого контракту з Мерседесом, що декілька видів роботів почали захоплювати світ. Перший робот з комп’ютерним зором з’явився в 1972 році під назвою «SHAKEY» завдяки комп’ютеру PDP-10, що був на борті зі самим роботом, він міг переміщатись по кімнаті самостійно.

Цей робот (рис. 1.17) за допомогою машинного навчання зміг розпізнавати об’єкти за їх кутами, тобто він продивлявся об’єкт з великою швидкістю та на опорних точках створював візуалізацію об’єкта перед та збоку себе.

Рисунок 1.17 – SHAKEY, перший автономний, інтелектуальний робот

Через рік відбувся вибух у розробці маніпулятор, коли KUKA створила перший 6-ти осьовий та електрично механічно пересувальні осі на стаціонарному роботі (рис. 1.18).

Рисунок 1.18 – KUKA Famulus – перший робот з шістьма електромеханічними вісями

У 1974 році був закладений початковий принцип керування робота-маніпулятора за допомогою допоміжного приладу – комп’ютера, цей робот отримав назву T3, що перекладалось як «The Tomorrow Tool» (Завтрашній інструмент). Так воно й склалось, що починаючи з 1974 року, всі виробництва почали успішну працю над встановленням автоматизованих ліній на виробництві за для звеличення безпеки, рентабельності та зменшення витрат.

Наступним проривним моментом був 1999 рік, коли KUKA встановила перший дистанцій центр для діагностики своїх роботів

1.7 Типи роботів

На сьогодні ми маємо велику кількість роботів, що були створені під свої певні задачі та умови використання, але багато із цих типів роботів були експериментально невдалими, тому маємо (рис. 1.19):

- лінійні (Картезіанські) роботи;
- циліндричні роботи;
- шарнірно-зчленовані (Антропоморфні) роботи;
- дельта-роботи;
- SCARA роботи.

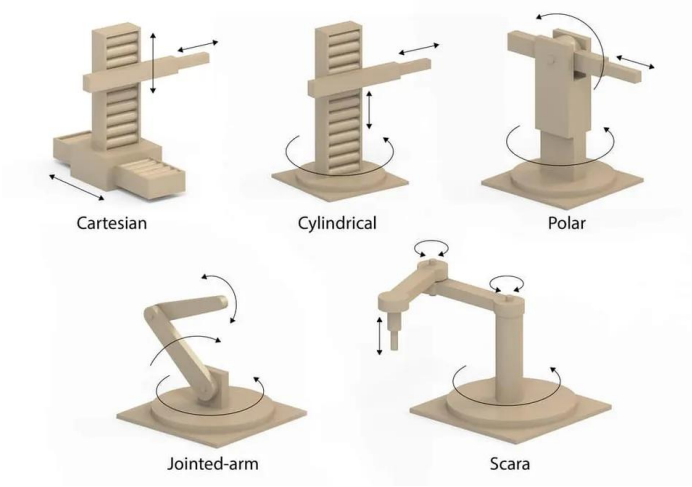


Рисунок 1.19 – Типи роботів

Кожен із цих типів має свою певну перевагу за даними критеріями:

– завдання, що має бути виконаним;
– вільне місце для його роботи;
– доступніть до об’єкта (до місця вивантаження);
– ціна;
– технічна підтримка.

Для систем сортування використовують SCARA-роботи, бо вони є універсальними за своїми властивостями управління, підлягають великій модернізації та додаванням додаткових модулей для детального управління. Основними постачальниками цих роботів у Європі є KUKA та ABB.

1.8 Висновки за розділом 1

Проаналізувавши технічний зір та роботів-маніпуляторів, було зроблено висновок, що OpenCV, тобто технічний зір, з використанням камери є найбільш рентабільним у зв’язку з цінами на інші варіанти ідентифікації об’єктів. SCADA-робот є найбільш функціональним для виконання даного завдання завдяки можливості його модернізації та гнучкості налаштування кожної вісі.

2 РОЗРОБКА СИМУЛЯЦІЙНОГО МАКЕТА В COPELLIASIM ТА ОКРЕМОГО МОДУЛЯ ІДЕНТИФІКАЦІЇ ОБ'ЄКТІВ ЗА ДОПОМОГОЮ ТЕХНІЧНОГО ЗОРУ

2.1 Інструментальні засоби створення симуляційних макетів

На сьогодні існує велика кількість програмних забезпечень для симуляцій проєктів, що спрямовує економію коштів та підвищення безпеки. Й системи автоматизації не є виключенням, бо ті ж самі ПЛС, роботи, програми мають бути напочатку протестовані у безпечному середовищі, тобто у симуляції, а потім вже реалізовані в реальному житті. Тому для реалізацій симуляцій роботів-маніпуляторів існує:

- RoboDK;
- AutoCAD;
- CoppeliaSim.

RoboDK – проривна програма у створеннях симуляційних макетів з можливістю моделювання повних ліній, інтеграцією CAD/CAM є автоматичний генератор коду для роботів, тобто є можливість змоделювати все у забезпеченні й "bolt-on" перенести в життя. У програму вбудовані всеможливі генератори кінематики або динаміки, що спрощує систему розрахунку кінематики або переміщення кожної віхи. Програма підтримує 2 мови програмування LUA та Python та MATLAB, що дає можливість створювати окремі скрипти для виконання певної роботи роботом, використовувати бібліотеки комп'ютерного зору, кінематики таймерів або можливість виведення даних на сервер для подальшої обробки або моніторингу. Сама програма має велику кількість готових проєктів, готових моделей роботів та інструментів.

AutoCAD – є однією з найпотужніших та найпоширеніших САПР-систем, яку розробляє Autodesk. У даному ПЗ можливо робити 2D та 3D

проєкти та документацію. Має гарну інтеграцію з іншими системи завдяки BIM, CAM або GIS, тобто можливість передачі документів напряму до ЧПУ має велику функціональну базу для створення, перетворення та удосконалення об’єкта чи системи. Сам розробник Autodesk пропонує офіційні курси, книги та онлайн-платформи для навчання використанням даним ПЗ.

CoppeliaSim (раніше V-REP) – універсальний робот-симулятор, що підтримує такі мови програмування як C/C++, Python, Java, MATLAB, Urbi. Має велику базу основних роботів-маніпуляторів. Має велику кількість плагінів на C/C++, що розширюють функціональність, можливість створення інтерфейсів до апаратури чи ROS. Є RemoteAPI для зовнішніх клієнтів для керування та обміну даними. Має велику кількість фізичних движків, власний движок для прямої та зворотної кінематики, можливість уникнення колізії та розрахування мінімальної відстані, вбудованість графіків, реальна візуалізація сенсорів (камера, LiDAR).

Також важливою частиною розробки проєкту є IDE в котрій буде реалізований код програми. Найпопулярнішим та відкритим IDE є Microsoft Visual Studio Code. ПЗ підтримує велику кількість плагинів та додатків, що були розроблені користувачамі для більш комфортної та інформативної роботи з кодом.

Вибір мови програмування також є важливим моментом. Так як проєкт буде реалізований в CoppeliaSim, що підтримує Lua або Python, то було обрано Python через його велику кількість бібліотек, що облегшують програмування.

2.2 UML-діаграми та схеми відпрацювання коду

Дуже важливою задачею перед створенням макета є задання напрямків руху по проєкту, тобто розробка UML-діаграма класів (рис. 2.1).

