Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)
«Автоматизована система позиціонування маніпулятора із застосуванням оптичної системи робота»
(тема)

Виконав:
студент 2 курсу, групи ІТМРТм-20-1

Кондратюк М.В.
(прізвище, ініціали)

Спеціальності 172 Інтелектуальні технології мікросистемної радіоелектронної техніки
(код і повна назва спеціальності)
Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Інтелектуальні технології мікросистемної радіоелектронної техніки
(повна назва освітньої програми)
Керівник Новоселов С.П
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАМ _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2021 р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАМ
Рівень вищої освіти другий (магістерський)
Спеціальність 172 Телекомунікації та радіотехніка
Тип програми Освітньо-професійна
Освітня програма Інтелектуальні технології мікросистемної радіоелектронної техніки
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАМ _____
(підпис)
«____»_____2021р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Кондратюку Максиму Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Автоматизована система позиціонування маніпулятора із застосуванням оптичної системи робота

Затверджена наказом по університету від 08.11.2021 № 1697Ст

2. Термін подання студентом роботи до екзаменаційної комісії 13.12.2021

3. Вихідні дані до роботи В якості об'єкту керування виступає учбовий макет робота маніпулятора
Розміри макету: 300 мм х 220 мм х 350 мм.
Напруга живлення макету – 12 В.
Формат команд управління: g-code.
Кількість ступенів свободи – 4.
Споживаний струм – 2,3 А.
інтерфейс передачі даних – USB.

4. Перелік витань, що потрібно опрацювати в роботі Вступ. Аналіз предметної області. Аналіз методів обробки зображень, що використовуються для системи комп'ютерного зору маніпулятором. Розробка методу позиціонування маніпулятора. Експериментальні дослідження. Охорона праці. Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*,ppt) – с формату А4.

6. Консультанти розділів роботи

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	Примітка
1	Аналіз предметної області	08.11.2021	Виконано
2	Аналіз методів обробки зображень, що	10.11.2021	Виконано
	використовуються для системи		Виконано
	комп'ютерного зору маніпулятором		Виконано
3	Розробка методу позиціонування	15.11.2021	Виконано
	маніпулятора		Виконано
4	Проведення експериментальних	20.11.2021	
	досліджень		Виконано
5	Оформлення пояснювальної записки	30.11.2021	Виконано
6	Подання роботи на перевірку інтернет	8.12.2021	
	сервісу Unicheck		Виконано
7	Подання роботи на рецензію	9.12.2021	Виконано
8	Подання роботи на підпис зав. кафедри	13.12.2021	Виконано
9	Подання атестаційної роботи до ЕК	13.12.2021	Виконано

Дата видачі завдання 08.11.2021

Студент _____ Кондратюк М.В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. каф. КІТАМ Новоселов С. П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 78 с., 33 рис., 3 таблиць, 13 джерел, 2 додаток.

C#, EMGU CV, MYSQL, MVC, OPENCV, АВТОМАТИЗОВАНИЙ МОДУЛЬ, СИСТЕМА КОМП'ЮТЕРНОГО ЗОРУ, МЕТОД ВИЗНАЧЕННЯ КООРДИНАТ ОБ'ЄКТІВ, РОЗПІЗНАВАННЯ, СУБД.

Об'єкт дослідження – система позиціонування промислового маніпулятора в робочому просторі.

Предмет дослідження – метод позиціонування макету маніпулятора із застосування системи комп'ютерного зору.

Мета атестаційної роботи – вибір методу визначення координат об'єктів маніпуляції та розробка алгоритмічних і програмних рішень для вирішення задачі використання системи комп'ютерного зору для позиціонування промислового маніпулятора в робочому просторі.

Досліджені особливості використання систем комп'ютерного зору на виробництві.

Проаналізована конструкцію макету промислового маніпулятора.

Розроблена структурна та функціональні схеми системи позиціонування.

Розроблена програма для визначення координат об'єктів в робочому просторі та керування макетом маніпулятора.

Виконані експериментальні дослідження для підтвердження правильності обраного методу позиціонування маніпулятора.

На стороні серверу використовуються такі технології та мови програмування: .Net, мова програмування C#, СУБД MySQL, аналог Open CV – Emgu CV.

ABSTRACT

Explanatory note: 78 pp., 33 pic., 3 tables, 13 sources, 2 additional.

C#, EMGU CV, MYSQL, MVC, OPENCV, AUTOMATION MODULE, COMPUTER VISION SYSTEM, COORDINATE DETECTION METHOD, RECOGNITION, SUBD.

The object of the research is the system of industrial manipulator positioning in the working space.

The subject of the research is the method of positioning the manipulator layout using the computer vision system.

Objective of the test work – the choice of method for determining the coordinates of the manipulation objects and development of algorithmic and software solutions to solve the problem of using the computer vision system for positioning the industrial manipulator in the working space.

Peculiarities of using computer vision systems in production were investigated.

The design of the industrial manipulator layout has been analyzed.

The structural and functional schemes of the positioning system were developed.

Developed a program for determining the coordinates of objects in the workspace and manipulation of the manipulator layout.

Experimental studies to confirm the correctness of the reverse method of positioning the manipulator.

The following programming languages are used on the server side: .Net, C# programming language, DBMS MySQL, Open CV analogue – Emgu CV.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, СИМВОЛІВ І ТЕРМІНІВ

ДСТУ – державний стандарт України;

МПА – мікропрограмний автомат;

ОЗП – оперативний запам’ятовуючий пристрій;

ПК – персональний комп’ютер;

ПУ – пульт управління;

FBD – функціонально-блокові діаграми;

IL – список інструкцій;

LD – мова сходових діаграм;

OpenCV – open source computer vision (комп’ютерний зір з відкритим кодом);

SFC – діаграми послідовних функцій;

SLAM – simultaneous localization and mapping (одночасна локалізація та побудова карти місцевості);

ST – структурований текст.

ВСТУП

Технічний (машинний) зір – це сфера застосування комп'ютерного зору в промисловості або виробництві. На сьогоднішній день все більшої популярності набувають системи комп’ютерного зору в якості пристроїв визначення положення координат корисного вантажу при роботі з маніпуляційними роботами.

Прості представники подібних систем дозволяють визначати декартові координати x і y, а також кут орієнтації R корисних вантажів, що знаходяться в одній площині робочого простору маніпуляційного робота. Більш складні рішення дозволяють визначити три координати x, y і z в робочому просторі. Комплексні рішення дозволяють визначити всі шість координат x, y, z, φ, θ, ψ, але такі рішення вимагають особливих функціонування, тому їх область застосування сильно обмежена.

Функціональні завдання СТЗ, характерні для роботехнічних комплексів, умовно поділяються за рівнем їх відносної складності. До простих завдань можна віднести: виявлення наявності будь-якого об'єкта, вимір відстані до нього, обчислення його лінійних і кутових переміщень, швидкості; вимір геометричних параметрів об'єкта (лінійні і кутові розміри, площа), визначення фізичних характеристик випромінювання від об'єкта, підрахунок числа об'єктів в кадрі.

Більш складні задачі виконує система, яка надає маніпулятору інформацію, необхідну для переміщення невпорядкованих об'єктів. До цих задач відносять: огляд робочого простору для пошуку об'єкта, що цікавить, який може бути одиночним, або одним з декількох, або його місце розташування може бути ізольовано, перекриватися іншими об'єктами. При цьому об'єкти можуть відрізнятися не тільки формою і розміром, але й кольором, текстурою і т.д., можуть знаходиться в русі або покоїться на місці.

Таким чином, використання можливостей комп'ютерного зору безпосередньо в процесі виробництва продукції збільшує точність виконання виробничих операцій, контролю якості виготовлення продукції, підвищення швидкості виконання операцій.

Актуальністю досліджень є необхідність вдосконалення методів позиціонування промислових маніпуляторів для вирішення задачі переміщення предметів в робочому просторі для виконання заданого технологічного процесу.

Об'єкт дослідження – система позиціонування промислового маніпулятора в робочому просторі.

Предмет дослідження – метод позиціонування макету маніпулятора із застосування системи комп’ютерного зору.

Мета атестаційної роботи – вибір методу визначення координат об’єктів маніпуляції та розробка алгоритмічних і програмних рішень для вирішення задачі використання системи комп’ютерного зору для позиціонування промислового маніпулятора в робочому просторі.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– дослідити особливості використання систем комп’ютерного зору на виробництві;

– проаналізувати конструкцію макету промислового маніпулятора;

– розробити структурну та функціональні схеми системи позиціонування;

– розробити програму для визначення координат об’єктів в робочому просторі та керування макетом маніпулятора;

– виконати експериментальні дослідження для підтвердження правильності обраного методу позиціонування маніпулятора;

– оформити магістерську атестаційну роботу згідно ДСТУ 3008:2015, а також з методичними вказівками з розробки й оформлення магістерської атестаційної роботи другого (магістерського) рівня вищої освіти галузі знань 17 Електроніка та телекомунікації за спеціальністю 172 Телекомунікації та радіотехніка.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз методів застосування машинного зору для промислових роботів

Сучасна промисловість активно використовує роботи-маніпулятори різної модифікації і потужності. Особливо це актуально для впровадження концепції Industry 4.0. Промислові маніпулятори універсалізують, спрощують і виконують однотипні повторювані операції швидше і ефективніше ніж людина, при цьому мінімізують будь-яку похибку і виробничі ризики. Але відомо, що істотним недоліком даних маніпуляторів є їх нездатність до самоадаптації до постійно змінюваних параметрів навколишнього середовища. Наприклад, якщо об'єкт маніпуляції змінив своє базове розташування або має відмінні характеристики і властивості від заданих, такі роботи не зможуть вже виконувати закладену в них програму.

На рисунку 1.1 показано приклад застосовування маніпуляторів на виробництві.

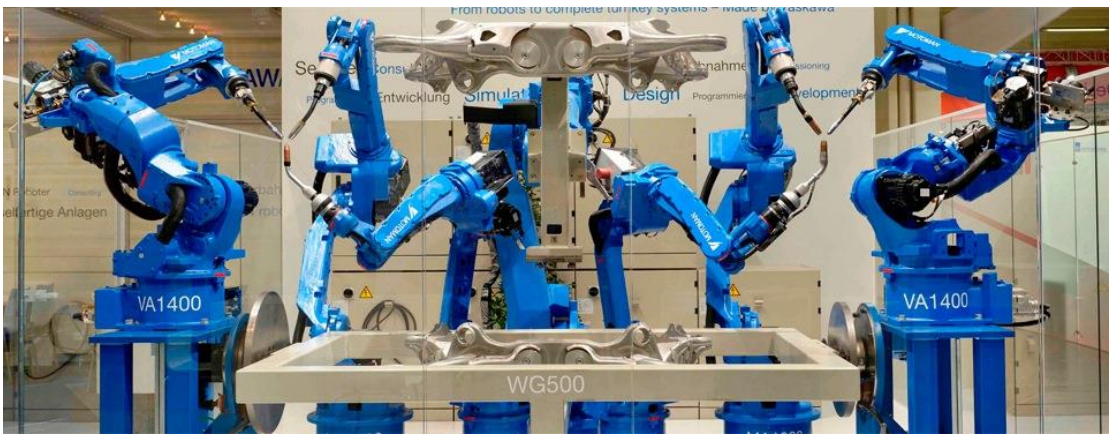


Рисунок 1.1 – Приклад застосовування маніпуляторів на виробництві

Із застосуванням машинного зору маніпулятор набуває можливість бути більш гнучким і автоматично адаптуватися до умов, що змінюються, пристосовуватись до нетипових ситуацій на виробництві.

Таким чином, задача модифікації робота-маніпулятора за рахунок додавання можливостей автоматично аналізувати ситуацію та пристосовуватись до її змін, при цьому не потребуватиме повного перепрограмування, е досить актуальною.

Рішення поставленої мети, спрощую подальше використання робота-маніпулятора на виробництві, сприяє підвищенню стабільності роботи, збільшує час безперебійної роботи автоматизованого комплексу із застосуванням адаптованих роботів. Також, мінімізуються надзвичайні ситуації та збільшуються виробничі потужності.

1.2 Аналіз об’єкту керування

В якості об’єкту керування виступає учбовий макет робота маніпулятора. Маніпулятор має два рухомих суглоби, може обертатись навколо вертикальної вісі, та може переміщуватись на рельсах вліво та вправо. Також маніпулятор має схват для захвату та переміщення деталей в межах його робочої зони.

В основі конструкції лежать чотири крокових двигуна. Кожен крековий двигун реалізує певну ступень свободи. Керуються двигуни від модуля управління, шо побудовано на основі контролера Arduino Mega.

Маніпулятор має кінцеві датчики по одному на кожну ступень свободи. На початку роботи виконується початкова ініціалізація системи керування. При цьому виконується тестовий запуск кожного крокового двигуна і відстежується спрацювання відповідного кінцевого датчика. Якщо всі датчики спрацювали, то пристрій переходить в режим очікування команд від користувача.

На рисунку 1.2 показано зовнішній вигляд маніпулятора.

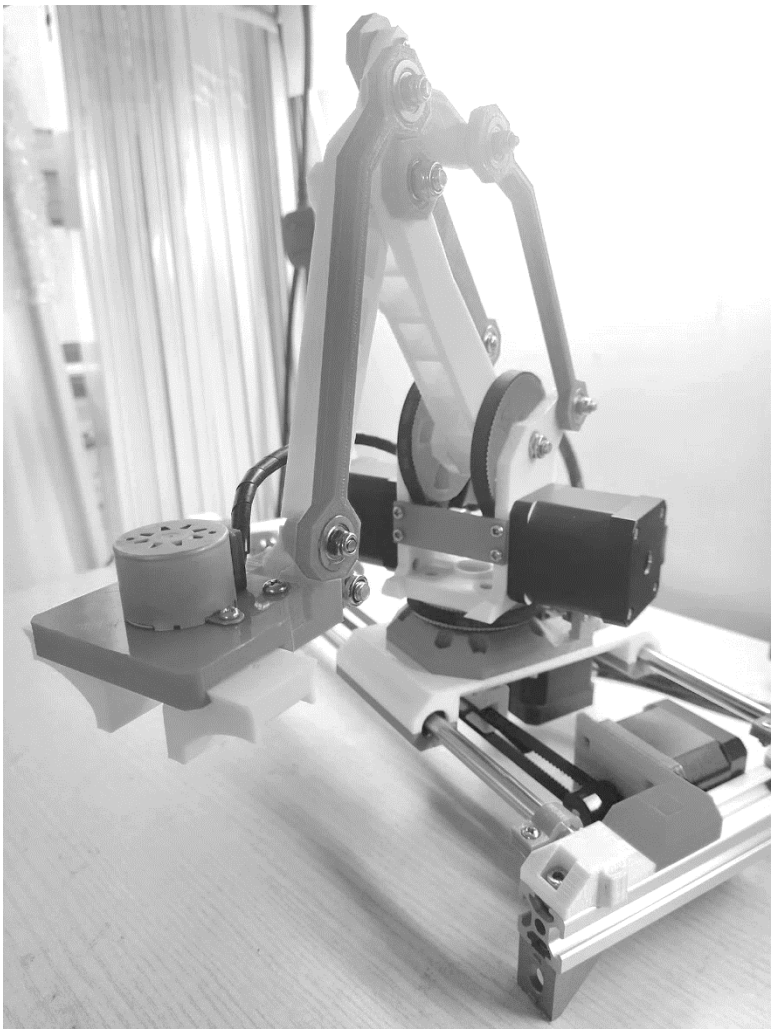


Рисунок 1.2 – Зовнішній вигляд маніпулятора

Програма керування використовує в якості ядра модифіковану прошивку Marlin від 3D-принтеру. Прошивка значно спрощена по відношенню до оригіналу та містить лише частину функцій керування пристроєм.

В якості команд керування використовуються стандартна для промислового обладнання мова G-code. Поєднання з персональним комп’ютером виконується за допомогою послідовного інтерфейсу RS-232.

Для того, щоб розробити програму керування маніпулятором та поєднати її з системою комп’ютерного зору, необхідно визначити кінематичну схему пристрою, розробити його математичну модель, визначити габаритні розміри конструкції та робочу область.

На рисунку 1.3 показано ескіз конструкції маніпулятора.