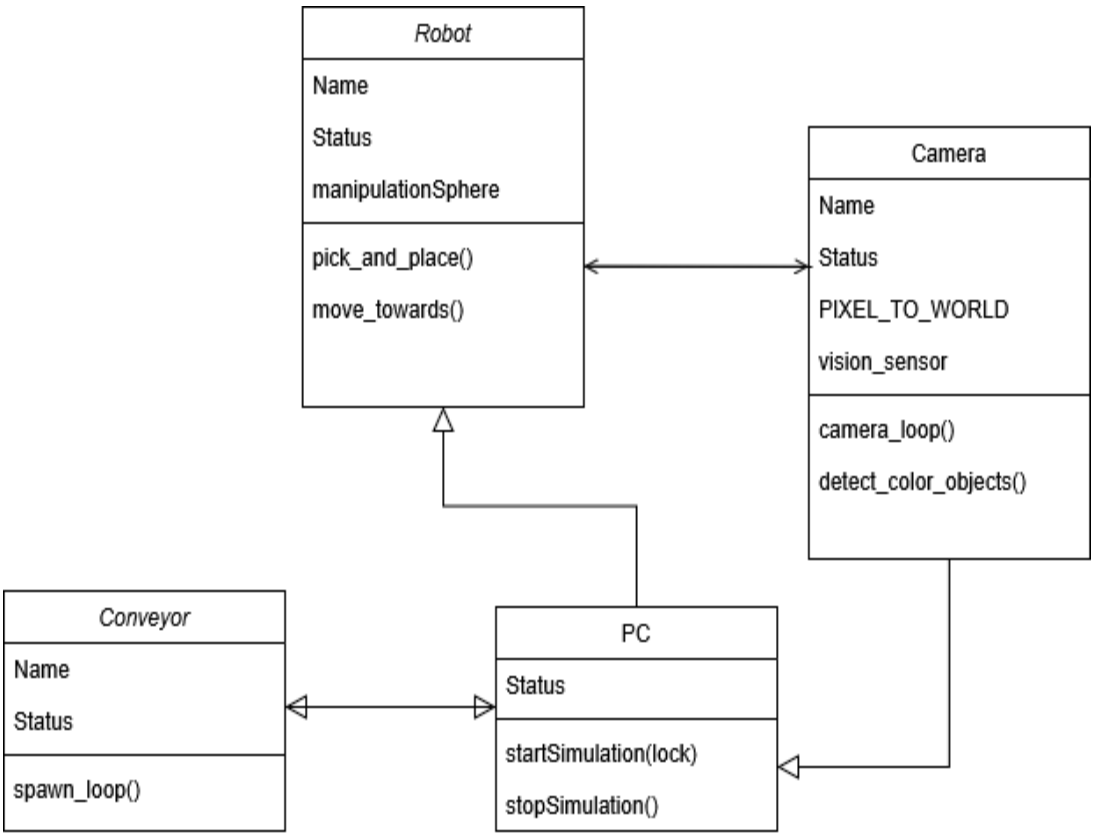


Рисунок 2.1 – UML-діаграма класів

Де читаючи її (рис. 2.1) можна зрозуміти, які функції та основні атрибути нам потрібні для виконання поставленої задачі. Де об’єкт «PC» є основним об’єктом, що керує усією симуляцією, є «Conveyor», що виконує важливу функцію з створення об’єкта на конвеєрі, «Robot» - він виконує роботу з переміщення об’єкта до заданих місць, а «Camera» - є джерелом зору, що показує пересування об’єкта по конвеєру та його підбирання до короба.

Також не мало важливим є UML-діаграма послідовності для розуміння взаємодії кожного об’єкту у проєкті (рис. 2.2). Бачимо, що PC запускає симуляцію та далі вже CoppeliaSim починає працю з Camera, Robot та Conveyor, де Camera відсилає дані до робота та дублює їх до PC. Robot своєю чергою отримує дані від камери для виконування своїх дій, а саме для виконання функцій move_toward та pick_and_place. Конвеєр же відправляє дані до PC, що об’єкт був створений для детальних логів симуляції.

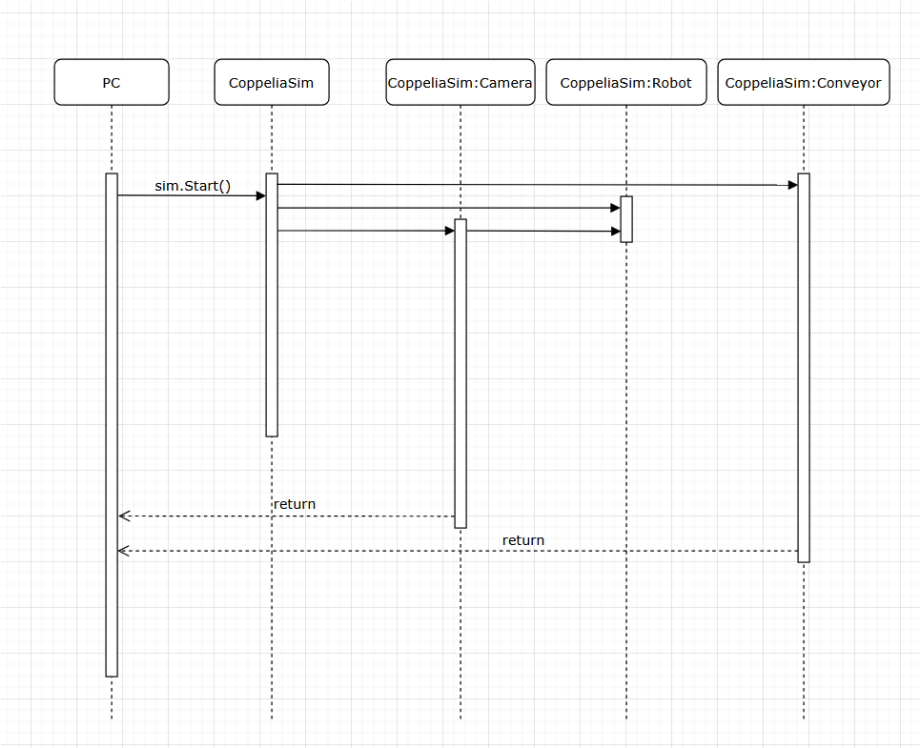


Рисунок 2.2 – UML-діаграма послідовності

Обирання системи ідентифікації об’єктів за допомогою технічного зору є важливим кроком у розробці усієї системи, тому вибір є між:

– Yolo;
– SSD MobileNet V2;
– OpenCV-Python.

Всі системи є великим дата-банком зображень різних об’єктів, що є вже відсортованими, що спрощує майбутню ідентифікацію.

2.3 Теорія автоматичного управління

Система складається з таких основних компонетів, як:

– об’єкт управління – роботизований маніпулятор, що виконує фізичне пересування об’єкту;
– датчик - камера;
– керуючий пристрій – комп’ютер на котрому встановлений Python,

який реалізує алгоритм управління;

– виконавчі елементи – серводвигуни та приводи робота та серводвигун конвеєрної лінії.

Для підвищення точності у проєкту реалізовано замкнену систему зворотнього зв'язку, де сигнал з камеру обробляється виконавчим елементов та на основі цього сигналу надаються команди роботу для фізичного пересування у площині.

Розрахуємо ПІД-регулятор для осі маніпулятора (2.1).

$$u(t) = K_p e(t) + K \int e(t)dt + K_d \cdot \frac{de(t)}{dt} \quad (2.1)$$

Де візьмемо, що відстань до об'єкту від камери становить 0.35м. Та візьмемо похибку у 0.05м.

$$e(t) = 0.35 - 0.05 = 0.30\text{м} \quad (2.2)$$

Для простоти візьмемо коефіцієнти пропорційності, інтегральності та диференціальної дії відповідно 100, 10 та 5, тоді нехай $\int e(t)dt = 0.1$, а $de(t)/dt = 0.02$, тоді (2.3):

$$u(t) = 100 \cdot 0.05 + 10 \cdot 0.1 + 5 \cdot 0.02 = 61 \quad (2.3)$$

Для оцінки енерговитрат приводів візьмемо формулу (2.4):

$$P = \frac{T \cdot \omega}{\eta} \quad (2.4)$$

де T – момент на вісі двигуна у Нм (середній момент становить 5,6 Нм); ω – кутова швидкість, що у IBB ARB 140 становить 2 рад/с та коефіцієнт корисної дії у 0.85, це є найбільш стандартизованим ККД, тоді

$$P = \frac{5.6 \cdot 2}{0.85} = 13.177 \text{ Вт}$$

Середній цикл виконання пересування об’єкта складає 8 секунд, тоді за одну ітерацію робот витрачає 105 Дж (2.5):

$$E = P \cdot t = 13.177 \cdot 8 = 105 \text{ Дж} \quad (2.5)$$

Враховуючи, що система виконує 1 ітерацію раз у 30 секунд, тоді за годину система виконує 120 ітерацій, загальний час праці робота становить 16 хвилин за году, тому робот витрачає 4,23 кВт/годину.

Це вказує на те, що система управління є енергоефективною при відповідному налаштуванні регуляторів.

2.4 Висновки до розділу 2

Отже, підбиваючи висновки, можна сказати, що CoppeliaSim є гарною песочницею для виконання симуляційних макетів для автоматизованих ліній, UML-діаграми відіграють велику роль у розумінні праці роботи системи та вибір правильного технічного зору прищвидшує виконання розробки. Розрахунок ПІД-регулятора дає розуміння, що система енергоефективною та підходить під наші умови використання.

3 РЕАЛІЗАЦІЯ КАМЕРИ ТА СИМУЛЯЦІЙНОГО МАКЕТА В COPELLIASIM

Для розробки симуляційного макета була використана CoppeliaSim Edu ver.4.9.0 (rev.6), Microsoft Visual Studio та Blender для створення 3D-моделей. Сама CoppeliaSim вже розташовує вбудованим ZMQ remoteAPI, що розташовується за адресою 23000. Була обрана ZMQ через те, що цей додаток дозволяє керувати сами проєктом та самою симуляцією, коли у той час WebSocket розташовує лише Back-end керування.

Для самого макету були обрані такі об'єкти:

– ABB IRB 140 (робот-маніпулятор);
– вбудована камера з CoppeliaSim;
– само розроблені контейнери для сміття;
– куб, сфера, циліндр, як об’єкти симуляції сміття;
– конвеєрна лінія.

На робота також був доданий вбудований двигун інвертованої кінематики для побудови пересування роботу. Тому ієрархія проєкту виглядає як зазначено на рисунку 3.1.

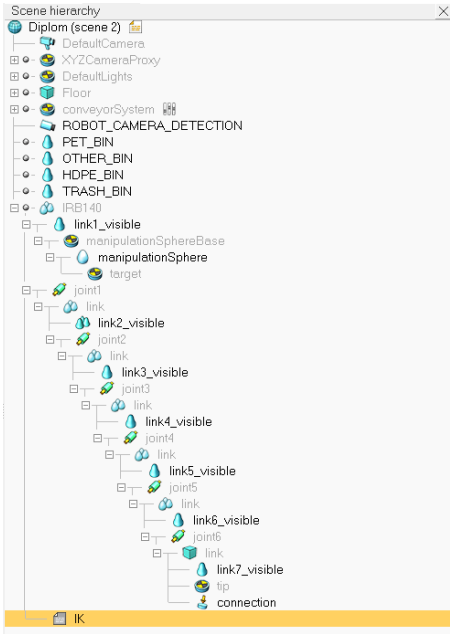


Рисунок 3.1 – Ієрархія проєкту

Далі було розпочато писання коду, де було треба внести правки до структури проєкту, а саме перейменувати кожну вісь робота з додаванням її номеру, тобто за принципом «joint{i+1}». На початку нашого коду треба додати всі необхідні бібліотеки: coppeliasim_zmqremoteapi_client для підключення до ПЗ, випадкових значень, багатопоточності, роботи з часом, обробки числових даних (NumPy) та зображень (OpenCV).

Далі встановлюємо підключення до CoppeliaSim через ZMQ API та захист від одночасного доступу в багатопотоковому режимі (рис. 3.2).

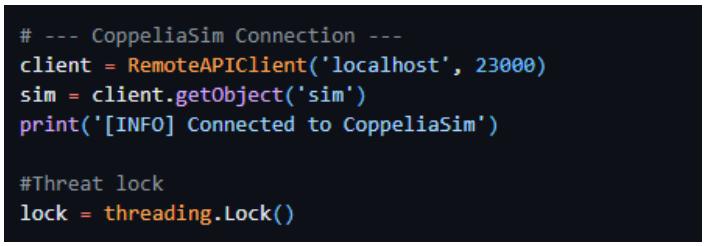


Рисунок 3.2 – Підключення за допомогою ZMQ API та захист в багатопотоковому режимі

Також була додана строка коду, де можна бачити, що з’єднання було успішно виконане. Після цього ми маємо задати наші початкові позиції, константи для проєкту та технічного зору (рис. 3.3).

```
# Handles
manipulation_sphere = sim.getObject('/IRB140/manipulationSphere')
vision_sensor       = sim.getObject('/ROBOT_CAMERA_DETECTION')

# Home pose
HOME_POS = [-0.320, 0.210, 0.620]
HOME_ORI = [0, 0, -90]
with lock:
    sim.setObjectPosition(manipulation_sphere, sim.handle_world, HOME_POS)
    sim.setObjectOrientation(manipulation_sphere, sim.handle_world, HOME_ORI)

# Constants
PIXEL_TO_WORLD = 0.0005  # meters per pixel
data_params = {
    'MAX_SPEED':    0.5,     # m/s
    'HEIGHT_OFFSET':0.2,     # m above object
    'FRAME_SLEEP':  0.05,    # s between steps
    'GRASP_OFFSET': 0.05     # m above object to grasp
}

# Primitives and color HSV ranges
shape_constants = {
    'Cuboid':   sim.primitiveshape_cuboid,
    'Cylinder': sim.primitiveshape_cylinder,
    'Sphere':   sim.primitiveshape_spheroid
}
# only pick these colors (trash/gray excluded)
color_ranges = {
    'Green':  ((50,100,100),(70,255,255)),
    'Yellow': ((20,100,100),(30,255,255)),
    'Blue':   ((100,100,100),(130,255,255))
}

# Bins for each color
bin_positions = {
    'Green':  [0.325, 1.250, 0.450],
    'Yellow': [-0.025, 1.250, 0.450],
    'Blue':   [-0.900, 1.250, 0.450]
}
```

Рисунок 3.3 – Початкові точки, константи для проєкту та технічного зору

Після базових підключень та змінних можна почати додавання функцій, що необхідні для виконання поставленої задачі, а саме:

– появлення об’єкта сортування (spawn_loop);
– пересування робота до об’єкта (move_towards);
– pick-and-place (pick_and_place);
– вивід камери (camera_loop).