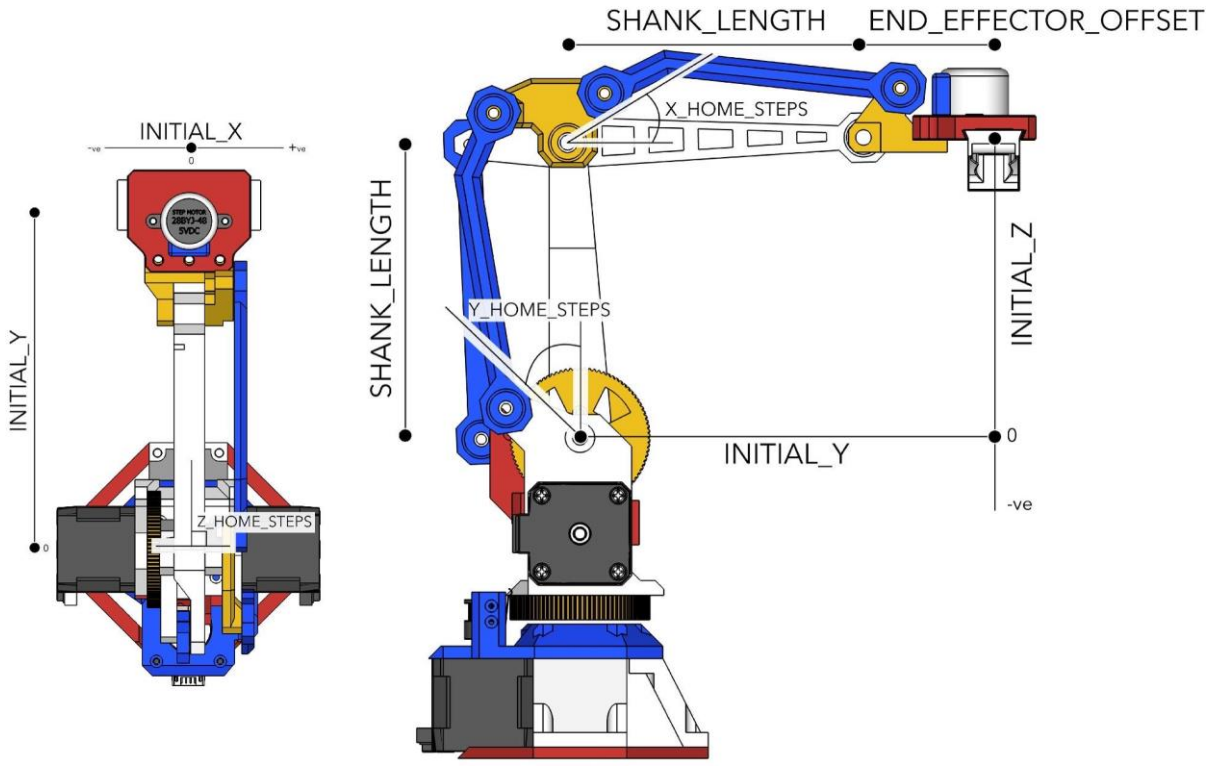


Рисунок 1.3 – Ескіз конструкції маніпулятора

Конструкція, що використовується в роботі має такі розміри:

– LOW_SHANK_LENGTH (нижнє плече): 140 мм;

– HIGH_SHANK_LENGTH (верхнє плече): 140 мм;

– END_EFFECTOR_OFFSET (відстань від місця кріплення інструменту до його кінцевої точки): 55мм;

– INITIAL_X (початкова координата X): 0, 0;

– INITIAL_Y (початкова координата Y): HIGH_SHANK_LENGTH + END_EFFECTOR_OFFSET = 195 мм;

– INITIAL_Z (початкова координата Z): дорівнює розміру LOW_SHANK_LENGTH, та становить 140 мм.

На рисунку 1.4 показано просторові зони досяжності маніпулятора.

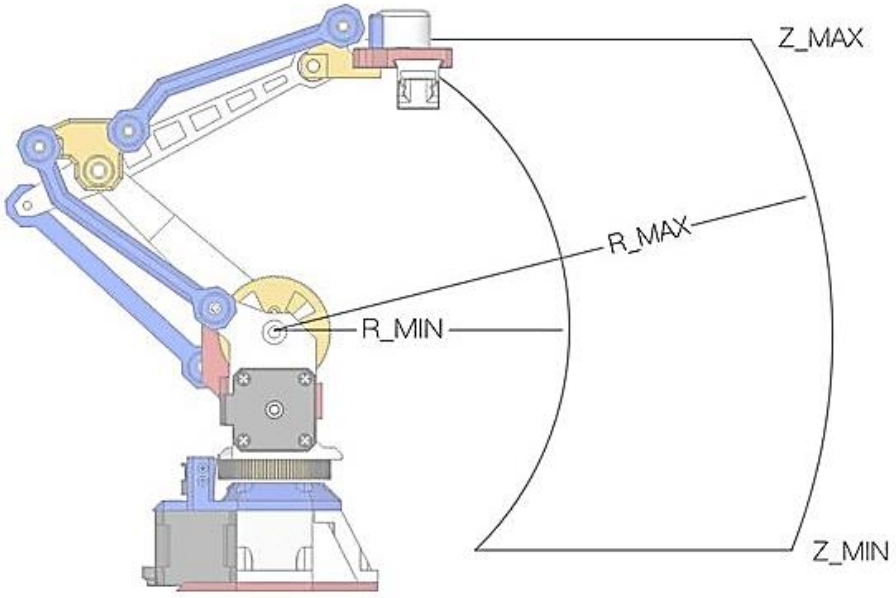


Рисунок 1.4 – Просторові зони досяжності маніпулятора у вертикальній плоскості

Просторова зона дії маніпулятора в вертикальній плоскості обмежена конструктивними особливостями пристрою та на рисунку 1.4 представлена радіусами R_MIN, R_MAX, та поверхнею Z_MIN, та Z_MAX.

В горизонтальній плоскості рух маніпулятора обмежений радіусами R_MIN, RMAX.

Таким чином, ми маємо множину координат, що розташовані в межах зони досяжності маніпулятора, що можна представити наступним чином (1.1):

$$M_{xyz} \in \{R_{xyz}^{min}, R_{xyz}^{max}, Z_{xyz}^{min}, Z_{xyz}^{max}\} \quad (1.1)$$

На рисунку 1.5 показано ескіз зони досяжності маніпулятора у горизонталній плоскості

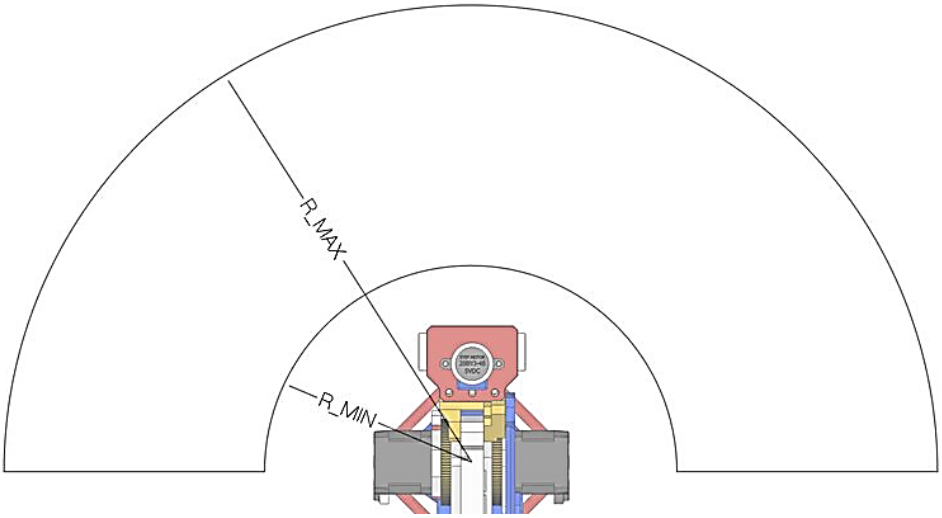


Рисунок 1.5 – Просторові зони досяжності маніпулятора у горизонтальній плоскості

Маніпулятор має можливість переміщуватись вліво та вправо за допомогою рейок та ремінної передачі. На рисунку 1.6 показано 3D-модель конструкції механізму лінійного переміщення маніпулятора.

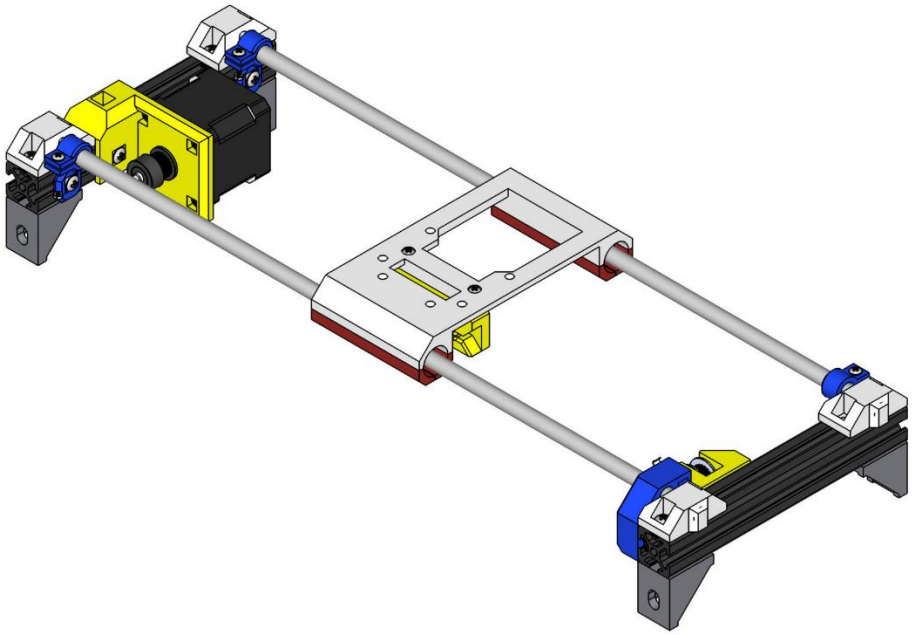


Рисунок 1.6 – 3D-модель конструкції механізму лінійного переміщення маніпулятора

1.3 Аналіз кінематичної схеми маніпулятора

Обраний маніпулятор має дві поєднані ланки, що утворюють елементарну складову механізму – кінематичну пару. Послідовність попарно пов'язаних ланок складає кінематичний ланцюг. Кінематичний ланцюг може бути розімкненим або замкнутим. Ланки і зчленування маніпулятора нумеруються за збільшенням від основи до робочого органу. Нульова ланка сполучена з нерухомою основою, а до останньої ланки прикріплений робочий орган (захват).

Кінематична схема маніпулятора показана на рисунку 1.7.

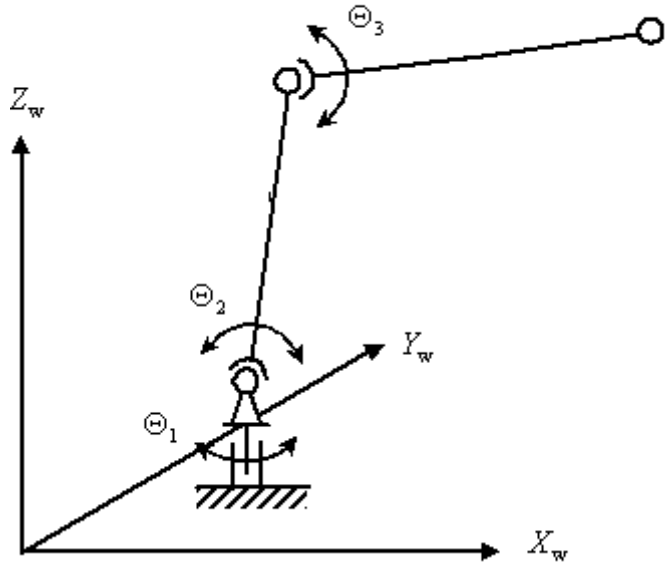


Рисунок 1.7 – Кінематична схема маніпулятора

Кожне зчленування переміщає відповідну ланку, тому у нумерації і-тій ланці передує і-те зчленування. Ланки маніпулятора беруть участь у відносному русі, в результаті якого досягається певне положення і орієнтація робочого органу. Переміщення ланок в просторі здійснюється за допомогою приводів, розташованих, як правило, в зчленуваннях. Кожна пара, що складається з ланки і зчленування, має один ступінь свободи, отже, маніпулятор з n парами «ланка-зчленування» має n ступенів свободи. Таким чином, маніпулятори можуть

розрізнятися послідовностями і комбінаціями обертальних і поступальних зчленувань, тобто кінематичними схемами, які визначають характер основних рухів і робочої зони маніпулятора. Кінематична схема маніпулятора є послідовністю і комбінаціями обертальних і поступальних зчленувань.

На рисунку 1.8 показано орієнтацію ланок маніпулятора.

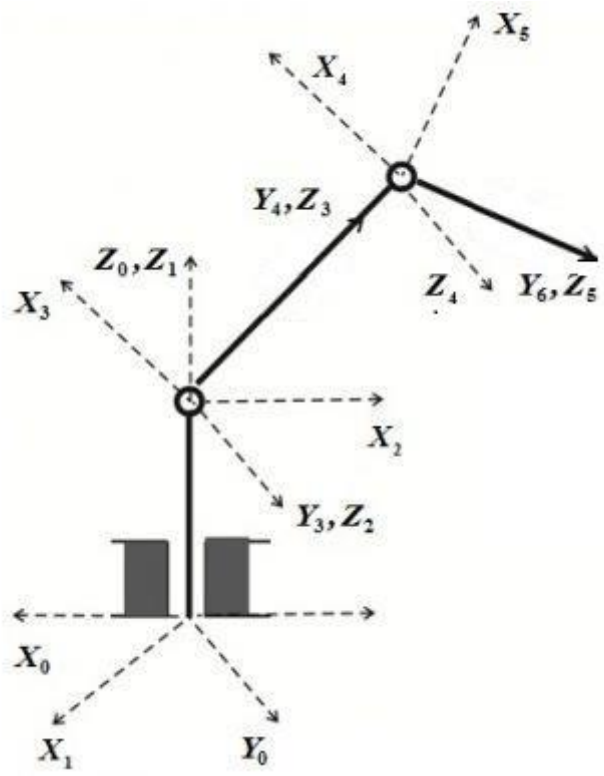


Рисунок 1.8 – Орієнтації ланок маніпулятора

Для робота маніпулятора, що складається з сукупності ланок, як правило, необхідно визначити його просторову конфігурацію з урахуванням положення усіх ланок. Конфігурація може бути визначена шляхом послідовного опису взаємного розташування сусідніх ланок маніпулятора. Для виконання переміщення захоплення маніпулятора в задану точку нам необхідно здійснити вибір орієнтації ланок маніпулятора. Для цього необхідно виконати наступні пункти:

– побудувати абсолютну системи координат;

– побудувати ортогональну систему координат $X_0Y_0Z_0$, направивши Z_0 уздовж осі першого зчленування у напрямі схвата;

– побудувати Z_i. Направити вісь Z_i уздовж осі (i+1)-го шарніра. При i = N (тобто для схвата) виберемо вісь Z_N у напрямі осі $Z_N - 1$;

– визначити геометричні параметри маніпулятора;

– виставити координати кожної ланки у відповідне положення;

– визначення початкових значень кутів повороту кожної ланки.

Виконаємо аналіз розрахунку кута повороту маніпулятора.

Відомо, що точки, по яких переміщається ланка маніпулятора, розташовані на колі з центром в точці опори ланки і радіусом рівним довжині самої ланки. Тоді для визначення кута повороту маніпулятора скористаємося існуючими формулами при перетині двох кіл і усіма відомою теоремою Піфагора, як показано на рисунку 1.9.

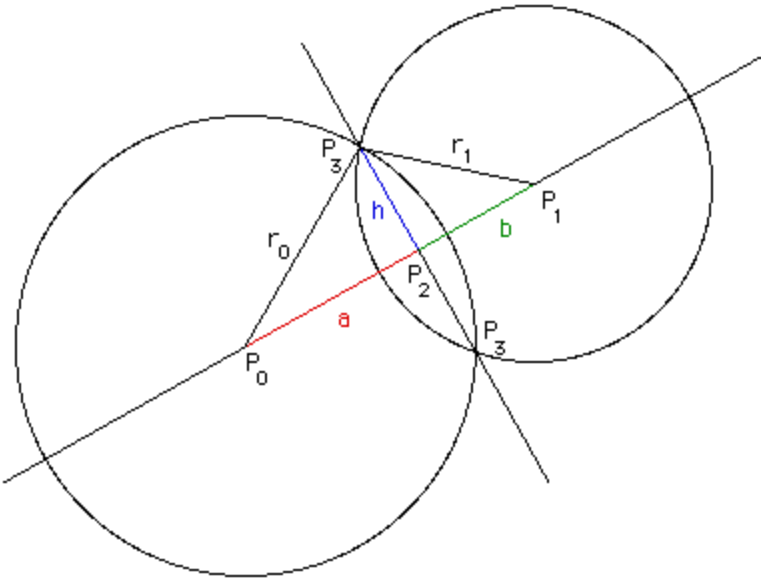


Рисунок 1.9 – Перетин двох кіл

Знайдемо відстань між центрами пересічених кіл, рисунок 2.1. d = |P1 - P0|. Якщо d > r0 + r1, тоді рішень немає: круги лежать окремо. Аналогічно у разі d < | r0 - r1 | – тоді немає рішень, оскільки одне коло знаходиться усередині іншої.

Розглянемо два трикутники $P_0P_2P_3$ і $P_1P_2P_3$. Маємо

$$a^2 + h^2 = r_0^2$$

$$b^2 + h^2 = r_1^2$$

Використовуючи рівність d = a + b, можемо виконати рішення відносно a:

$$a = (r_0^2 - r_1^2 + d^2) / (2\ d)$$

У разі зіткнення кіл, це перетвориться на r_0, оскільки:

$$d = r_0 + r_1$$

Виконаємо рішення відносно h, підставивши в перше рівняння

$$h^2 = r_0^2 - a^2$$

Таким чином,

$$P_2 = P_0 + a\ (\ P_1 - P_0\ ) / d$$

Отже, отримуємо координати точок $P_3 = (x_3,y_3)$:

$$x_3 = x_2 + h\ (\ y_1 - y_0\ ) / d$$

$$y_3 = y_2 -+ h\ (\ x_1 - x_0\ ) / d$$

Для того, щоб отримати кут повороту базової ланки маніпулятора необхідно скористатися тригонометричними функціями і в результаті отримаємо остаточну формулу кута повороту ланки.

$$\alpha = \arccos(a/r_0)$$

Аналогічно можемо отримати кут розвороту інших ланок.