Появлення об’єктів з використанням RemoteAPI виконується лише у тих випадках, коли маємо якісь інструкції від розробки ПЗ, або розробника бібліотеки для RemoteAPI. Також нам треба симуляція та оптимізований проєкт, тому було обрано рішення створювати прості об’єкти з різними кольорами (рис. 3.4), для симуляції пластикових пляшок для того, щоб можна було отримати гарну частоту кадрів, яка дозволить більш детальну перевірку виконання програми, бо як побачимо далі бібліотека OpenCV у поєднанні з

виводом камери потребує багато ресурсів комп’ютера.

```python
# Spawn loop: create objects periodically, track only non-gray
# and map handles to their color
spawned = []
shape_color_map = {}
def spawn_loop():
    while True:
        # choose shape and random color including gray
        name, prim = random.choice(list(shape_constants.items()))
        colors = {
            'Green': (0,1,0),
            'Yellow': (1,1,0),
            'Blue': (0,0,1),
            'Gray': (0.5,0.5,0.5)
        }
        color_name, color_rgb = random.choice(list(colors.items()))
        try:
            with lock:
                h = sim.createPrimitiveShape(prim, [0.15]*3, 0)
        except Exception as e:
            print(f"[WARN] spawn_loop: {e}, retrying...")
            time.sleep(0.5)
            continue
        if h != -1:
            with lock:
                sim.setObjectPosition(h, sim.handle_world, [0.225,-0.55,0.54])
                sim.setObjectInt32Param(h, sim.shapeintparam_static, 0)
                sim.setObjectSpecialProperty(h,
                    sim.objectspecialproperty_collidable|
                    sim.objectspecialproperty_measurable)
                sim.setObjectInt32Param(h, sim.shapeintparam_respondable, 1)
                sim.setShapeColor(h, None, sim.colorcomponent_ambient_diffuse, color_rgb)
            # track only if not gray
            if color_name != 'Gray':
                spawned.append(h)
                shape_color_map[h] = color_name
            print(f"[INFO] Spawned {name} with color {color_name} (handle={h})")
        time.sleep(20)
```

Рисунок 3.4 – Функція появлення об'єкта сортування (spawn_loop)

У коді (рис. 3.4) були задані певні змінні для кольору об’єкта та де буде з’являтись об’єкт для сортування. Так само у консоль (рис. 3.5) виводиться основна інформація об об’єкту, що він був створений, який має колір, та handle-номер у програмі для можливого налагодження програми при появленні помилок.

```
PS C:\Users\Dmytro Savenko\Desktop\УНИВЕР\diplom\CopelliaSim> & 'c:\Python312\python.exe' 'c:\Users\Dmytro Savenko\.vscode\extensions\ms-python.debugpy-2025.8.0-win32-x64\bundled\libs\debugpy\launcher' '5819
' '--' 'C:\Users\Dmytro Savenko\Desktop\УНИВЕР\diplom\CopelliaSim\connect_test.py'
[INFO] Connected to CoppeliaSim
[INFO] Spawned Sphere with color Green (handle=467)
[INFO] Spawned Cylinder with color Blue (handle=468)
[INFO] Spawned Cylinder with color Yellow (handle=469)
[INFO] Spawned Sphere with color Gray (handle=470)
[INFO] Spawned Cylinder with color Green (handle=471)
[INFO] Spawned Cylinder with color Blue (handle=472)
[INFO] Spawned Cylinder with color Blue (handle=473)
[INFO] Spawned Cylinder with color Yellow (handle=474)
[INFO] Spawned Sphere with color Gray (handle=475)
[INFO] Spawned Cuboid with color Gray (handle=476)
[INFO] Spawned Cylinder with color Green (handle=477)
[INFO] Spawned Cylinder with color Green (handle=478)
[INFO] Spawned Cylinder with color Green (handle=479)
```

Рисунок 3.5 – Вид консолі при роботі програми

Через те, що CoppeliaSim розташовує у собі інверсивну кінематику, то можемо використовувати лише manipulationSphere (імітує пневматичну вакуумну присоску) (рис. 3.6) для пересування робота, а вже інші вісі будуть

рухатись за цією сферою завдяки вбудованій функції інверсивної кінематики (ІК) (рис. 3.6).

```
# Move sphere toward target at speed scaled by dt target at speed scaled by dt
def move_towards(target, dt):
    with lock:
        pos = sim.getObjectPosition(manipulation_sphere, sim.handle_world)
    vec = np.array(target) - np.array(pos)
    dist = np.linalg.norm(vec)
    if dist < 1e-3:
        return
    step = min(data_params['MAX_SPEED']*dt, dist)
    newp = (np.array(pos) + vec/dist*step).tolist()
    with lock:
        sim.setObjectPosition(manipulation_sphere, sim.handle_world, newp)
```

Рисунок 3.6 – Пересування робота до об’єкта сортування

Далі за списком йде сама велика та масивна функція під назвою pick-and-place, бо вона всередині себе має такі задачі як:

– захват об’єкт;
– повернення до початкової точки;
– пересування до потрібного кошику;
– відпустити об’єкт;
– повернення на домашню позицію.

Сама функція захвату в CoppeliaSim виконується дуже простим способом, зі залежністю батьки-діти в ієрархії проєкту, але через те, що маємо динамичні об’єкти, що рухаються у сцені, маємо передавати захопленому об’єкту статичний параметр для його успішного пересування, тому функція захвату виглядає саме так (рис. 3.7).

Функція повернення на домашню позицію після захоплення об’єкта також реалізована зі задаванням домашньої позиції для робота та невеличкими паузами для стабілізації системи (рис. 3.7).

Функція переміщення до короба виконана з невеликим підвищенням по Z-вісі для уникнення зіткнень, де при збігу всіх даних, об’єкт буде переміщений до правильного короба та над ним вже буде відпущенним із

поверненням фізики об’єкта (рис 3.7).

```
# attach
with lock:
    sim.setObjectInt32Param(obj, sim.shapeintparam_static, 1)
    sim.setObjectParent(obj, manipulation_sphere, True)
# return home with object
while True:
    with lock:
        sp = sim.getObjectPosition(manipulation_sphere, sim.handle_world)
    if np.linalg.norm(np.array(sp)-np.array(HOME_POS))<0.01:
        break
    move_towards(HOME_POS, data_params['FRAME_SLEEP'])
    time.sleep(data_params['FRAME_SLEEP'])
# move to bin
binp = bin_positions[color]
target_bin = [binp[0], binp[1], binp[2]+data_params['GRASP_OFFSET']]
while True:
    with lock:
        sp = sim.getObjectPosition(manipulation_sphere, sim.handle_world)
    if np.linalg.norm(np.array(sp)-np.array(target_bin))<0.01:
        break
    move_towards(target_bin, data_params['FRAME_SLEEP'])
    time.sleep(data_params['FRAME_SLEEP'])
    # release
with lock:
    sim.setObjectParent(obj, -1, True)
    sim.setObjectInt32Param(obj, sim.shapeintparam_static, 0)
# intermediate return home
```

Рис. 3.7 – Pick-and-place функція

Реалізація виводу картинки з камери була, що показує, як відбувається комунікація робота з об’єктом за допомогою камери (рис. 3.8), котра робить список кандидатів, та при збіганні даних, буде система передавати дані до робота, щоб він виконав команду pick-and-place (рис. 3.9).