1.4 Аналіз систем програмного управління промисловими роботами

Терміном управління в техніці зазвичай називають процес автоматичної реалізації сукупності дій, що додаються до деякого об'єкту, або з метою підтримки його функціонування на заданому рівні, або з метою зміни у бажаному напрямі його регульованих параметрів.

Робот, як об'єкт управління, є складною системою, що включає багатоланкову механічну конструкцію з виконавчими пневмо-, гідро- чи електроприводами, що активно взаємодіє з довкіллям і характеризується сукупністю параметрів, що змінюються в часі.

Виділяють три ієрархічні рівні автоматизації роботизованого устаткування:

– погоджене управління роботом з одиницею промислового устаткування, при якому команди управління роботом подає обслуговувана ним технологічна машина;

– управління роботом і декількома одиницями технологічного устаткування, при якому поведінка робота визначається запитами від обслуговуваних їм машин;

– управління розподіленими роботами, одиницями устаткування, складами, транспортними засобами від ПК з метою виконання замовлень, що поступають.

Модуль управління приймає сигнали від датчиків і ПК, після чого виробляє команди на виконавчі пристрої відповідно до записаної програми управління. Модуль управління роботом повинен додатково:

– регулювати положення і швидкості переміщення приводів ланок;

– враховувати стан обслуговуваних одиниць устаткування.

Зазвичай в системах управління використовуються три основні принципи: розімкненого управління, управління по обуренню і принцип зворотного зв'язку.

Принцип розімкненого управління реалізується тільки на основі бажаного алгоритму поведінки керованого об'єкту і не враховує можливість появи зовнішніх обурюючих дій, здатних викликати неконтрольовані відхилення в процесі функціонування об'єкту.

Наприклад, швидкість обертання вихідного валу електродвигуна постійного струму з незалежним збудженням пропорційна напрузі, прикладеній до якоря, рисунок 1.10.

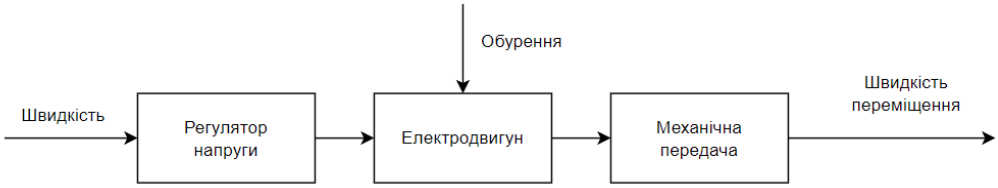


Рисунок 1.10 – Схема розімкненого управління

Подаючи на двигун напругу бажаної величини, управляють швидкістю обертання його вихідного валу і, як наслідок, швидкістю переміщення ланки робота, сполученого з цим двигуном.

Проте, якщо на цю ланку діють зовнішні сили, наприклад, статичний момент сили тяжіння ланки, що змінюється за величиною залежно від кутового положення міри рухливості, то швидкість обертання електродвигуна істотно відрізнятиметься від заданої і, крім того, змінюватиметься у функції від кута повороту ланки робота.

Принцип управління по обуренню може використовуватися в системах розімкненого типу, що перебувають під впливом деяких обурюючих дій, для того, щоб компенсувати відхилення регульованого параметра, викликаного домінуючою дією, рисунок 1.11.

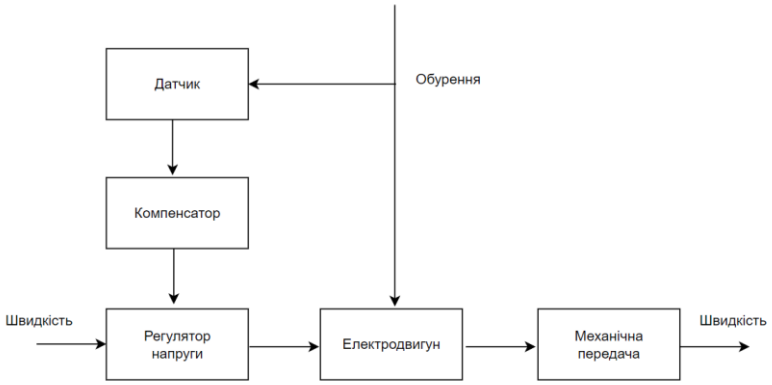


Рисунок 1.11 – Схема управління по обуренню

При управлінні по обуренню вдається компенсувати небажаний вплив на процес регулювання тільки тієї дії, яка вимірюється датчиком, а інші дії як і раніше можуть викликати неконтрольовані відхилення.

Відмітною ознакою систем управління, в яких реалізований принцип зворотного зв'язку, являється вимір регульованого параметра і використання отриманої інформації при формуванні закону управління, рисунок 1.12.

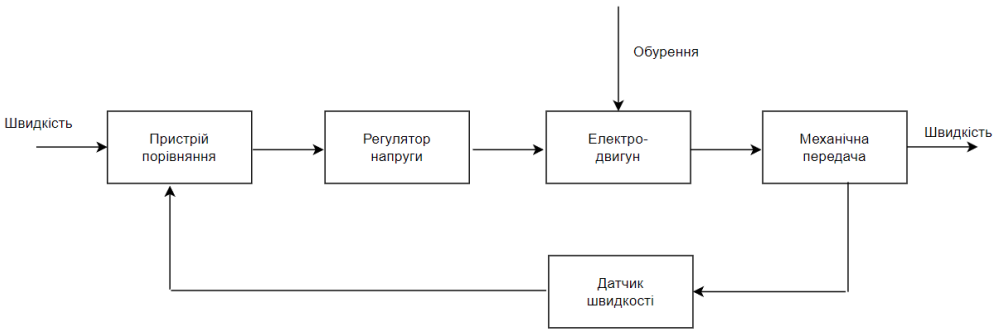


Рисунок 1.12 – Схема управління із зворотним зв'язком.

Системи управління, замкнуті по регульованій координаті, мають кращі характеристики в порівнянні з системами, створеними на основі принципу управління по обуренню, оскільки незалежно від причини, що викликала відхилення цієї координати від заданого значення, величина і знак відхилення можуть бути оцінені і на їх основі можуть бути здійснені дії, що коригують, підвищують міру збігу поточної і необхідної поведінки об'єкту.

У замкнутих системах автоматичного регулювання застосовують різні закони управління. Законом управління зазвичай називають математичну залежність, відповідно до якої регулятор формує дію на об'єкт управління в припущенні, що останній є безінерційним.

При проектуванні конкретних робототехнічних систем необхідно враховувати можливості робота по відробітку заданого положення або бажаної траєкторії рухи захватного пристрою і вимоги, що витікають з технології виконання того або іншого виробничого завдання.

Наприклад, якщо передбачається автоматизувати операцію завантаження-вивантаження пресу, то не обов'язково оснащувати робот замкнутою системою управління. Досить використати принцип розімкненого управління. Для автоматизації дугового зварювання необхідно управляти з високою мірою точності не лише положенням електроду, але і його швидкістю в умовах значних зовнішніх дій, і, отже, робот, призначений для цієї роботи, повинен мати систему управління, що реалізовує найдосконаліші принципи і закони управління положенням і швидкістю.

Системи програмного управління промисловими роботами підрозділяються на системи:

– циклового;

– позиційного;

– контурного управління.

Кожна з них має багато різновидів залежно від характеру операцій, кінематики робота, приводів.

Цикловий режим програмного управління є найбільш простим, рисунок 1.13. Він застосовується, як правило, в тих випадках, коли по кожній з n мір рухливості qi (i =1, 2, .n) робота можливе позиціонування лише в двох крайніх точках qін, qік, де відповідні індекси Н, До означає початкове і кінцеве положення відповідних рухливих елементів.

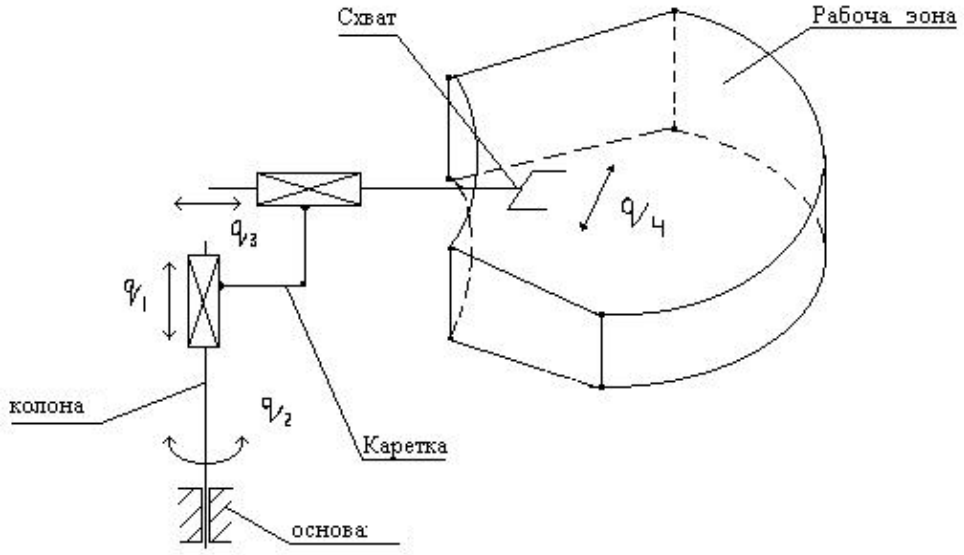


Рисунок 1.13 – Робот з цикловою системою програмного управління

Для перекладу і-ої ланки в стан qін необхідно на його привід подати нульовий сигнал Ui = 0, що управляє. Для перекладу цієї ланки в стан qік на той же привід необхідно подати сигнал Ui = 1. Причому під час переходу робота в новий стан при незмінному керівнику векторі U = (U1, U2, …, Un) цей стан залишається незмінним.

При організації циклового управління роботом можуть бути використані досить прості автомати, побудовані на елементах цифрової обчислювальної техніки. Такі автомати включають: пам'ять послідовності кодів необхідних станів робота, пам'ять послідовності тимчасових інтервалів, тобто тих інтервалів часу, в яких вектори, що управляють, залишаються незмінними, тимчасовий пристрій і блок управління переходами.

Для побудови подібних автоматів потрібні регістри, дешифратори, лічильники, тригери і тому подібне. Враховуючи, що контролери ПР є автоматами з пам'яттю, при їх розробці можуть бути використані більші вузли обчислювальної техніки, такі, як пам'ять і блоки мікропрограмного управління сучасних мікропроцесорних комплектів.

Для побудови універсальних контроллерів доцільно не лише використати окремі вузли мікропроцесорних комплектів, не лише застосовувати в структурі контролерів мікропроцесори, але і самі контролери створювати на основі мікропроцесорів і міні-ПК.

Для введення програми використовують пульт управління, портативний програматор або персональний комп'ютер із спеціальним програмним забезпеченням. Ускладнення програм управління привело до відмови від застосування програматорів, а різноманіття контролерів з власними мовами програмування до появи п'яти технологічних мов програмування: мови сходових діаграм (LD), функціонально-блокових діаграм (FBD), списку інструкцій (IL), діаграм послідовних функцій (SFC), структурованого тексту (ST).

Пульт управління і модуль управління складають технічні засоби спілкування оператора з мікроконтролером. Модулі введення і виводу призначені для зв'язку мікроконтролера з виконавчими облаштуваннями маніпулятора робота і з технологічним устаткуванням. Модуль послідовного інтерфейсу служить для обміну інформацією між мікроконтролером і обчислювальним комплексом, що управляє, верхнього рівня.

Мікроконтролер має наступні режими роботи:

– автоматичне управління за програмою, записаною в пам'ять робочої програми;

– ручне управління по командах, поданих з пульта управління;

– покрокове виконання програми;

– програмування (запис команд в пам'ять робочих програм);

– перегляд програми (вивід на індикацію вмісту пам'яті робочих програм).

Позиційні системи управління роботами відносяться до загального класу систем автоматичного управління. Основними технічними характеристиками систем управління, що відносяться до цього класу являються: число керованих координат, об'єм пам'яті програм (кадрів), число технологічних команд обміну інформацією із зовнішнім устаткуванням, тип приводу. За способом обробки інформації (послідовна або паралельна) позиційні системи будуються по

структурі з центральним обчислювачем і з децентралізованою структурою, коли обчислювач входить до складу кожного координатного блоку.

Приклад системи з централізованою структурою показаний на рисунку 1.14.

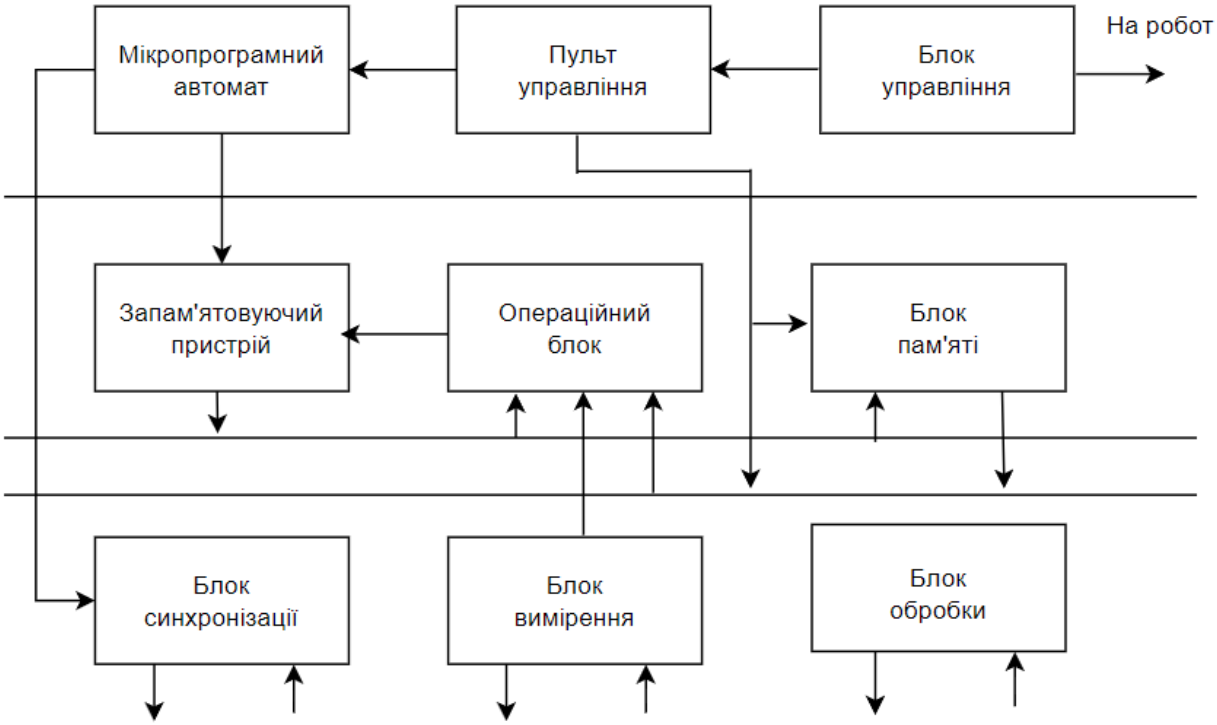


Рисунок 1.14 – Схема централізованої структури позиційної системи програмного управління

Вона побудована за принципом синхронного мікропрограмного автомата (МПА) з кінцевим числом станів і жорстким циклом управління. МПА призначений для формування мікрокоманд, що управляють, відповідно до алгоритму функціонування позиційного пристрою. Реалізація функцій центрального управління і логічної обробки інформації здійснюється операційно-логічним пристроєм (ОЛП), який разом з МПА є спеціалізованим обчислювачем. Основним носієм програм є накопичувач, який зі своїм блоком управління здійснює прийом, зберігання і видачу за запитом з МПА необхідної програми.

Інформація в накопичувача формується по зонах. В одній зоні може бути записані декілька програм. Зв'язок між зонами здійснюється за допомогою команд умовного або безумовного переходів. У режимі запису і читання інформації обмін йде через ОЗП, який призначений для оперативного зберігання робочої програми, розміщеної в одній зоні, що використовується безпосередньо для автоматичного управління роботом. Пульт управління (ПУ) призначений для завдання режимів роботи і організації управління в режимах:

– навчання;

– пошук кадру;

– програма, та ін.

Ручне управління маніпулятором здійснюється з виносного пульта управління.

1.5 Висновки по першому розділу роботи

В результаті виконання першого розділу атестаційної роботи проведено аналіз методів застосування машинного зору для промислових роботів. Виконано аналіз об’єкту керування, в якості якого виступає учбовий макет робота маніпулятора. Виконано аналіз кінематичної схеми маніпулятора для подальшої можливості створення програми керування переміщенням його суглобів в залежності від визначеної координати розташування об’єкту.

2 АНАЛІЗ МЕТОДІВ ОБРОБКИ ЗОБРАЖЕНЬ, ЩО ВИКОРИСТОВУЮТЬСЯ ДЛЯ СИСТЕМИ КОМП'ЮТЕРНОГО ЗОРУ МАНІПУЛЯТОРОМ

2.1 Аналіз основних задач, що виконуються системами комп'ютерного зору

Отримані з відеокамери зображення потрібно попередньо обробити перед виконанням процедури розпізнавання. Основними задачами, що ставляться на даному етапі є аналіз параметрів об'єкта (колір, форма, розмір, статичні або динамічні характеристики, швидкість і напрямок руху) і піксельні характеристики (насиченість, контрастність) для визначення характерних і унікальних рис. Якісний аналіз та ідентифіковані унікальні риси необхідні для отримання корисної інформації при обробці зображення.

На даний час відомі наступні методи обробки зображень комп'ютерного зору:

– Deep Learning (глибоке навчання);

– Machine Leaning (машинне навчання);

– Neural Networks (нейронні мережі).

Всі перераховані вище методи базуються на отриманні великої кількості даних при аналізі зображення та використанні певного алгоритму для виконання процесу аналізу і розпізнавання.

Існують основні три способи вирішення завдань комп'ютерного зору:

– контурний аналіз;

– пошук за шаблоном (template matching);

– пошук по характерних рис або точкам (feature detection).

Також існують інші і методи вирішення задач комп'ютерного зору, але кожен з них має індивідуальні інструменти характерні для використання в конкретному середовищі. способами і методами, наприклад, розпізнавання осіб маю алгоритми, які здійснюється особливостями генетичних даних.

2.2 Методи обробки зображень

2.2.1 Контурний аналіз

Даний метод передбачає аналіз зображення безпосередньо по контурах (межах об'єкта). Перевага використання методу контурного аналізу полягає в використання простих алгоритмів обробки при невеликих обчислювальних потужностях, за рахунок особливості роботи тільки з контуром об'єкта, а не з повним аналізом всього зображення.

Контуром об'єкта є крива, яка представляє собою межі об'єкта на зображенні. Щоб оперувати отриманими контуром, його необхідно якось закодувати [8]. Наприклад, можна вказувати вершини відрізків, що становлять контур. Або використовувати інший відомий спосіб кодування контуру – ланцюгової код Фрімена (Freeman Chain Code) [8]. На рисунку 2.1 показано приклад кодування контуру даним методом.

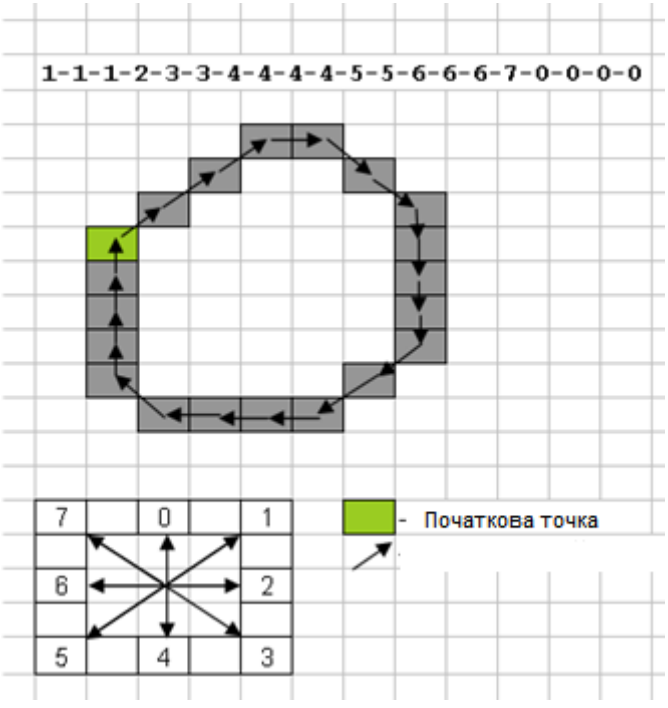


Рисунок 2.1 – Приклад використання ланцюгового коду Фрімена

Ланцюгові коди застосовуються для подання кордону у вигляді послідовності відрізків прямих ліній певної довжини і напряму. В основі цього уявлення лежить 4- або 8- зв'язкова решітка. Довжина кожного відрізка визначається здатністю решітки, а напрямки задаються обраним кодом. Для подання всіх напрямків в 4-зв'язковий решітці досить 2-х біт, а для 8-зв'язковий решітки ланцюгового коду потрібно 3 біта.

Звичайна послідовність дій при розпізнаванні об'єктів методом контурного аналізу:

– попередня обробка зображення (згладжування, фільтрація перешкод, збільшення контрасту);

– бінаризація зображення;

– виділення контурів об'єктів;

– первинна фільтрація контурів (по периметру, площі і т.п.);

– еквалізація контурів (приведення до єдиної довжині, згладжування) - дозволяє домогтися інваріантності до масштабу;

– перебір всіх знайдених контурів і пошук шаблону, максимально схожого на даний контур (або ж сортування контурів за якоюсь ознакою, наприклад, площі).

На основі аналізу літератури можна виділити деякі недоліки методу контурного аналізу:

– неправильне визначення меж об'єкта через нечіткість контуру, який виник за рахунок ефекту угруповання об'єктів або їх перекриття:

– неприпустимість визначення контуру при недостатній чіткості кордонів об'єкта на зображенні через однакової яскравості з фоном:

– помилкове детектування контуру об'єкта через низький ступень стійкості метода до шумів і перешкод.

2.2.2 Пошук за шаблоном (Template matching)

Template matching є поширеним методом комп'ютерного зору для детектування і розпізнавання об'єктів. Метод грунтується на пошуку

характерних рис (фичи / ХЧ) і порівняно вихідного зображення (оригінал) з шаблоном. Можна виділити кілька основних типів пошуку за шаблоном:

– Simple Matching. Метод ґрунтується на принципі поетапного сканування вихідного зображення шаблоном. При процесі сканування в кожному етапі часу аналізується і визначається ступінь відповідності зображення, що сканується. В результаті розпізнається область, яка має велику ступінь ідентичності шаблоном:

– Feature-based matching. Даний метод будується на відповідності оригінального зображення і шаблону за характерними рисами, що виражаються кривими, точками і моделями поверхонь зображення. Результатом відповідності є велика ступінь ідентичності оригіналу і шаблону за характерними рисами;

– Area-based matching. Кореляційний метод відповідності є комбінаційним методом, так як визначення відповідності будується на основі двох інших алгоритмів: відповідність з характерних рис (feature detection) та відповідність за шаблоном (template detection). В даному алгоритмі відповідність визначається за піксельною відповідністю вихідного зображення і шаблону.

При вимірі відповідності розраховуються величини, що містять інформацію про параметри контрастності, освітленості або подібності образів. Цими величинами є власний простір і власне значення. Дані величини відіграють важливу роль для знаходження відповідності при неможливості визначення схожості за параметрами характерних рис і шаблону [11].

2.2.3 Кореляція зображень (Image Correlation Matching)

Даний метод використовує параметри Similarity Metrics (метрики подібності) для порівняння даних вихідного зображення з певним шаблоном. Перевагою кореляційного методу є незалежність від інтенсивності зображення та незначного рівня шуму, завдяки метрикам подібності. Порівняння вихідного зображення з шаблоном відбувається на основі їх кореляції.

Використання можливостей комп'ютерного зору безпосередньо в процесі виробництва продукції збільшує точність виконання виробничих операцій,

контролю якості виготовлення продукції, підвищення швидкості виконання операцій.

2.2.4 Глибоке машинне навчання (Deep learning)

Застосування методу глибокого навчання вимагає використання великого масиву даних. У комп'ютерному зорі дана вимога досягається використанням комбінацією елементів масиву пікселів.

Глибоке навчання (Deep Learning) має кілька видів навчання:

– контрольоване навчання – це навчання з учителем (supervised learning) має на увазі навчання структури, що містять вхідні дані і очікувані вихідні дані (результати навчання). Метод навчання з учителем є ітераційним процесом. Поетапне навчання моделі нейронної мережі припиняється тільки тоді, коли мережа перестає робити помилки:

– неконтрольоване навчання – навчання без вчителя (unsupervised learning). На відміну від контрольованого навчання метод навчання без вчителя використовує набір даних з невизначеною структурою. В процесі навчання мережу логічно класифікують відповідно вихідним даним і далі вона автоматично визначає вихідний результат.

Саме використання в системі CV алгоритму обробки зображення, за допомогою моделі глибокого неконтрольованого навчання (unsupervised feature learning) є найбільш ефективним при роботі в промисловому середовище, де є велика вірогідність появи неочікуваних сторонніх предметів в робочій зоні маніпулятора.

2.3 Аналіз методів і алгоритмів обробки зображень системами комп'ютерного зору

Існують кілька типів систем комп'ютерного зору. Одні системи є «однорідними» системами і працюють автономно, виконуючи функції виявлення, розпізнання, детектування і т.д. Інші технології виконують роль підсистеми в більшій системі, наприклад в автоматичних промислових маніпуляторах. Однак найчастіше системами комп'ютерного зору виконують типові функції: отримання зображень, попередня обробка, детектування і сегментація, подальша високорівнева обробка. Ці функції є основними етапами роботи модуля обробки зображень:

– цифрові зображення, отримані за допомогою ультразвукових і світлочутливих камер і датчиків відстані й зображення, форматуються в двовимірні або тривимірні зображення, в залежності від методу фіксації зображення;

– перед роботою методів комп'ютерного зору необхідно попередньо обробити отримане зображення, необхідно провести ряд заходів пов'язаних з фільрацією спотворень і поліпшення якості вихідного зображення для чіткого і швидкого доступу до необхідної інформації;

– локалізація і детектування деталей на зображенні. Залежно від рівня складності, деталі на зображенні виділяються різними формами (лінії, локалізовані точки кордонів, різні геометричні форми та інше) і сегментуються на різні пріоритетні групи, що містять важливу інформацію про різні об'єкти (набір точок) дослідження;

– завершальним етапом є високорівнева обробка. Даний етап полягає в перевірці всіх вимог і умов (положення і розмір об'єкта, інші характерні параметри, наявність класифікації за категоріями і т.д.), що відносяться до вхідних даних (набір пріоритетних точок, або ділянки зображення).

Основним завданням в системах комп'ютерного зору, особливо на стадії обробки зображень, є визначення і розпізнавання особливостей характерних

точок або об'єкта. У разі комп'ютерного зору рішення даного завдання мають деякі труднощі і, в загальному випадку, є приватними або точковими рішеннями (визначення випадкового об'єкта в випадкових ситуаціях).

Таким чином, методи вирішення класичного завдання є більш ефективними тільки для визначення і обробки окремих об'єктів (наприклад, геометричні фігури, людські обличчя, автомобільні номери, символи і т.д.).

Класичні методи розпізнавання, ідентифікації і виявлення об'єкта в системах комп'ютерного зору здійснюються різними методами і алгоритмами: створення класів об'єктів, використання бази даних для попереднього навчання та ін.

Також використовуються спеціалізовані завдання, засновані на розпізнаванні:

– пошук зображень за структурою змісту, наприклад, знаходження всіх зображень у великому наборі зображень, які мають певний зміст;

– оцінка і аналіз стану об'єкта, наприклад, допомога промисловому роботу при маніпуляції об'єктами на конвеєрної лінії;

– розпізнавання символів і знаків на зображеннях, наприклад, розпізнавання текстових символів.

Існують завдання розпізнавання об'єкта в процесі його руху. Для оцінки і аналізу руху враховуються деякі динамічні характеристики певної досліджуваної системи:

– оцінка швидкості кожної досліджуваної точки на зображенні;

– стеження за динамічними змінами параметрів об'єкта;

– розрахунок 3-х мірного переміщення камери. [10]

Існує три основних типи систем CV:

– одномірні (1D);

– двовимірні (2D);

– об'ємні (3D) системи CV.

Методи обробки зображень:

– лічильник пікселів;

– бінаризація;

– сегментація;

– читання штрих-кодів;

– оптичне розпізнавання символів;

– вимір;

– виявлення країв;

– зіставлення шаблонів.

2.4 Висновки за результатами виконання другого розділу роботи

В результаті виконання другого розділу атестаційної роботи проведено аналіз основних задач, що виконуються системами комп'ютерного зору, розглянуті основні методи обробки зображень. Виконано аналіз методів і алгоритмів обробки зображень системами комп'ютерного зору. Проаналізовані переваги та недоліки використання основних методів для обробки зображень. В якості основного методу обрано контурний аналіз.

3 РОЗРОБКА МЕТОДУ ПОЗИЦІОНУВАННЯ МАНІПУЛЯТОРА

3.1 Розробка структурної схеми системи позиціонування Розробка структурної схеми системи позиціонування

В систему позиціонування маніпулятора входять такі модулі:

– центральний модуль керування;

– модуль керування маніпулятором;

– відеокамера;

– маніпулятор.

Структурна схема системи позиціонування маніпулятора показана на рисунку 3.1.

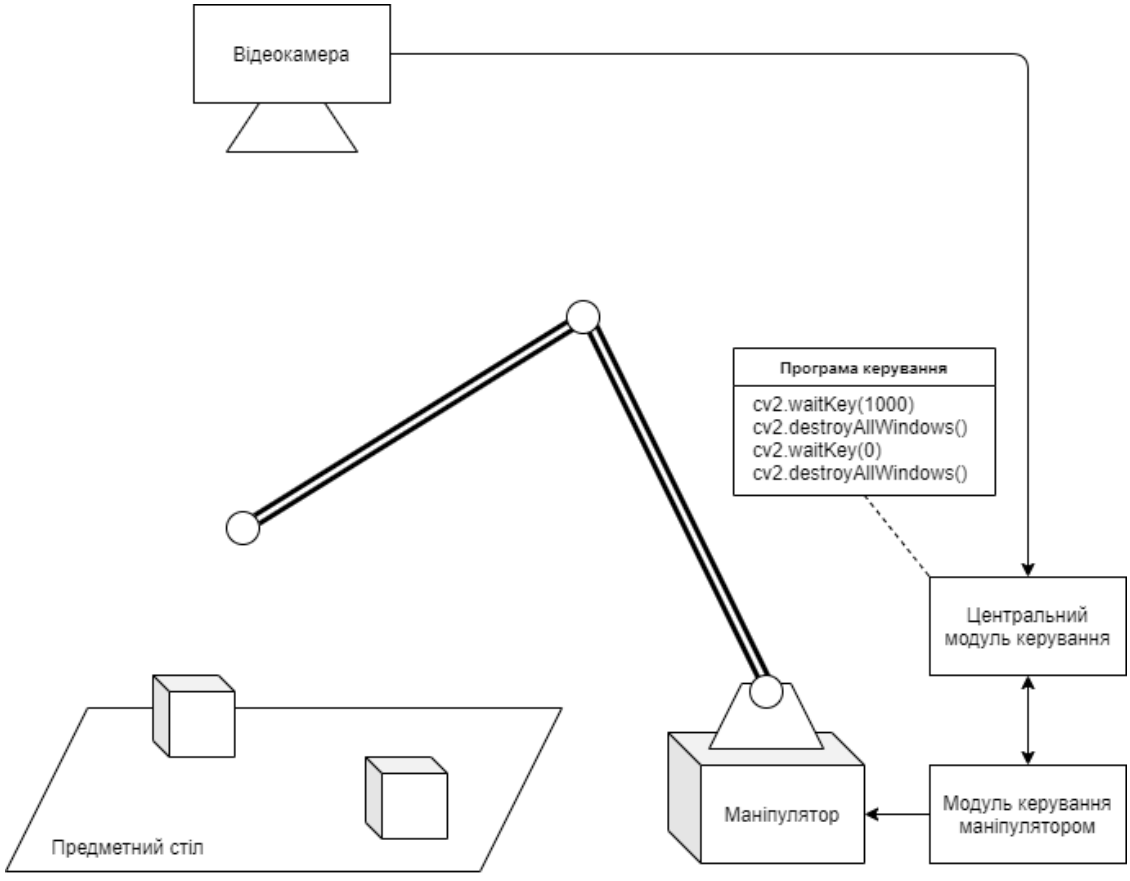


Рисунок 3.1 – Структурна схема системи позиціонування маніпулятора

В якості центрального модуля керування можна використовувати одноплатний комп’ютер Raspberry, або персональний комп’ютер. На даному модулі запускається розроблена програма, що виконує такі функції:

– отримання зображення з відеокамери;

– попередня обробка зображення;

– аналіз зображення;

– визначення координат розташування предметів на столі перед маніпулятором;

– формування команди управління маніпулятором;

– контроль за виконанням команд маніпулятором;

– формування звіту для користувача.

Модуль керування маніпулятором отримує команди від центрального модуля керування, визнає поточне положення рухомих компонентів та управляє механізмами пристрою.

На рисунку 3.2 показана функціональна схема системи позиціонування маніпулятора.

Всі функції системи позиціонування маніпулятора розподілені між двома пристроями: центральним модулем керування та модулем керування маніпулятором.

Центральний модуль керування безпосередньо взаємодіє з відеокамерою. Блок аналізу зображення виконує функції попередньої обробки отриманого зображення, фільтрації та бінарізації.

Блок визначення об’єктів на основі підготовленого для роботи зображення за обраним методом детектування виділяє контури об’єктів, що розташовані на предметному столі в зоні дії маніпулятора.