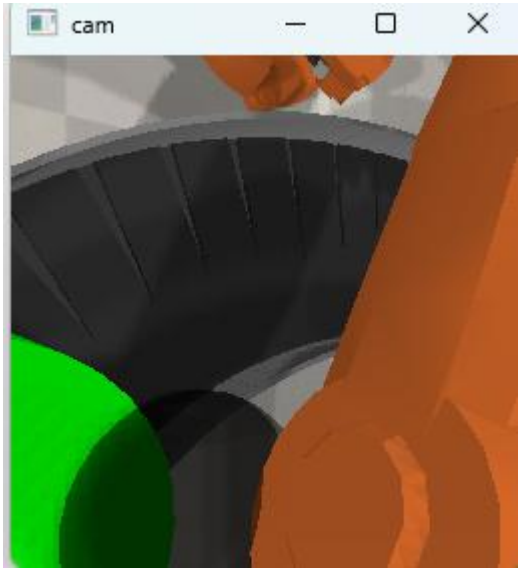


Рисунок 3.8 – Вивід камери комп’ютерного зору

```python
def camera_loop():
    prev = time.time()
    while True:
        now = time.time(); dt = now-prev; prev=now
        try:
            with lock:
                img_bytes, res = sim.getVisionSensorImg(vision_sensor)
        except:
            time.sleep(0.1)
            continue
        img = np.frombuffer(img_bytes,dtype=np.uint8).reshape((res[1],res[0],3))
        bgr = cv2.cvtColor(img,cv2.COLOR_RGB2BGR)
        hsv = cv2.cvtColor(bgr,cv2.COLOR_BGR2HSV)

        # build list of candidates
        cands=[]
        for color,(lo,hi) in color_ranges.items():
            mask=cv2.inRange(hsv,np.array(lo),np.array(hi))
            cnts,_=cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
            for c in cnts:
                M=cv2.moments(c)
                if M['m00']==0: continue
                cx=int(M['m10']/M['m00']); cy=int(M['m01']/M['m00'])
                # approximate world xy
                dx=(cx-res[0]//2)*PIXEL_TO_WORLD
                dy=-(cy-res[1]//2)*PIXEL_TO_WORLD
                approx=[HOME_POS[0]+dx,HOME_POS[1]+dy]
                if not spawned: continue
                # nearest obj handle
                obj=min(spawned, key=lambda h: np.linalg.norm(np.array(sim.getObjectPosition(h,sim.handle_world)[:2])-np.array(approx)))
                op=sim.getObjectPosition(obj,sim.handle_world)
                tgt=[op[0],op[1],op[2]+data_params['GRASP_OFFSET']]
                cands.append((color,c,(cx,cy),tgt,obj))
        if cands:
            # pick best by image-center distance
            best=min(cands,key=lambda x:abs(x[2][0]-res[0]//2)+abs(x[2][1]-res[1]//2))
            color,cnt,cent,tgt,obj=best
            move_towards(tgt,dt)
            if np.linalg.norm(np.array(sim.getObjectPosition(manipulation_sphere,sim.handle_world))-np.array(tgt))<0.02:
                pick_and_place(obj,color)
                spawned.remove(obj)
        else:
            move_towards(HOME_POS,dt)

        cv2.imshow('cam',bgr)
        if cv2.waitKey(1)&0xFF==ord('q'): break
    cv2.destroyAllWindows()
```

Рисунок 3.9 – Вивід камери (camera_loop)

3.2 Розрахунки ефективності

Для проведення тестів симуляційного макета було обрано часові інтервали: 5 хв., 10 хв., 15 хв., та 1 год. Тому можемо бачити результати такі як:

$$N = \frac{t*60}{S_{time}} = \frac{60*60}{30} = 120 \text{ об'єктів} \quad (3.1)$$

де t – це є тестувальний час, S_{time} – час створення об'єкта. Час у 30 секунд був обраний через те, що при великих розрахунках ПЗ не може стабільно передавати через RemoteAPI необхідну кількість даних для стабільної роботи технічного зору, тому маємо час у 30 секунд є оптимальним для стабільної роботи проєкту.

Також важливим є розрахунок коштів які треба для встановлення даної системи під будівлею та кількість заощаджених коштів при даному методі сортування.

Для початку розрахунків нам треба обрати автомобіль яким це буде перевезено до переробної станції, візьмемо стандартний автомобіль довжиною 10 метрів, висотою у 3 метри та 2,20 метра ширини, тому маємо об’єм, ще маємо враховувати об’єм, що стоїть на задніх колесах тягача, тому маємо (рис 3.10).

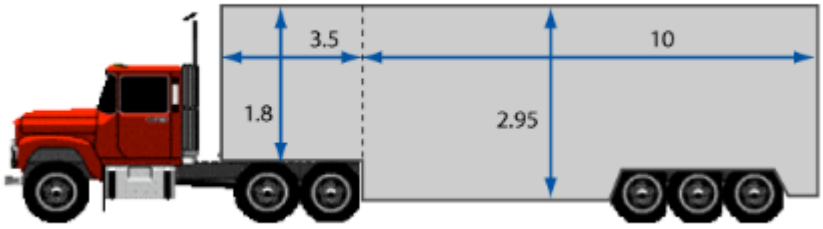


Рисунок 3.10 – Вигляд тагяча із розмірами причепу

$$V = (1.8 * 3.5 * 2) + (2.95 * 10 * 2) = 71.6m^3 \quad (3.2)$$

Це є загальний об'єм причепу, але маємо враховувати те, що сировина не буде спресована, тому орієнтуючись на [22] візьмемо число у 28% наповнення причепу, тому маємо (3.2) відняти 28%, та отримуємо

$$V_{чист} = V - 28\% = 71.6 - 28\% = 51.5m^3 \quad (3.3)$$

одна пляшка важить ~30 грамів за штуку та має висоту у 20 см та діаметр у 6,35 см. Тому об’єм однієї пляшки становить, сама пляшка це є складної форми циліндр, тому можемо взяти циліндр за основу пляшки, тому маємо, але для початку знайдемо радіус з діаметра, де 2r = d, тому r = 2/d = 6,35/2 = 3.175

$$V_{\text{пляшка}} = \pi R^2 h = \pi * 3.175^2 * 20 = 629.03\, cm^3 \quad (3.4)$$

далі переводимо $см^3$ в $м^3$ та отримуємо, що 1 пляшка це 0,000629 $м^3$, тому загальна кількість пляшок, що може вміститись у причеп є 81.875 пляшок за 1 причеп. Візьмемо те, що від будинку до переробної станції треба проїхати 100 км, тягач потребує 27 л/100 км дизельного пального. 1 літр дизелю це є ~38MJ енергії, тому за формулою (3.5) можемо винайти скільки потребує енергії 1 перевезення пластикових виробів до переробного заводу.

$$E_{\text{тягач}} = \frac{L_{avg} * E_{diesel}}{N} = \frac{27*(38*10^6)}{81.875} = 1{,}25 * 10^7 J = 12{,}5MJ \quad (3.5)$$

при прибутті на місце тягач висипає пляшки до дробильника, що переробляє за час 1500 пляшок та має мотори 7,5kW, тому він потребує27MJ / годину, що при його годинній праці він потребує приблизно 18kJ енергії (3.6).

$$E_{\text{подріб}} = \frac{27MJ}{1500\ \text{пляшок}} \approx 18kJ \quad (3.6)$$

Пластик потребує своєї температури плавлення, що становить ~220 ^oC, та має удільну тепломісткість, що становить 1446 $kg^{-1}K^{-1}$ та маса 1 пляшки, що становить 30 грамів. Тому лише для перетоплення (3.6) та розігрівання печі необхідно

$$E_{\text{перетопл}} = T * C_p * m = 220 * 1446 * 0.03 = 9.54kJ \qquad (3.7)$$

Також треба враховувати, що 1 пляшка для перетоплення потребує 125J, тому необхідно 9,67kJ енергії для 1 пляшки.

Оскільки зазвичай перероблювальний завод працює за принципом сортування всіх видів пластикових матеріалів (рис. 3.11).