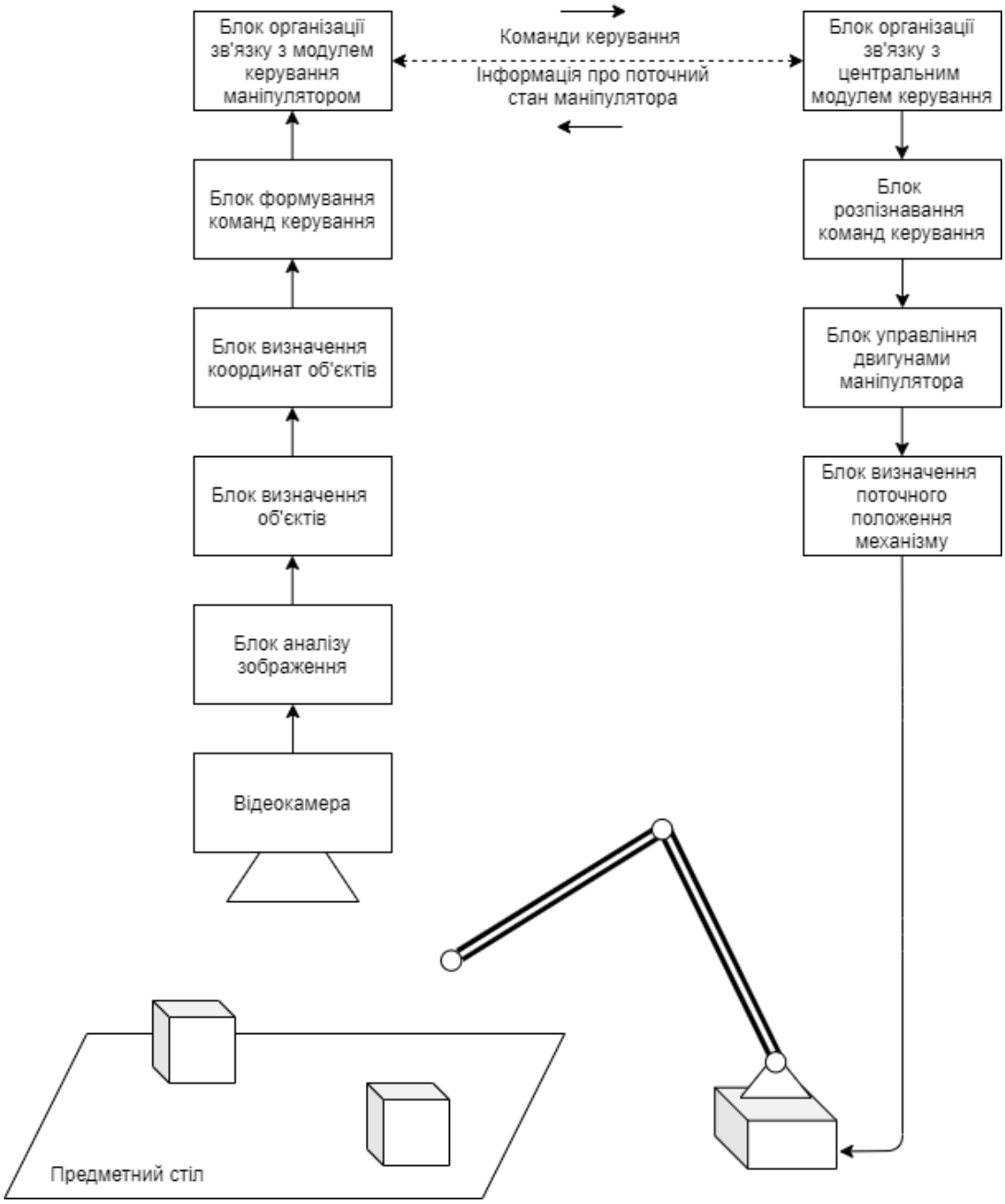


Рисунок 3.2 – Функціональна схема системи позиціонування маніпулятора

Блок визначення координат знаходить геометричні центри локалізованих на зображені об’єктів. Отримані координати становлять основу для блоку формування команд керування маніпулятором. Даний блок використовує в якості інструменту опису необхідних дій маніпулятора G-Code. Дана мова є стандартом для організації обміну даними між промисловим обладнанням.

Функцію організації зв’язку з блоком керування маніпулятором виконує відповідний блок. Від центрального модулю надходять команди керування маніпулятором. В зворотному напрямку поступає інформація про поточний стан пристрою.

Модуль керування маніпулятором виконує такі функції:

– отримання команд від центрального модуля керування;

– розбір команд та визначення траєкторії руху;

– управління кроковими двигунами для переміщення захвату в потрібне місце;

– визначення поточних координат робочого інструменту та передача цієї інформації на центральний модуль керування.

3.2 Перетворення координат розміщення об’єктів в координатну систему робота

Розглянемо дві системи координат: систему координат маніпулятора OXYZ з осями OX, OY, OZ і систему координат камери OUVW з осями OU, OV, OW. На рисунку 3.3 показано принцип поєднання координат.

Однорідна матриця перетворення являє собою матрицю розмірністю 4x4, яка перетворює вектор, виражений в однорідних координатах, з однієї системи відліку в іншу. Однорідна матриця перетворення може бути розбита на чотири підматриці:

$$T = \begin{bmatrix} R_{3x3} & P_{3x1} \\ f_{1x3} & S_{3x3} \end{bmatrix}. \quad (3.1)$$

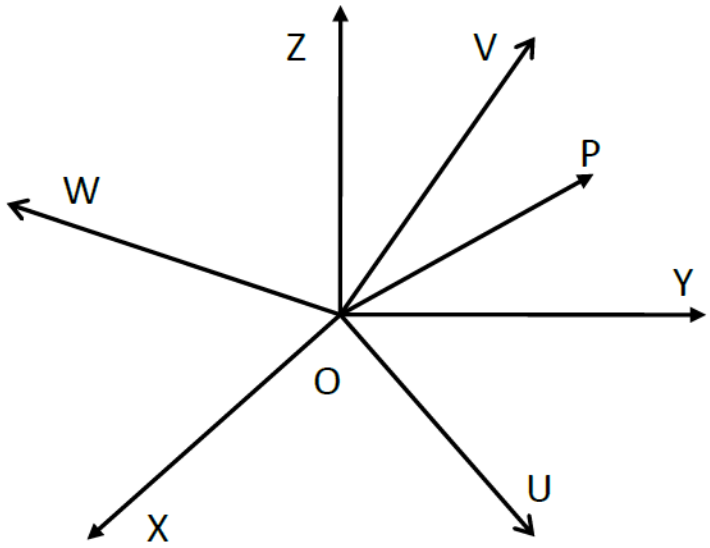


Рисунок 3.2 – Загальна система координат маніпулятора та камери

Верхня ліва підматриця розмірністю 3x3 являє собою матрицю повороту. Верхня права підматриця розмірністю 3x1 являє собою вектор положення початку координат поверненою системи відліку щодо абсолютної. Нижня ліва матриця розмірністю 1x3 задає перетворення перспективи. Нижня права підматриця є глобальним маштабуючим множником. Однорідна матриця перетворення дозволяє виявити геометричний зв'язок між зв'язаною системою відліку OUVW і абсолютної системою OXYZ.

Якщо вектор P тривимірного простору виражений в однорідних координатах, тобто

$$P = (px, py, pz)\ T, \tag{3.2}$$

то, використовуючи поняття матриці перетворення, можна сформувати однорідну матрицю перетворення задану перетворенням повороту навколо вектора k на кут Q і має розмірність 4x4 . Однорідна матриця повороту виходить відповідним розширенням звичайної матриці повороту, що має розмірність 3x3.

Так однорідне перетворення, яке описує поворот навколо осі X на кут β матиме вигляд:

$$ROT(k,Q) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\beta & -sin\beta & 0 \\ 0 & sin\beta & cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$

Поворот навколо осі Y на кут β матиме вигляд:

$$ROT(k,Q) = \begin{bmatrix} cos\beta & 0 & sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\beta & sin\beta & cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

Поворот навколо осі Z на кут β матиме вигляд:

$$ROT(k,Q) = \begin{bmatrix} cos\beta & -sin\beta & 0 & 0 \\ sin\beta & cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.5)$$

Верхня права подматріця однорідної матриці перетворення, що має розмірність 3x1, задає паралельний перенос системи координат OUVW щодо абсолютної системи OXYZ на вектор $(dx, dy, dz)\ T$

$$TRANS = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.6)$$

Ця матриця розмірністю 4x4 називається однорідною матрицею елементарного зсуву.

Права нижня подматріця однорідної матриці перетворення визначає глобальне перетворення масштабу

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ s \end{bmatrix}, \tag{3.7}$$

де s > 0. Таким чином, четвертий діагональний елемент однорідної матриці перетворення визначає глобальне стиснення координат, якщо s > 1, і розтягнення, якщо 0 < s < 1.

Ліва нижня підматриця однорідної матриці перетворення розмірністю 1x3 визначає перетворення перспективи.

3.3 Вибір засобів програмної реалізації методів комп’ютерного зору

3.1.1 Бібліотека OpenCV

В даний час реалізація методів комп'ютерного зору ґрунтується на готових фреймворках, які представлені у вигляді набору бібліотек для програмного забезпечення. Найбільш відомою мовою програмування для реалізації комп'ютерного зору є Python. Ця мова поєднує в своїй структурі необхідні інструменти і засоби для реалізації системи комп’ютерного зору. Наприклад, для рішення задач розпізнавання об’єктів використовується бібліотека OpenCV з великим набором вбудованих функцій обробки, детектування і розпізнавання зображень [13].

На рисунку 3.3 показана приклад реалізації архітектури бібліотеки OpenCV.

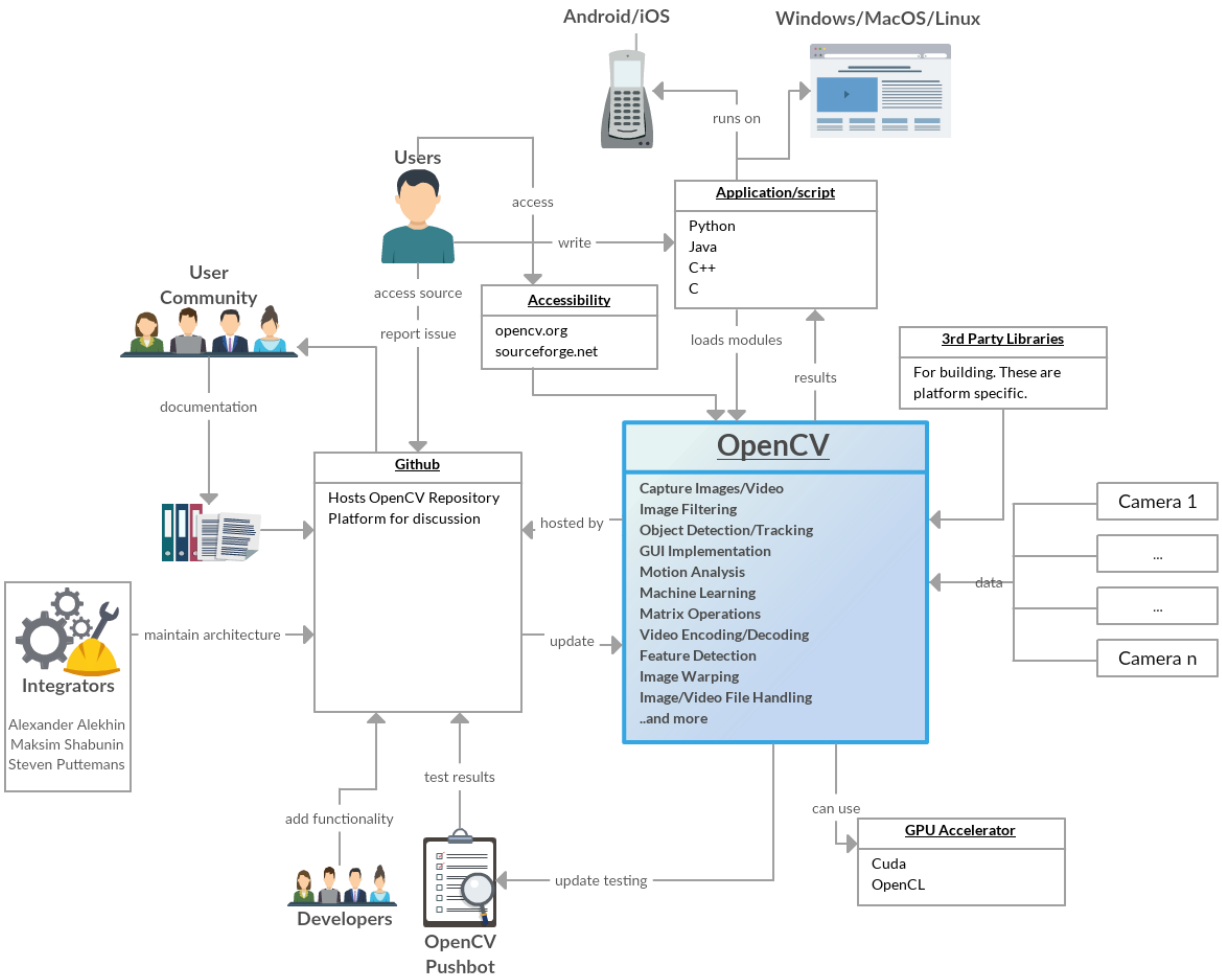


Рисунок 3.3 – Приклад реалізації архітектури бібліотеки OpenCV

OpenCV (Open Source Vision Library) – це бібліотека алгоритмів обробки зображень і комп'ютерного зору з відкритим вихідним кодом. OpenCV написана мовою високого рівня C++ і містить алгоритми для обробки зображень, калібрування камери за зразком, усунення оптичних спотворень, визначення подібності, аналізу переміщення об'єкта, визначення форми об'єкту та стеження за об'єктом. Для роботи з зображеннями використовуються такі основні бібліотеки:

– opencv core – ядро бібліотеки. Включає в себе базові структури, обчислення (математичні функції, генератори випадкових чисел) і лінійну алгебру і т.д.;

– opencv imgproc – обробка зображень (фільтрація, геометричні перетворення, перетворення колірних просторів);

– opencv_highgui – простий інтерфейс користувача для введення/виведення зображень і відео;

– opencv_ml – моделі машинного навчання:

– opencv_features2d – розпізнавання і опис плоских примітивів:

– opencv_video – аналіз руху і відстеження об'єктів (оптичний потік, шаблони руху, усунення фону):

– opencv_objdetect – виявлення об'єктів на зображенні (перебування осіб за допомогою алгоритму Віоли-Джонса, розпізнавання людей алгоритмом HOG пунктів);

– opencv_calib3d – калібрування камери і елементи обробки тривимірних даних.

Аналіз функцій Python для роботи із зображеннями.

NumPy (Numeric Python) – це бібліотека дозволяє швидко і зручно працювати з великими багатовимірними масивами. Основні можливості бібліотеки:

– підтримка високорівневих математичних функцій;

– робота з багатовимірними масивами, включаючи матриці.

Бібліотека NumPy дозволяє реалізацію обчислювальних алгоритмів, які оптимізовані для роботи з багатовимірними масивами. В результаті будь-який алгоритм, який може бути виражений у вигляді послідовності операцій над масивами (матрицями) і реалізований з використанням NumPy, працює так само швидко, як еквівалентний код, що виконується в MATLAB. Таким чином, можна вважати NumPy, як вільну альтернативу MATLAB.

SciPy (Scientific Python) – це пакет прикладних математичних процедур, заснований на розширенні Numpy Python. Ця бібліотека містить такі модулі як фізичні константи, перетворення Фур'є, інтегрування і інтерполяції, лінійна алгебра, статистика. Даний модуль може біти застосований для рішення наступних задач:

– пошук мінімумів і максимумів функцій;

– обчислення інтегралів функцій;

– рішення диференціальних рівнянь;

– обробка сигналів;

– обробка зображень;

– робота з генетичними алгоритмами.

Matplotlib – ця бібліотека представляє модуль, необхідний для візуалізації даних 2D і 3D діаграмами і створення високоякісних графічних даних різних форматів.

Matplotlib є гнучким, легко конфігурованим пакетом, який разом з NumPy і SciPy надає програмі можливості, подібні MATLAB.

Пакет підтримує багато видів графіків і діаграм:

– графіки (line plot);

– діаграми розкиду (scatter plot);

– стовпчасті діаграми (bar chart);

– гістограми (histogram);

– кругові діаграми (pie chart);

– поля градієнтів (quiver);

– спектральні діаграми (spectrogram).

Типові підтримувані формати зображень: EPS, EMF, JPEG, PDF, PNG, Postscript, RGBA, SVG, SVGZ, TIFF.

Scikit-learn – це бібліотека, яка надає реалізацію цілого ряду алгоритмів для навчання з учителем (Supervised Learning) і навчання без вчителя (Unsupervised Learning) через інтерфейс для мови програмування Python.

Даний модуль включає в себе:

– IPython: інтерактивна оболонка для мови програмування Python, яка надає розширену можливість керування ходом виконання програми, додатковий командний синтаксис, підсвічування коду і автоматичне доповнення;

– Sympy: бібліотека для символьних обчислень;

– Pandas: різні структури даних і аналіз.

Бібліотека scikit-learn реалізує рішення деяких популярних методів і завдань:

– кластеризація (Clustering): для угруповання нерозмічену даних, наприклад, метод k-середніх (k-means);

– перехресна перевірка (Cross Validation): для оцінки ефективності роботи моделі на незалежних даних;

– набори даних (Datasets): для тестових наборів даних і для генерації наборів даних з певними властивостями для дослідження поведінкових властивостей моделі;

– скорочення розмірності (Dimensionality Reduction): для зменшення кількості атрибутів для візуалізації та відбору ознак (Feature Selection);

– алгоритмічні композиції (Ensemble Methods): для комбінування прогнозів декількох моделей;

– витяг ознак (Feature Extraction): визначення атрибутів в зображеннях і текстових даних;

– відбір ознак (Feature Selection): для виявлення значущих атрибутів на основі яких буде побудована модель;

– оптимізація параметрів алгоритму (Parameter Tuning): для отримання максимально ефективної віддачі від моделі;

– множинне навчання (Manifold Learning): для нелінійного скорочення розмірності даних;

– алгоритми навчання з вчителем (Supervised Models).

3.1.2 Програмні забезпечення (фреймворки) для реалізації завдань комп'ютерного зору.

Tensor-flow – це відкрита програмна платформа машинного навчання для вирішення завдань побудови і тренування нейронних мереж з метою автоматичного знаходження та класифікації образів.