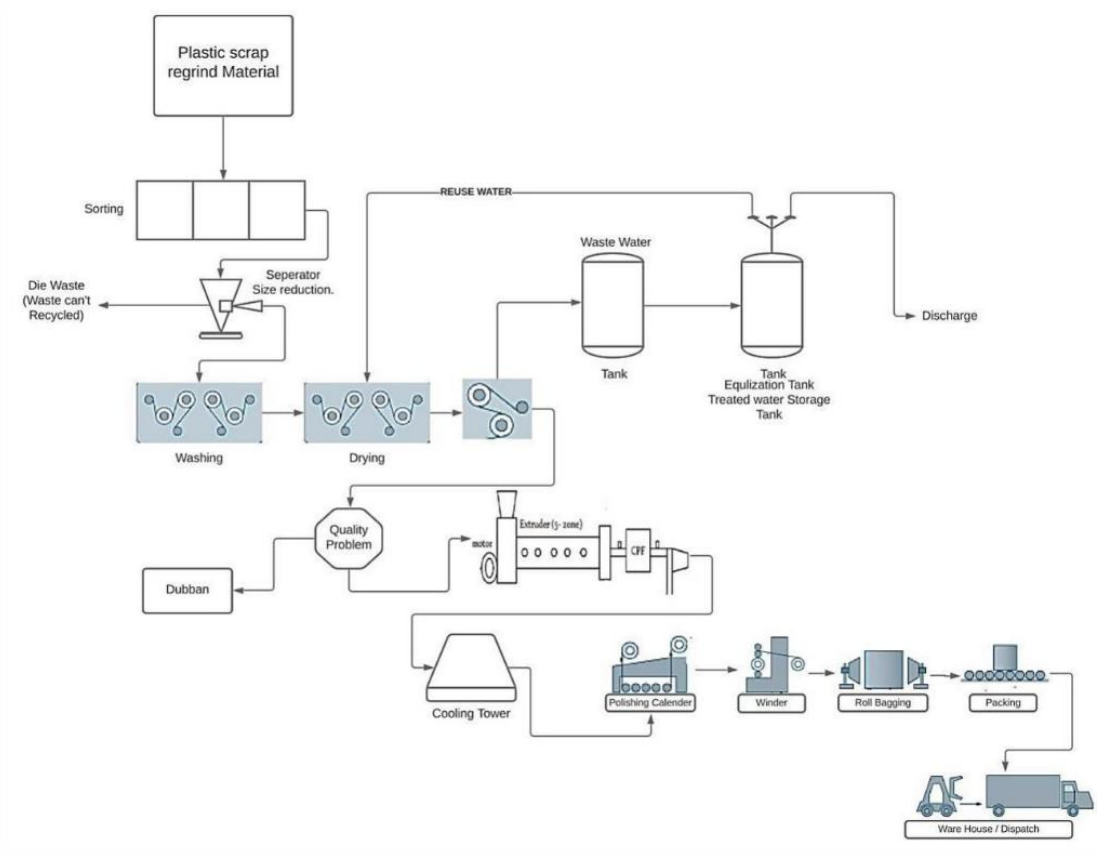


Рисунок 3.11 – Схема перероблювального заводу [13]

Сортування на даних підприємствах відбувається все ще завдяки механічній силі, тобто людської праці, що викликає не лише загрозу життю працівників, а й знижує ефективність. Тому завдяки невеликій системі сортування під будівлями можемо зменшити ризики небезпеки, підвищити ККД перероблювальних заводів, тим що буде лише сортування за розмірами пластикових виробів. Це підвищить не лише ККД підприємства, а й зменшить навантаження на електричну систему, бо якщо рахувати систему

сортування в будівлях, то маємо, що:

– конвеєрна лінія при навантаженні у 2 тони та завдовжки 30 метрів потребує приблизно 3 HP, 1HP це є 0.746 кВт, тому конвеєрна лінія потребує 2.238 кВт/год [24];

– робот ABB IRB 140 потребує без навантаження ~200 – 300 Вт, при середньому навантаженні ~500 – 800 Вт, й будемо використовувати середнє навантаження за для того, щоб не було зайвого використання ресурсу робота на пікових навантаженнях та збільшити строк його життя;

– комп’ютер, що буде виконувати обчислювальні процеси потребує приблизно 200 Вт.

Тому загальна необхідна затрата електроенергії становить від 2.9 кВт до 3.2 кВт на годину, що є гарним показником, враховуючи, що сортуємо не одну сировину, а одразу 3 види пластику та харчове сміття у разі його потрапляння у систему сортування.

Британська компанія [25] провела аналіз того, скільки 1 британська людина викидує пластику, й ця цифра є дуже вразливою, бо 1 людину населення це становить 76 кг пластику, що у 20 разів більше ніж 50 років тому.

Визначаємо, що житловий будинок приймає 20 сімей, це є приблизно 50 людей, тому 1 житловий будинок викидує до 4 т сміття у рік, у день це становить 10 кг сміття. За тиждень дана система може сортувати до 100 кг пластику, що становить приблизно 3300 пластикових виробів при їхній середній вазі у 30 грамів.

Тому якщо взяти дану систему у житловому будинку з 50 мешканцями, то можемо зрозуміти, що даної системи буде достатньо, тому що 1 людина викидує до 8 грамів сміття на годину (3.5), що при даному житловому будинку дорівнює 400 грамів на годину, а це є до 14 пластикових виробів на годину, тому дана система, орієнтуючись на формулу (3.8) буде виконувати свої обов'язки на 100%.

$$N_{буд} = \frac{\frac{76}{365}}{24} = 0.008 \text{ кг/год} \quad (3.8)$$

3.2 Тестування системи

Тестування системи відбувалось у різних режимах та розумінням помилок, що були допущені на моменті розробки, а саме, що API в CoppeliaSim не є таким стабільним як кажеться та вміння працювання ПЗ із зовнішнім виконувачем скрипту може викликати проблеми з оптимізацією. А саме, при виконанні скрипту у момент функції Pick and Place ми можемо бачити великі проблеми з виводом зображення на зовнішнє вікно камери, бо у цей момент API виконує закриття порту, та вже після виконання функції та перезавантаження бачимо, що далі все йде як під час старту, тому було прийнято рішення змінити час появи нового об’єкта на 30 секунд, бо API потребує від 15 до 20 секунд на перезавантаження. Тому для оптимізації були прийняті такі заходи:

– зменшення кадрів на секунду у камері до 5, щоб було меншим навантаження на систему;

– збільшений інтервал для появлення нового об’єкта;

– спрощена система пересування робота за допомогою задавання спеціальних точок для його пересування.

Завдяки цим крокам була отримана задовільна продуктивність проєкту для коректної праці (рис. 3.12).

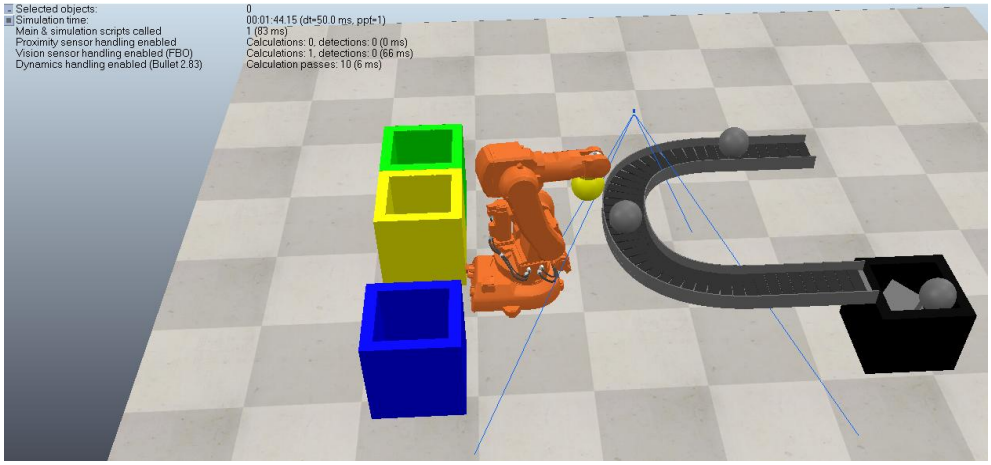


Рисунок 3.12 – Момент пересування об’єкта до жовтого короба

Також проведений тест при моменті, коли камера бачить 2 або більше об’єктів та правильному виборці об’єкта сортування (рис. 3.13).

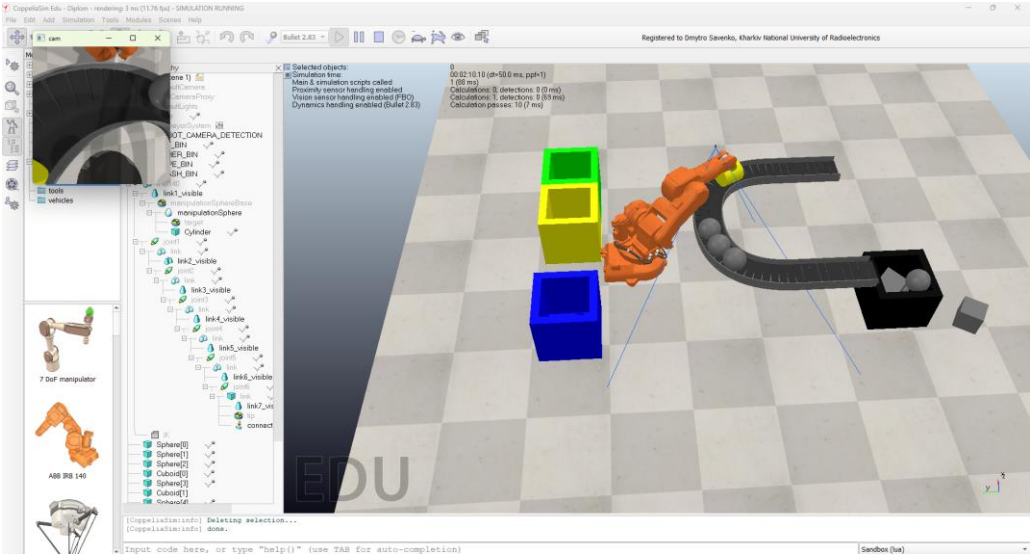


Рисунок 3.13 – Вибірка жовтого об’єкта сортування серед інших об’єктів сортування

Також у терміналі Visual Studio Code ми можемо бачити які об’єкти були створені та при на помилки звірити з тим, що маємо у програмному забезпечені CoppeliaSim (рис. 3.14).

Рисунок 3.14 – Логи у терміналі Visual Studio Code

Також була розроблена окрема камера для ідентифікації пластикових виробів для можливості встановлення до HMI-монітора (рис. 3.15). Сама програма написана на Python, складається з 4 файлів, має українську та англійські мови та має оффлайн-версію Yolo5su для можливості праці без мережі інтернет.

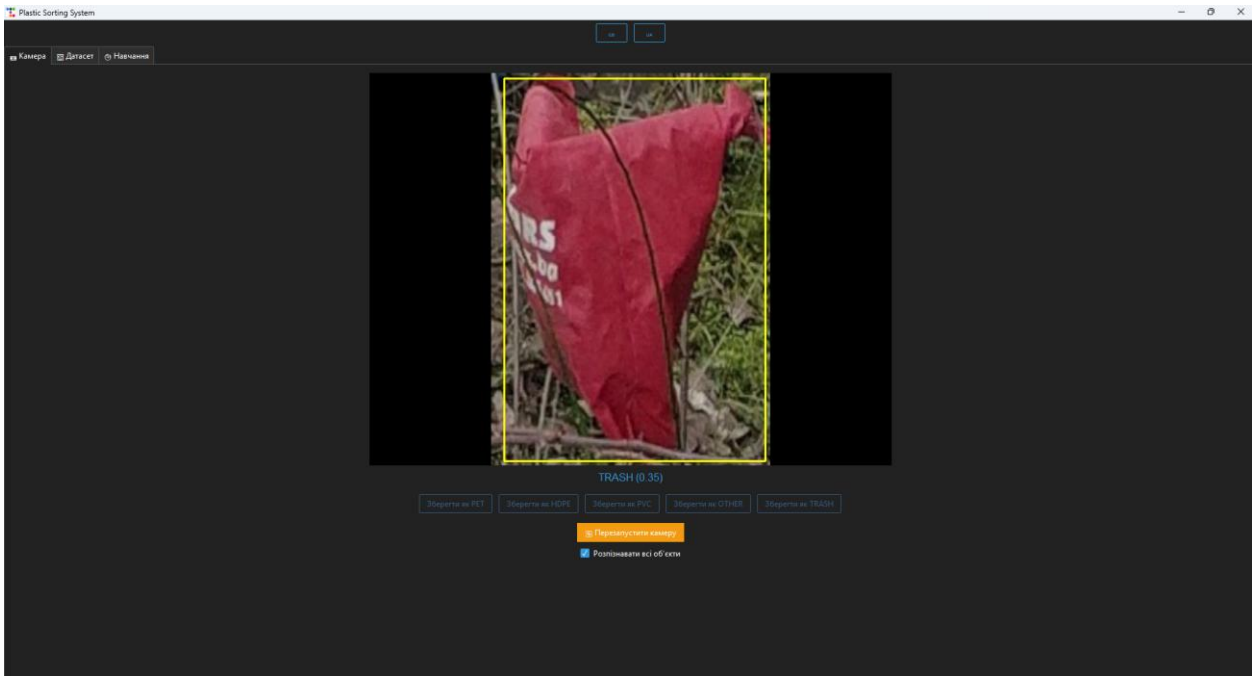


Рисунок 3.15 – Інтерфейс програми

Програма має в собі 3 сторінки для управління систему ідентифікації «Камера» «Датасет» та «Навчання». Завдяки сторінці навчання можливо завантажити дані до системи та запросити самоідентифікацію об’єктів за класами. На жаль це є технічний зір та він не має 100% ефективності в ідентифікації кожного об’єкту та може їх відсортовувати до неправильних класів, тому була додана сторінка «Датасет» для можливості видалення неправильних об’єктів з класу. Для мінімальної праці «Навчання» необхідно мінімум по 1 зображенню в кожному класі, щоб система змогла почати ідентифікувати об’єкти. Чим більше буде об’єктів у класах, тим точніше буде працювати система, також у сторінці «Навчання є логи» для можливості відстежування того, що було зроблено у системі, наприклад «Перенавчити систему ідентифікації», «Перезапуск камери» чи куди були відсортовані об’єкти для ідентифікації.

Сторінка датасет має функцію видаленні зображенні лише по 1 натиску на неї та можливість передивлятись об’єкти, що належати до певного класу (рис. 3.16).

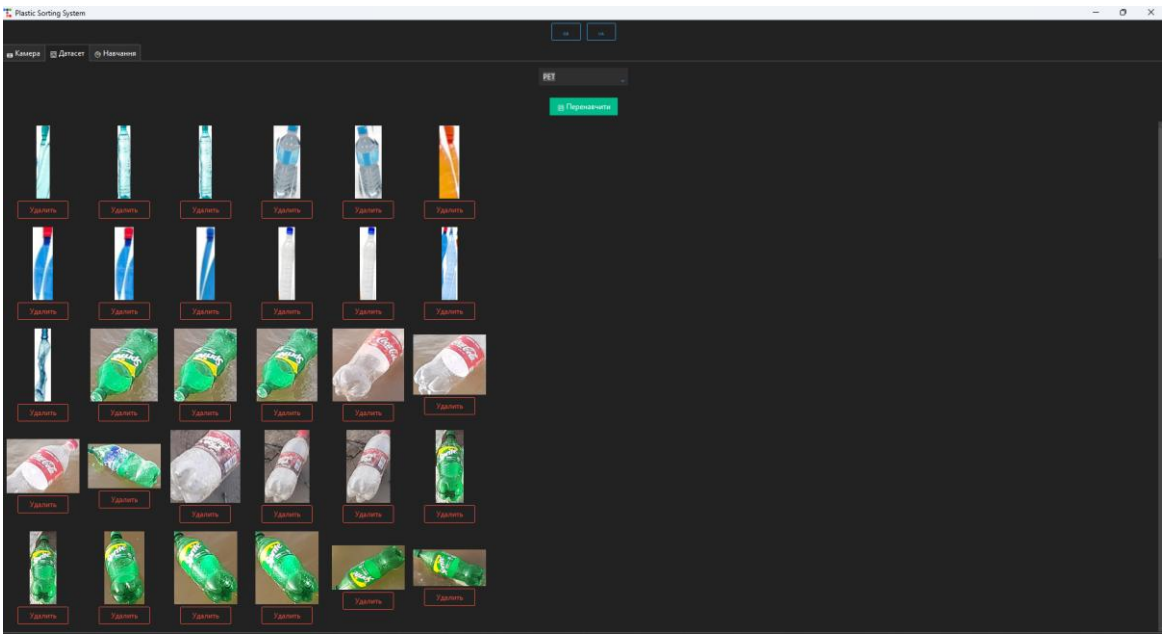


Рисунок 3.16 – Вигляд сторінки «Датасет»

Кожен об’єкт, що не може бути ідентифікованим за допомогою Yolo чи локального датасету, буде ідентифікованим за основними признаками певного об’єкта. Цей спосіб не є 100% точним, але при великому дата-банку можна підвищити точність.