Caffe (Convolution Architecture For Feature Extraction) – це відкрите програмне забезпечення для реалізації методів deep learning. Caffe має підтримку багатьох видів машинного навчання, які вирішують завдання класифікації і сегментації образів. Основними достоїнствами середовища Caffe є підтримка

алгоритмів згортальних нейронних мереж (CNN), довгу короткострокову пам'ять, повнозв'язні нейронні мережі та підтримка систем графічних процесорів (GPU) для збільшення швидкості навчання.

CatBoost – це бібліотека для реалізації глибокого навчання градиентного бустінга на деревах рішень, тим вона і відрізняється від Tensor-flow і Caffe, які ґрунтуються на нейронних мережах.

YOLO (You Only Look Once) – це сучасна і одна з потужних систем із виявлення об'єктів в реальному часі. Є модифікації цієї системи для вирішення завдань виявлення об'єктів з різним ступенем потужності і апаратної області використання. Завдяки широкому діапазону доступних варіантів можна вибрати версію, найбільш підходящу для ваших потреб. Наприклад, Tiny YOLO – це самий «компактний» варіант, який може працювати швидко навіть на смартфонах або Raspberry Pi. Оригінальна архітектура YOLOv3 реалізована за допомогою фреймворка DarkNet.

На рисунку 3.4 показана архітектура YOLOv3.

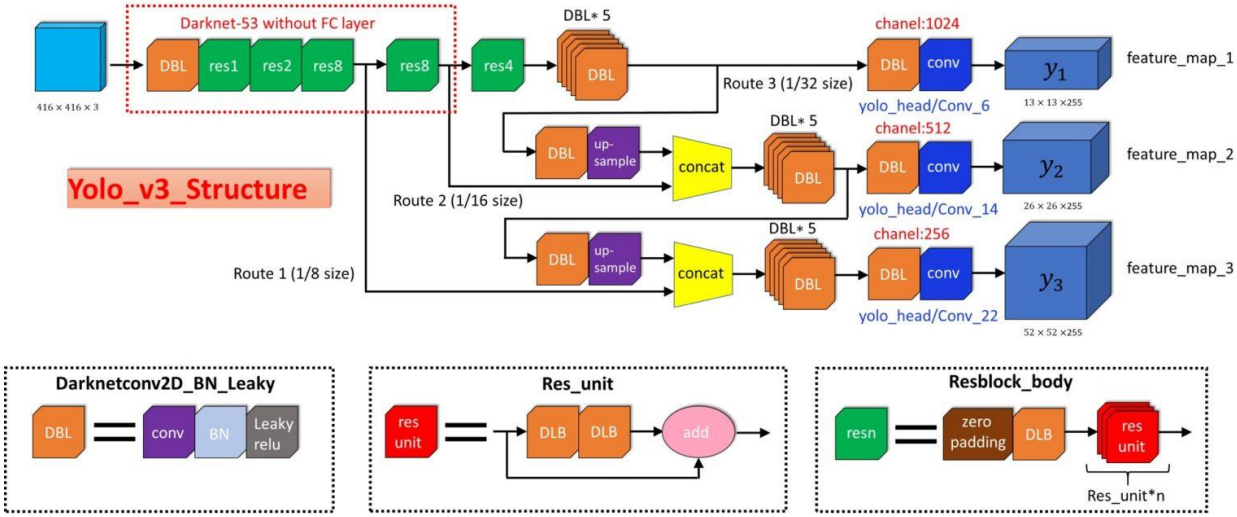


Рисунок 3.4 – Архітектура YOLOv3

YOLOv3 – це остання модифікація архітектури YOLO. Архітектура складається з 106 згортальних шарів (CNN). Дана версія відрізняється тим, що три останніх вихідних шару розраховані для виявлення об'єктів різного формату (розміру).

Головною перевагою даної моделі є її унікальний принцип роботи з зображеннями. Ця модель накладає на зображення сітку, розділяючи його на осередки. Кожна осередок намагається передбачити координати зони виявлення з оцінкою впевненості для цих полів і ймовірністю класів. Потім оцінка впевненості для кожної зони виявлення множиться на ймовірність класу, щоб отримати остаточну оцінку.

3.4 Висновки за результатами виконання третього розділу роботи

В результаті виконання третього розділу атестаційної роботи розроблено структурну схему системи позиціонування суглобів маніпулятора. В якості центрального модуля керування можна використовувати одноплатний комп’ютер Raspberry, або персональний комп’ютер. Розроблено функціональну схему системи позиціонування маніпулятора.

Наведено відомості про метод перетворення координат розміщення об’єктів на предметному столі в координатну систему робота для використання його в розділі експериментальних досліджень.

Виконано вибір засобів програмної реалізації методів комп’ютерного зору. В якості основної бібліотеки обрано OpenCV. Наведено приклад реалізації архітектури бібліотеки OpenCV та розглянуті основні функції, що використовуються для обробки зображень.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

4.1 Планування проведення експерименту

Експериментальні дослідження направлені на визначення дієвості розробленого методу позиціонування маніпулятора за допомогою системи комп’ютерного зору. Також планується провести дослідження точності визначення координат розміщення об’єкта на предметному столі.

Для проведення експериментальних досліджень необхідно розробити програмний засіб за допомогою якого буде проводитись наступні дії:

– визначення контурів об’єкту;

– визначення координат розміщення об’єкту;

– визначення типу об’єкту;

– формування послідовності команди для керування рухом маніпулятора;

– контроль за переміщенням маніпулятора.

4.2 Розробка програмного засобу

Для роботи із зображенням будемо використовувати бібліотеку Emgu CV.

Emgu CV – крос-платформна .NET обгортка для відкритої бібліотеки зображень Open CV. Emgu CV дозволяє відкривати CV функції, які можуть бути викликані з .NET-сумісних мов. Обгортка може бути використана у Visual Studio, Unity та безпосередньо з командної строки через "dotnet", вона може працювати на ОС Windows, Mac OS, Linux, iOS та Android.

Одна з цілей Emgu CV є забезпечення простоти використання інфраструктури комп'ютерного зору для .NET програмістів. За допомогою неї можна швидко створювати досить складні програми, що використовують функції комп’ютерного зору. Бібліотека Emgu CV охоплює багато областей у

сфері обробки зображень, включаючи інспекцію заводських продуктів, оброку медичних зображення, створення інтерфейсів користувачів, калібрування камери, вирішення задачі стереозору в робототехніці. Оскільки комп'ютерне бачення часто застосовується у промисловості, Emgu CV пропонує повну бібліотеку функцій загального призначення для вирішення задачі автоматизації станків із використанням бібліотеки OpenCV.

Архітектура Emgu CV представлена на рисунку 4.1.

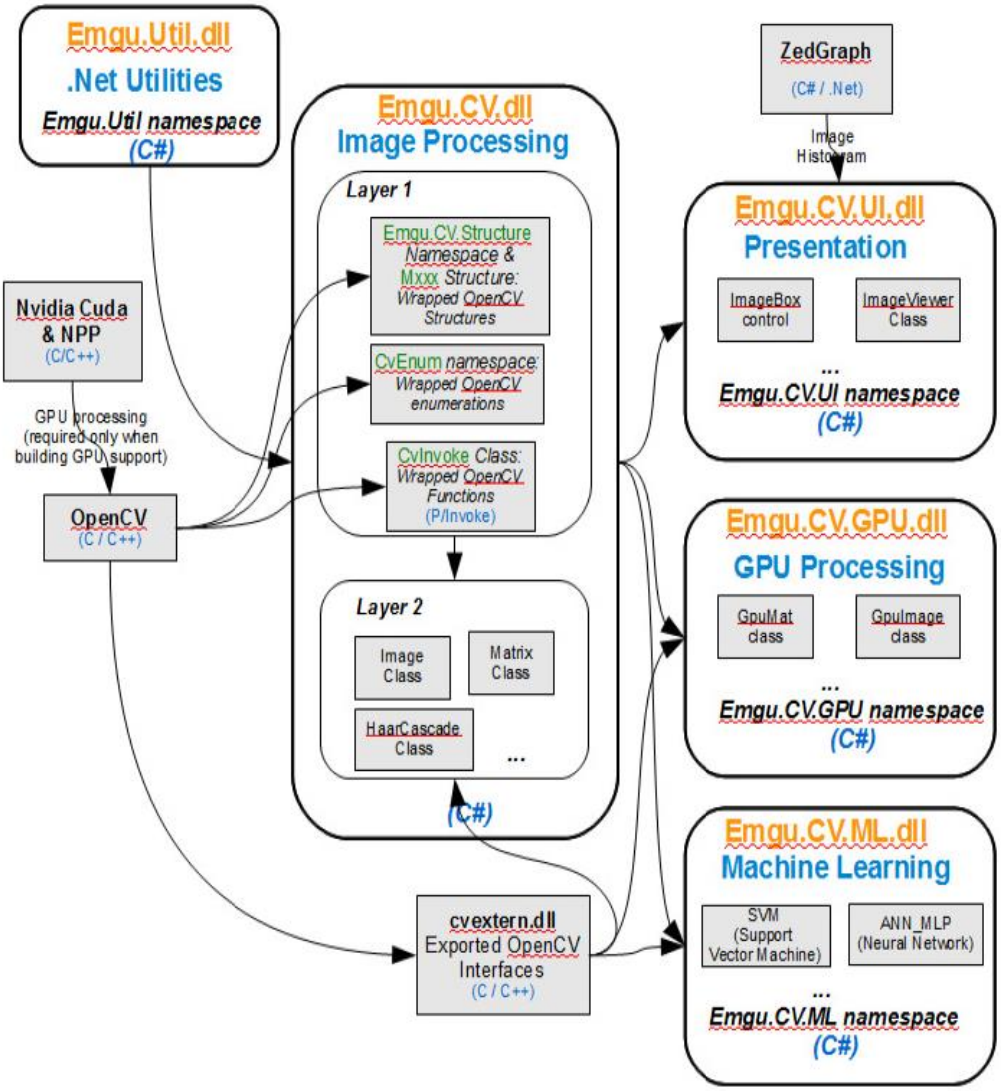


Рисунок 4.1 – Архітектура Emgu CV

Emgu CV має два шари обгортки, як показано нижче:

– основний шар (шар 1) містить функції, структури та переліки, які безпосередньо відображаються в OpenCV;

– другий шар (шар 2) містить класи, які змішуються з класами .NET.

Наприклад, клас CVLnvoke забезпечує спосіб безпосереднього виклику функції OpenCV в мовах .NET. Кожен метод у цьому класі відповідає одноіменній функції в OpenCV.

Для використання даної бібліотеки в своїй програмі спочатку необхідно додати необхідні файли до структури проекту. Зробимо це за допомогою інструменту Nuget. На рисунку 4.2 показано приклад обирання потрібної версії бібліотеки.

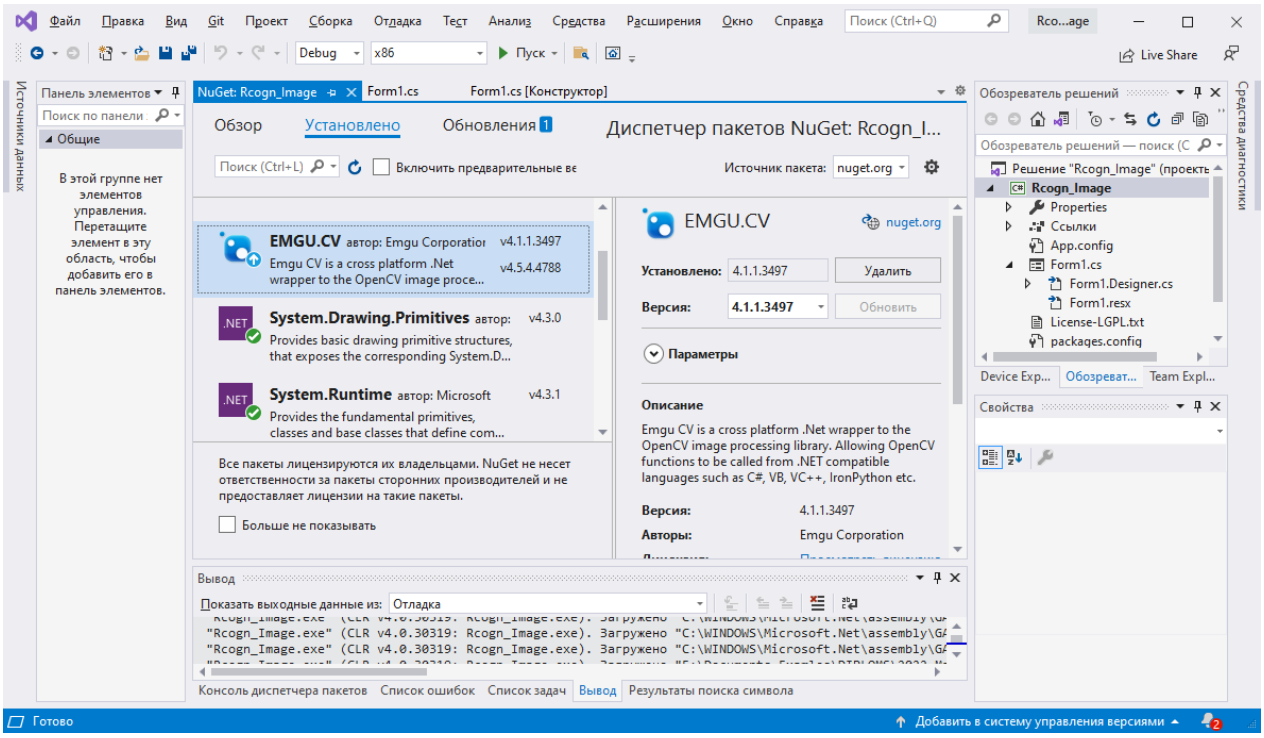


Рисунок 4.2 – Додавання Emgu CV до проекту

Після додавання бібліотеки можна виконати підключення необхідних класів до проекту. На рисунку 4.3 показано приклад використання класів Emgu, Emgu.CV, Emgu.CV.Structure, Emgu.CV.Util та Emgu.Util.

```
12      using Emgu;
13      using Emgu.CV;
14      using Emgu.CV.Structure;
15      using Emgu.CV.Util;
16      using Emgu.Util;
```

Рисунок 4.3 – Приклад підключення необхідних класів до проекту

На рисунку 4.4 показано приклад методу для пошуку контурів об’єктів на зображенні.

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        Image<Gray, byte> outputImage = inputImage.Convert<Gray, byte>().
            ThresholdBinary(new Gray(100), new Gray(255));
        VectorOfVectorOfPoint counturs = new VectorOfVectorOfPoint();
        Mat hierarhy = new Mat();
        CvInvoke.FindContours(outputImage, counturs, hierarhy, Emgu.CV.CvEnum.RetrType.Tree,
            Emgu.CV.CvEnum.ChainApproxMethod.ChainApproxSimple);
        CvInvoke.DrawContours(inputImage, counturs, -1, new MCvScalar(255, 0, 0), 3);
        pictureBox2.Image = inputImage.Bitmap;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Рисунок 4.4 – Пошук контурів об’єктів на зображенні

В дому методі спочатку виконується попередня обробка вхідного зображення. Для цього викликається метод Convert з параметром new Gray(100). Даний параметр перетворю зображення до сірого з порогом перетворення 100. Чим менший цей параметр, тим більше артефактів буде прибрано із зображення.

Далі створюється об’єкт counturs для зберігання всіх знайдених контурів на зображенні. Метод визначення контурів задається із переліку можливих значень Emgu.CV.CvEnum.RetrType. В даному прикладі обрано варіант Tree. Таким чином, будуть виділені всі можливі контури на зображенні. На рисунку 4.5, показано приклад роботи розробленого методу.

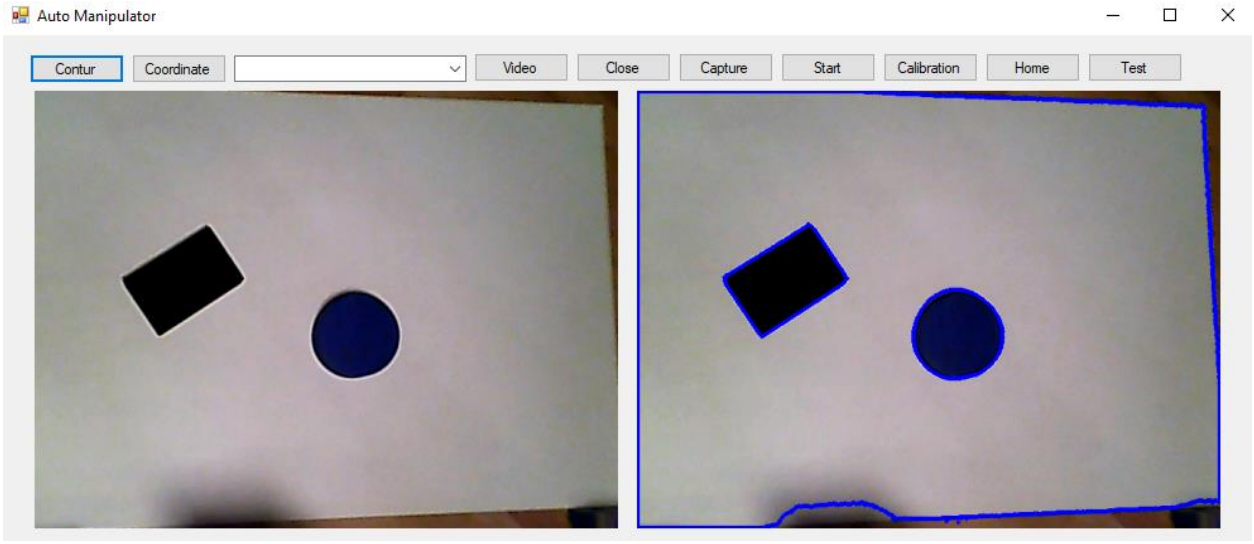


Рисунок 4.5 – Приклад розробленого методу визначення контурів об’єктів

Однією з перших задач є робота з відеокамерою. Для цього необхідно підключити відповідну бібліотеку. Серед багатьох варіантів була обрана DirecShowLib. На рисунку 4.6 показано приклад додавання даної бібліотеки до проекту.