3.3 Охорона праці

Охорона праці як й усюди є важливою частиною нашої безпеки при виконанні даної роботи, тому за для безпеки дана система потребує окремої кімнати для зменшення токсичних запахів та смороду у навколишню середу. Так само необхідні вбудовані перемикачі для повного вимкнення системи, окремих елементів.

Як й будь-яка інша система, дана система не є виключенням та потребує регулярних мастильних робіт, технічного огляду на працездібність системи у критичних навантаженнях та наявності дефектів після певного часу праці.

Як й будь-яка інша система, дана система має певні системи безпеки:

– система автоматичного відключення при некоректному роботі одного з основних агрегатів системи;
– система захисту від короткого замикання;
– система пожаротушіння;
– система вентиляції з фільтрами для зменшення вологи у кімнаті;
– система звукового та світлового сповіщення при поломках.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було здійнено дослідження, метою якого було розробка автоматизованого сортування з використанням робота з 6 вісями вільності. Основний акцент було зроблено на створення симуляційного макету та окремого елемнту «Камера» зі своїм внутрішнім інтерфейсом для ідентифікації об’єктів.

На основі аналізу сучасних систем сортування, що відбуваються за використанням людини було розроблено симуляційний макет з використанням програмного забезпечення CoppeliaSim, Microsoft Visual Studio Code та Blender. Ці інструменти дозволили створити реалістичний симуляційний макет в CoppeliaSim, окремий модуль «Камера» з своїм інтерфейсом та можливістю ідентифікації об’єктів на основі Yolo5su та внутрішнього дата-банку, що забезпечило високу точність виконання задач проєкту.

В результаті була створеня симуляційна система, що може бути встановлена в підвальних приміщеннях жилих будівль. Система дозволяє зменшити використання людської праці в небезпечних зонах перероблювальних заводів.

Таким чином, розроблене рішення демонструє високий потенціал для впровадження її у жилих будівлях, що прагне підвищити ефективність перероблення сміття в усьому світі за допомогою автоматизованих сортувальних ліній.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Документація. Звіти у сфері науково-технічної діяльності. Структура та правила оформлення. – [Чинний від 01.01.2016]. – К.: ДП «УкрНДНЦ», 2016. – 24 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І.Ш. Невлюдов, О.І. Филипенко, О.В. Токарєва, С.П. Новоселов, О.В Сичова. Харків: ХНУРЕ, 2023. 64 с.

3. Навчальний посібник з підготовки кваліфікаційної роботи бакалавра для здобувачів вищої освіти денної і заочної форм навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» : Навчальний посібник / І. Ш. Невлюдов, В.А. Андрусевич, О. В. Токарєва, С. П. Новоселов, О. В. Сичова. – Харків : Видавництво Іванченка І. С., 2022. 151 с.

4. BEAM робототехніка [Електронний ресурс] : Навчальний посібник / І. Ш. Невлюдов, В. В. Євсєєв, С. С. Максимова ; Харків : Видавець Чернявський Д. О., 2024. 276 с.

5. Plastic Food Packaging Symbols: Resin Identification Codes. Thong Guan Industries. URL: https://www.thongguan.com/plastic-food-packaging-symbols-and-what-they-mean/ (дата звернення: 07.04.2025).

6. Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions A European Strategy for Plastics in a Circular Economy. Announcement: server inaccessibility - European Commission. URL: https://webgate.ec.europa.eu/circabc-ewpp/d/d/workspace/SpacesStore/ad41d97c-ad7f-43be-803a-197a0f1c02ca/file.bin (дата звернення: 07.04.2025).

7. Rosato D. V., Rosato, D. V. Plastics Engineered Product Design. Elsevier Science, 2003. 588 с.

8. Fig. 2: Feature groups correlation and independence testing. | Communications Engineering. Nature. URL: https://www.nature.com/articles/s44172-024-00178-4/figures/2 (дата звернення: 10.04.2025).

9. What is Computer Vision: Benefits, Types, Libraries and more. RISHABH SOFTWARE. URL: https://www.rishabhsoft.com/blog/computer-vision. (дата звернення: 10.04.2025).

10. ALLISON MARSH. In 1961, the First Robot Arm Punched In. IEEE Spectrum. URL: https://spectrum.ieee.org/unimation-robot (дата звернення: 11.04.2025).

11. ECE2008 Robotics and Automation_Budhaditya_Bhattacharyya. studocu. URL: https://www.studocu.com/de/document/vellore-institute-of-technology/linear-algebra/4-classification-of-robot-coordinate-system-classification-of-joints-degrees-of-freedom-10-dec-2019-material-ii-10-dec-201/6922872 (дата звернення: 14.04.2025).

12. Machine Learning Tutorial - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/machine-learning/ (дата звернення: 14.04.2025).

13. Energy Consumption Analysis in the Plastic Waste Recycling Process: A Case Study of Amazia Vision Enterprise Private Limited, Satara, India| Semantic Scholar. URL: https://pdfs.semanticscholar.org/b230/ef5840c0a15ff1e36d67499ada8e34dc0151.pdf (дата звернення: 14.04.2025).

14. Prof Rohan Munasinghe. Lecture 02, Industrial Robot Manipulators. Sudath R. Munasinghe. URL: https://rohan.staff.uom.lk/teaching/ME5144/LectureNotes/Lec%202%20Robot%20Manipulators.pdf (дата звернення 24.04.2025).

15. Programming the Prab Versatran Robot. Audio projects and DIY guides -- since 2005. URL: https://www.gammaelectronics.xyz/tab-2421_5.html (дата звернення 24.04.2025).

16. The presentation of World Robotics 2024. International Federation of Robotics (IFR). URL: https://ifr.org/img/worldrobotics/Press_Conference_2024.pdf (дата звернення 30.04.2025)

17. ETHW. Milestones:SHAKEY: The World's First Mobile Intelligent Robot, 1972. ETHW. URL: https://ethw.org/Milestones:SHAKEY:_The_World's_First_Mobile_Intelligent_Robot,_1972 (дата звернення 30.04.2025)

18. CUED (groups) Web server. URL: http://www-g.eng.cam.ac.uk/impee/topics/RecyclePlastics/files/RecyclingEnergyBalance.pdf (дата звернення: 04.05.2025).

19. Engineeringtoolbox E. Conveyors - load & power consumption. The Engineering ToolBox. URL: https://www.engineeringtoolbox.com/conveyor-power-load-d_1560.html (дата звернення: 13.05.2025).

20. Plastic Waste Facts and Statistics. Business Waste. URL: https://www.businesswaste.co.uk/your-waste/plastic-recycling/plastic-waste-facts-and-statistics/ (дата звернення: 13.05.2025).