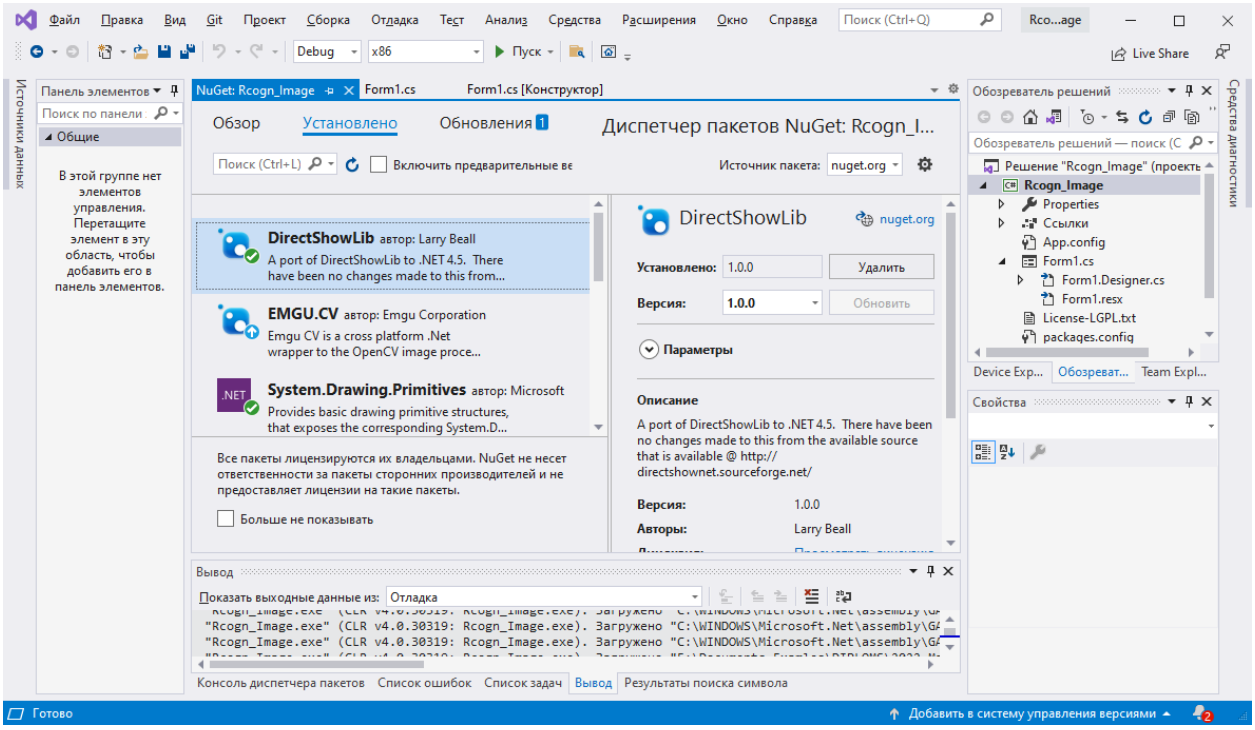


Рисунок 4.6 – Додавання бібліотеки DirecShowLib до проекту

Управління відеокамерами в програмі здійснюється за допомогою компоненту comboBox. Список доступних відеокамер заповнюється при старті програми за допомогою наступного фрагменту коду:

```
webCams = DsDevice.GetDevicesOfCat(FilterCategory.VideoInputDevice);
for (int i = 0; i < webCams.Length; i++)
{
    comboBox1.Items.Add(webCams[i].Name);
}
```

На рисунку 4.7 показано результат виконання фрагменту програми та розкриття заповненого списку.

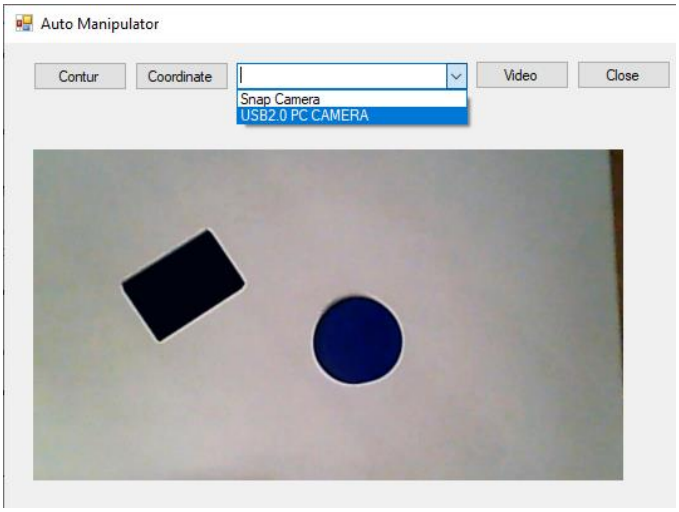


Рисунок 4.6 – Компонент вибору відеокамери

Для рішення задачі отримання зображення об’єктів на предметному столі обрана модифікована веб-камера. Модифікація полягала в видаленні захисного кожуха пристрою та отриманні друкованої плати з встановленою оптичною системою безпосередньо на чутливій матриці. На рисунку 4.7 показано зовнішній вигляд камери та варіант її встановлення.

Рисунок 4.7 – Зовнішній вигляд камери та варіант її встановлення

Камера закріплена безпосередньо на штативі з лампою. Таким чином вирішена ще одна задача – забезпечення освітлення робочого місця де розташовано деталі.

Після отримання зображення необхідно вирішити наступну задачу – визначення типу та пошук центра об’єкту. Для цього була розроблена підпрограма, що викликається при натисканні на кнопку «Coordinate» на графічному інтерфейсі користувача.

Підпрограму можна умовно поділити на два блоки:

– пошук контурів на зображення та визначення центра об’єкту;

– визначення типу об’єкту.

На рисунку 4.8 показано фрагмент коду для пошуку контурів на зображенні та визначення центра об’єкту.

```
Image<Gray, byte> grayImage = inputImage.SmoothGaussian(5).Convert<Gray, byte>().
    ThresholdBinaryInv(new Gray(30), new Gray(255));
VectorOfVectorOfPoint counturs = new VectorOfVectorOfPoint();
Mat hierarhy = new Mat();
CvInvoke.FindContours(grayImage, counturs, hierarhy, Emgu.CV.CvEnum.RetrType.External,
    Emgu.CV.CvEnum.ChainApproxMethod.ChainApproxSimple);

for (int i = 0; i < counturs.Size; i++)
{
    double perimeter = CvInvoke.ArcLength(counturs[i], true);
    VectorOfPoint approximation = new VectorOfPoint();
    CvInvoke.ApproxPolyDP(counturs[i], approximation, 0.04 * perimeter, true);
    CvInvoke.DrawContours(inputImage, counturs, i, new MCvScalar(0, 0, 255), 2);

    //Find center
    Moments moments = CvInvoke.Moments(counturs[i]);
    int x = (int)(moments.M10 / moments.M00);
    int y = (int)(moments.M01 / moments.M00);

    if ((x < 50) || (x > 600)) continue;
    if ((y < 50) || (y > 450)) continue;
```

Рисунок 4.8 – Фрагмент коду для пошуку контурів на зображенні та визначення центра об’єкту

З рисунку 4.8 можна бачити, що при обробці зображення використовується поріг чутливості в 30 одиниць. Це дає можливість розрізняти контури об’єктів при недостатній освітленості, або якщо колір об’єкту на отриманому зображенні не досить контрастує з кольором фону.

Для визначення контурів обрано метод визначення тільки зовнішніх границь. Це дає можливість не враховувати особливості поверхні об’єкту при розпізнаванні.

Визначення моментів у обробці зображення виконується за таким методом: припустимо, що кожен піксель у зображенні має вагу, яка дорівнює його інтенсивності. Тоді точка, яку ми визначаємо є центроїд (центр мас) зображення [7].

Таким чином, координати центрів визначаються на основі відношення моментів всіх відрізків, що входять до складу кожного контуру:

int x = (int)(moments.M10 / moments.M00);

int y = (int)(moments.M01 / moments.M00).

Точки, що знаходяться за межами робочого поля зображення відсікаються:

if ((x < 50) || (x > 600)) continue;

if ((y < 50) || (y > 450)) continue;

На рисунку 4.9 показано фрагмент коду для визначення типу об’єкту.

```
if (approximation.Size == 3)
{
    CvInvoke.PutText(inputImage, "Triangle", new Point(x, y),
        Emgu.CV.CvEnum.FontFace.HersheyPlain, 1, new MCvScalar(255, 255, 255), 1);
}
else if (approximation.Size == 4)
{
    Rectangle rect = CvInvoke.BoundingRectangle(counturs[i]);
    double aspectRatio = (double)rect.Width / (double)rect.Height;
    if (aspectRatio >= 0.95 && aspectRatio <= 1.05)
    {
        string coordinate = String.Format("{0}.{1}", (int)((x - nullCoordinates.X) * scale_x), 0 -
            (int)((y - nullCoordinates.Y) * scale_y));
        CvInvoke.PutText(inputImage, coordinate, new Point(x - 70, y),
            Emgu.CV.CvEnum.FontFace.HersheyPlain, 2, new MCvScalar(255, 255, 255), 3);
        objectCoordinate = new Point((int)((x - nullCoordinates.X) * scale_x), 0 -
            (int)((y - nullCoordinates.Y) * scale_y));
        figure = "Square";
    }
    else
    {
        string coordinate = String.Format("{0}.{1}", (int)((x - nullCoordinates.X) * scale_x), 0 -
            (int)((y - nullCoordinates.Y) * scale_y));
        CvInvoke.PutText(inputImage, coordinate, new Point(x - 70, y),
            Emgu.CV.CvEnum.FontFace.HersheyPlain, 2, new MCvScalar(255, 255, 255), 3);
        objectCoordinate = new Point((int)((x - nullCoordinates.X) * scale_x), 0 -
            (int)((y - nullCoordinates.Y) * scale_y));
        figure = "Rectangle";
    }
}
else if (approximation.Size == 5)
{
    CvInvoke.PutText(inputImage, "Pentagon", new Point(x, y),
        Emgu.CV.CvEnum.FontFace.HersheyPlain, 1, new MCvScalar(255, 255, 255), 1);
}
else if (approximation.Size >= 6)
{
    string coordinate = String.Format("{0}.{1}", (int)((x - nullCoordinates.X) * scale_x), 0 -
        (int)((y - nullCoordinates.Y) * scale_y));
    CvInvoke.PutText(inputImage, coordinate, new Point(x - 70, y),
        Emgu.CV.CvEnum.FontFace.HersheyPlain, 2, new MCvScalar(255, 255, 255), 3);
    objectCoordinate = new Point((int)((x - nullCoordinates.X) * scale_x), 0 -
        (int)((y - nullCoordinates.Y) * scale_y));
    figure = "Circle";
}
```

Рисунок 4.9 – Фрагмент коду для визначення типу об’єкту

На даному рисунку можна бачити, що тип об’єкту визначається по кількості відрізків, що становлять визначений контур. Так, чотири відрізки дають

нам змогу визначити, що перед нами або прямокутник, або квадрат. Для більш точного визначення перевіряються співвідношення сторін a та b:

```
Rectangle rect = CvInvoke.BoundingRectangle(counturs[i]);
double aspectRatio = (double)rect.Width / (double)rect.Height;
if (aspectRatio >= 0.95 && aspectRatio <= 1.05) {…}
```

Круглим об’єкт вважається, якщо кількість відрізків контуру після апроксимації становить шість, або більше.

4.3 Виконання експериментальних досліджень

4.3.1 Дослідження правильності визначення форми об’єкту

Для виконання даного дослідження були використані чотири різні об’єкти:

– циліндрична шайба;

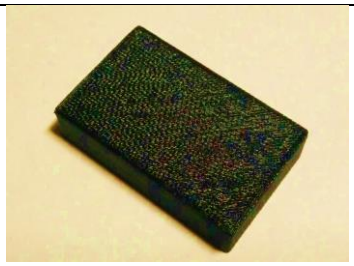
– прямокутна коробка;

– прямокутний фільтр для 3Д-принтера;

– квадратна заглушка для профіля.

Приклади тестових об’єктів представлені в таблиці 4.1.

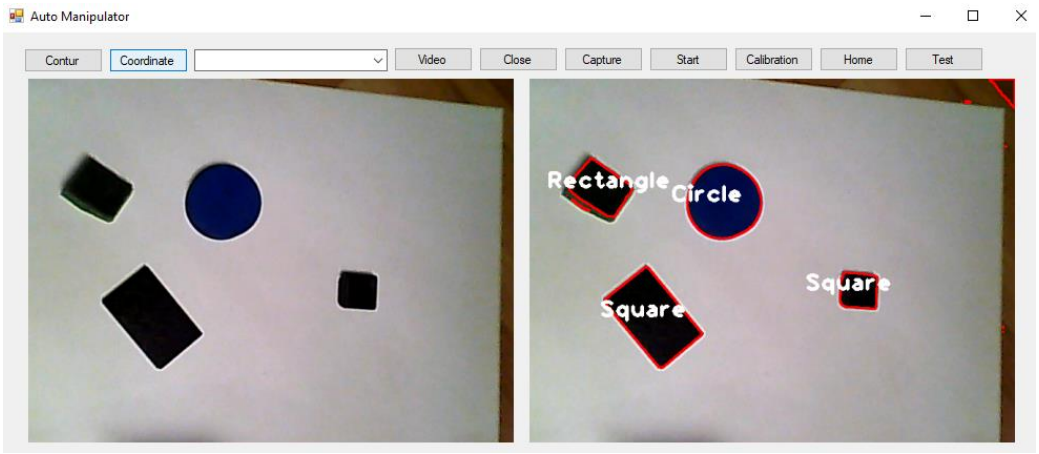
Таблиця 4.1 – Приклади тестових об’єктів

№ об’єкту	Тип об’єкту	Форма	Зовнішній вигляд об’єкту
1	Циліндрична шайба	Коло	
2	Прямокутна коробка	Прямокутник	
3	Фільтр для 3Д-принтера	Прямокутник	
4	Заглушка для профіля	Квадрат	

Після розміщення тестових об’єктів на предметному столі була завантажена розроблена програма та увімкнута функція захоплення відеоінформації. На рисунку 4.10 показані результати експерименту для перевірки правильності визначення типу фігури об’єкту. Можна бачити, що зліва розташовано оригінальне зображення з камери, а справа представлено оброблене

зображення з виділенням контурів всіх об’єктів, що потрапили до об’єктива та на кожному контурі вказано тип фігури.

а) результати першого експерименту



б) результати другого експерименту

Рисунок 4.10 – Результати експерименту для перевірки правильності визначення типу фігури об’єкту

Як можна бачити з проведеного експерименту, всі фігури розпізнані правильно. На правильність визначення типу форми об’єкту особливо впливає якість освітлення – необхідно щоб об’єкти не мали сторонніх артефактів (не утворювали тінь)

4.3.2 Дослідження точності визначення координат розташування об’єктів на предметному столі

Результати першого експерименту даного дослідження показані на рисунку 4.11.

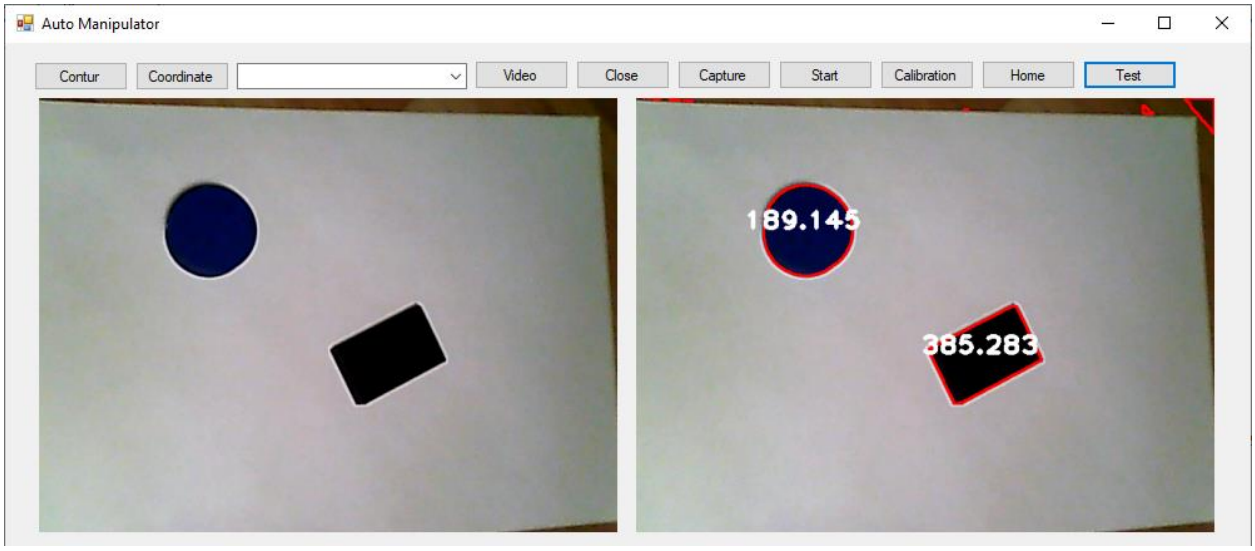


Рисунок 4.11 – Результати першого експерименту дослідження точності визначення координат розташування об’єктів на предметному столі

Як можна бачити з даного рисунку, кожен об’єкт визначений та має координати його розміщення на столі. Дані координати є локальними та обчислюються в пікселях по горизонталі та вертикалі захопленого кадру зображення. Тобто ці координати є, по суті, значенням розташування об’єкту на зображенні. Для керування маніпулятором необхідно перевести відстань в пікселях до міліметрів а надалі прив’язати локальні координати розташування об’єкту до координат самого маніпулятора.

Для вирішення даної задачі необхідно виконати наступні кроки:

– розробити метод калібрування оптичної системи;

– розробити програму для автоматичного трансформування локальних координат розміщення об’єктів в кадрі зображення до абсолютних координат роботи маніпулятора.

Принцип трансформації координат показано на рисунку 4.12.

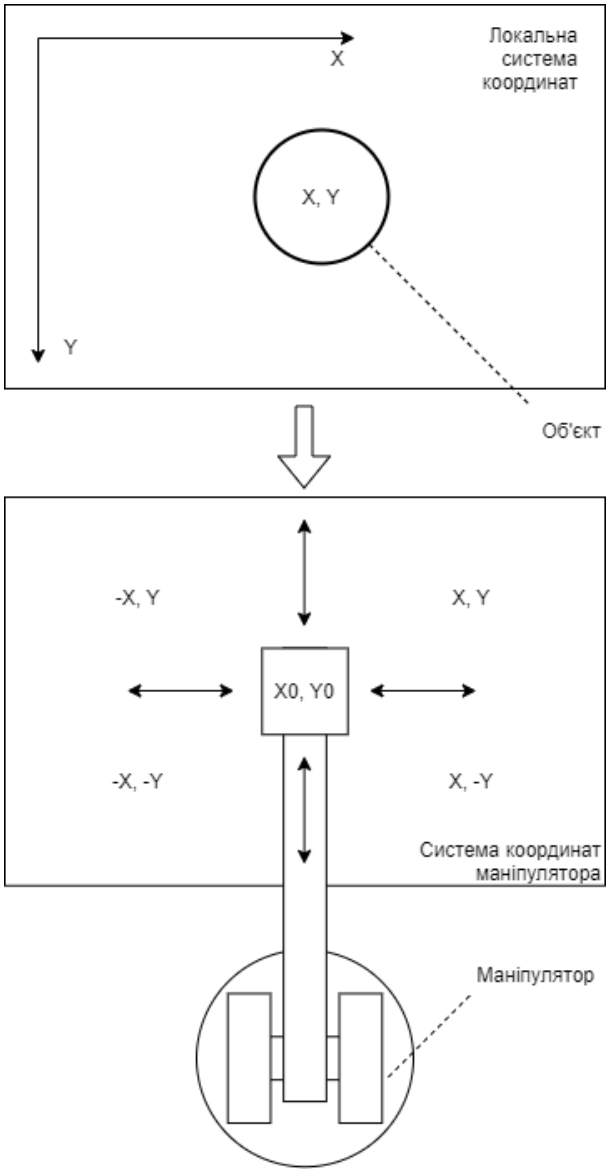


Рисунок 4.12 – Принцип трансформації локальних координат в координатну систему маніпулятора

Система координат маніпулятора відрізняється від локальної наявністю центральної нульової точки початку відліку та секторами з різними знаками параметрів. Нульова точка не є абсолютним нулем а представлена значеннями: X0 = 0, Y0 = 195. Тобто начальне положення захвату маніпулятора має зміщення по осі Y, що обумовлено конструктивними та програмними особливостями системи керування.

Для трансформування координат необхідно виконати процедуру калібрування оптичної системи. Для виконання цієї задачі розроблено тестове зображення, рисунок 4.13.

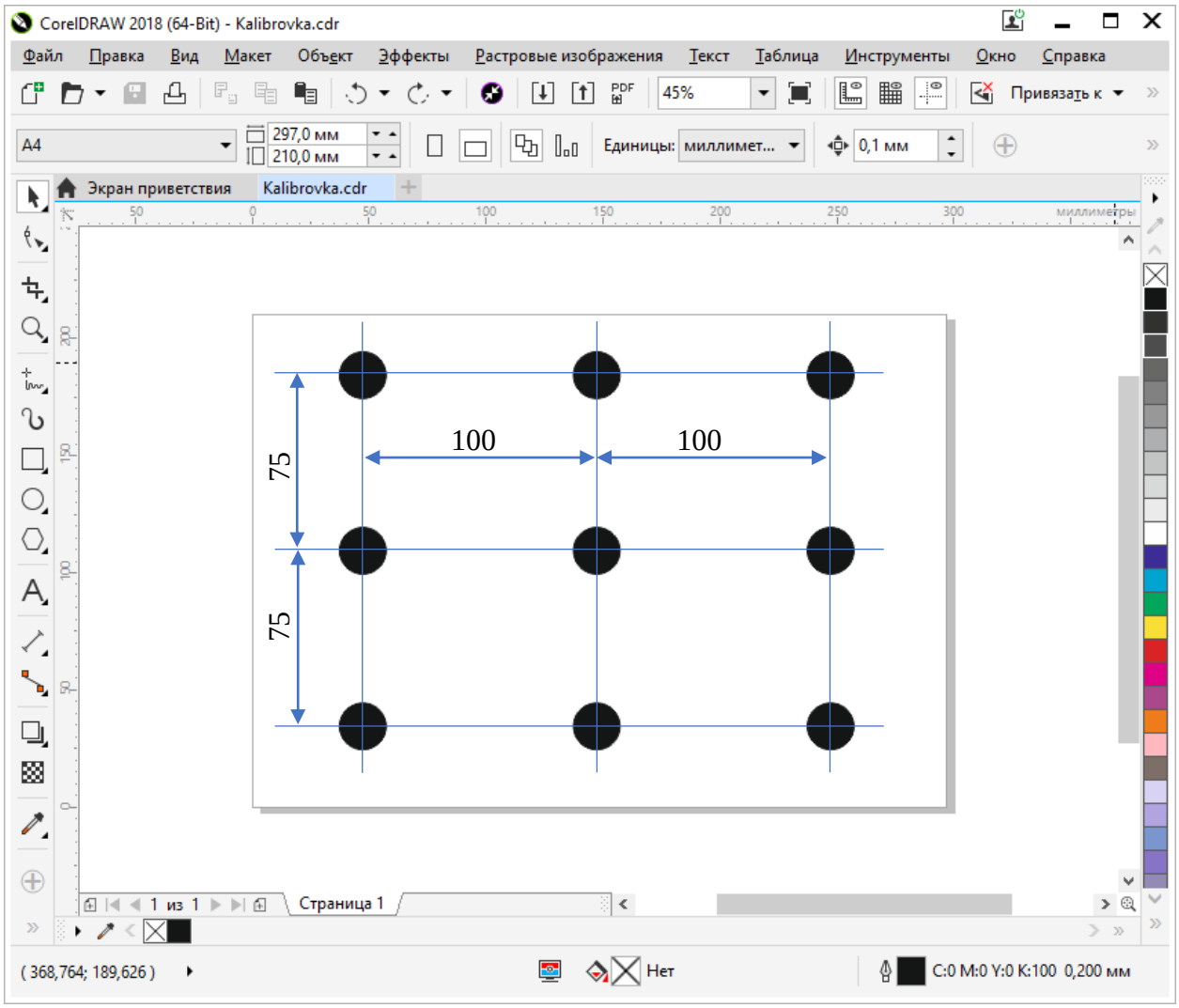


Рисунок 4.13 – Тестове зображення для виконання процедури калібрування

Кожна крапка діаметр 10 мм та по горизонталі розташовані на відстані 100 мм, а по вертикалі – 75 мм.

Для виконання процедури калібрування була розроблена підпрограма, фрагмент коду якої показано на рисунку 4.14.

```
for (int d = 0; d < 3; d++)
{
    pos = sortCoordinates_X(ref coordinatesList, ref sortList, 2, pos);
}

//3 fill rest
for (int d = 0; d < 3; d++)
{
    pos = sortCoordinates_X(ref coordinatesList, ref sortList, 3, pos);
}

int row = 0;
int col = 0;
for (int d = 0; d < 9; d++)
{
    coordinatesList[row, col] = sortList[d];
    row++;
    if (row == 3)
    {
        row = 0;
        col++;
    }
}

for (int d = 1; d < 4; d++)
{
    sortCoordinates_Y(ref coordinatesList, d);
}

nullCoordinates = coordinatesList[1, 1];
scale_x = (double)200 / (double)(coordinatesList[0, 2].X - coordinatesList[0, 0].X);
scale_y = (double)150 / (double)(coordinatesList[2, 1].Y - coordinatesList[0, 1].Y);

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        int x = coordinatesList[i, j].X;
        int y = coordinatesList[i, j].Y;
        string coordinate = String.Format("{0}.{1}", (int)((x - nullCoordinates.X) * scale_x), 0 - (int)((y - nullCoordinates.Y) * scale_y));
        CvInvoke.PutText(inputImage, coordinate, new Point(x - 70, y), Emgu.CV.CvEnum.FontFace.HersheyPlain, 2, new MCvScalar(255, 255, 255), 3);
    }
}
```

Рисунок 4.14 – Фрагмент коду для виконання процедури калібрування

Після запуску програми перед запуском процедури калібрування на екрані ПК отримаємо таке зображення, рисунок 4.15.

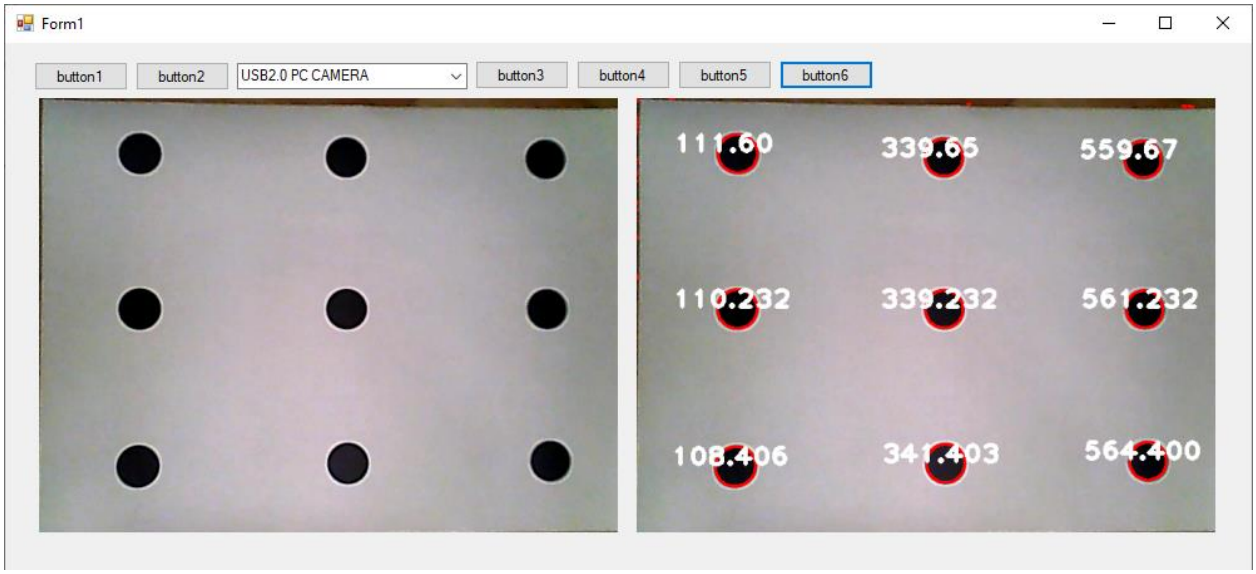


Рисунок 4.14 – Координати тестових об’єктів до калібрування

Після запуску процедури калібрування буде отримана матриця тестових координат, які потрібно трансформувати в абсолютну систему координат маніпулятора.

На рисунку 4.15 показано результат виконання даної процедури.

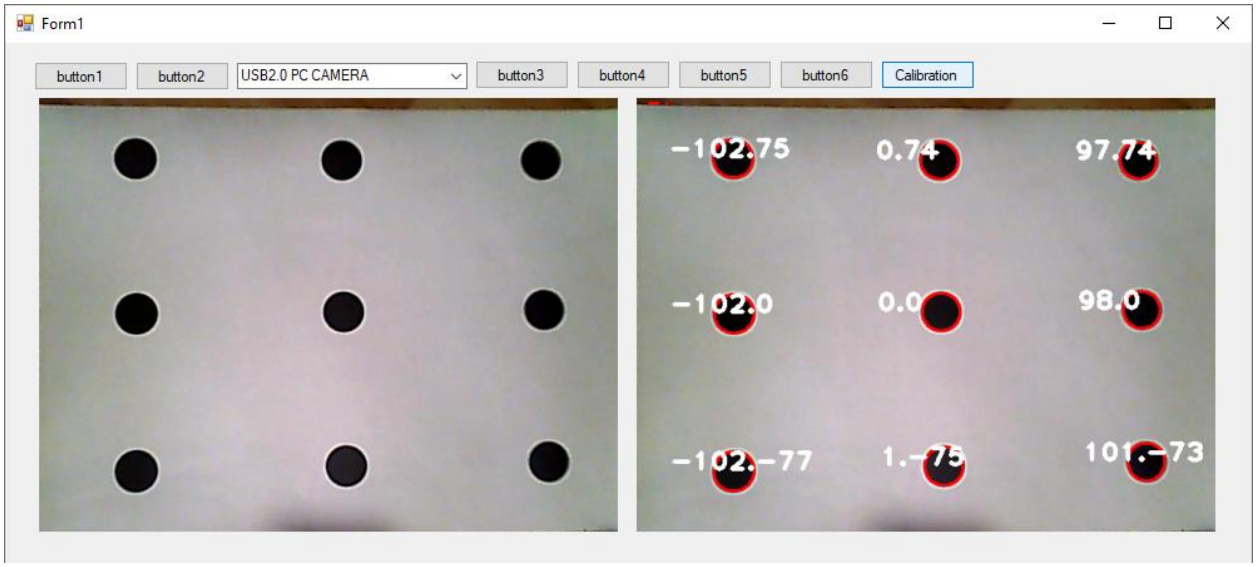


Рисунок 4.15 – Результат виконання процедури калібрування

Для перевірки правильності роботи запропонованого методу визначення координат розташування об’єкту на предметному столі тестовий малюнок було закрито, а на білому фоні в довільному вигляді розміщено прямокутну коробку із попереднього тесту. Результат даного експерименту показано на рисунку 4.16.

Як можна бачити з даного рисунку, форма об’єкту визначена правильно, а також визначені координати його розміщення на предметному столі з прив’язкою до системи координат маніпулятора.

Рисунок 4.16 – Результат експерименту з визначення координат розміщення об’єкту після калібрування

Для дослідження точності визначення координат розміщення об’єкту на предметному столі розроблено шаблон із позиціями для розташування тестового об’єкту. На рисунку 4.17 показано приклад шаблону.

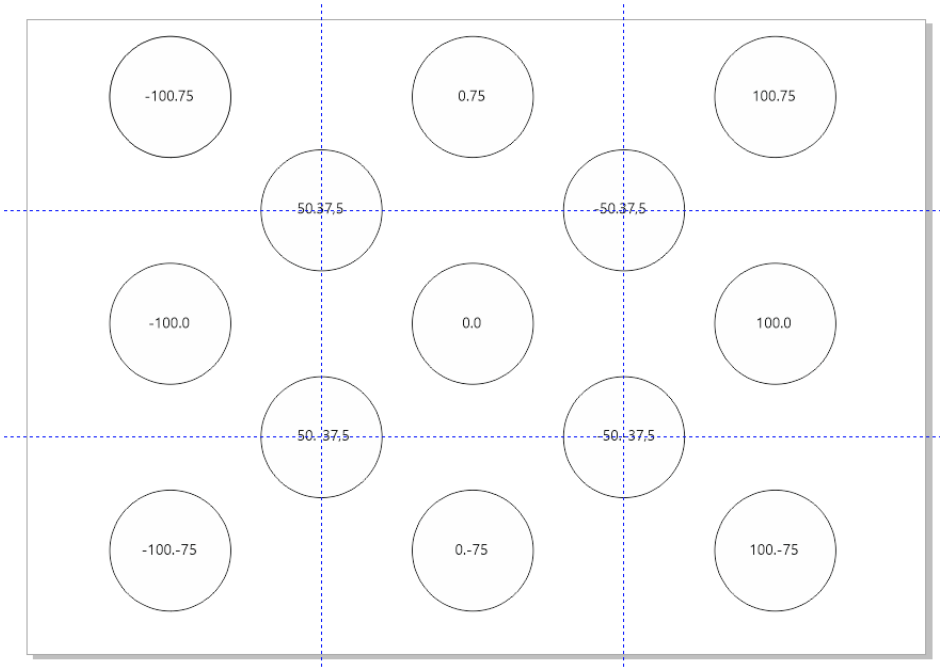


Рисунок 4.17 – Приклад шаблону для розташування тестового об’єкту

Кожна окружність має нанесені координати її центру. В процесі експерименту потрібно буде розмістити об’єкт на означеному місці та запустити підпрограму розпізнавання. На рисунку 4.18 показано приклад тестового шаблону на екрані комп’ютера.

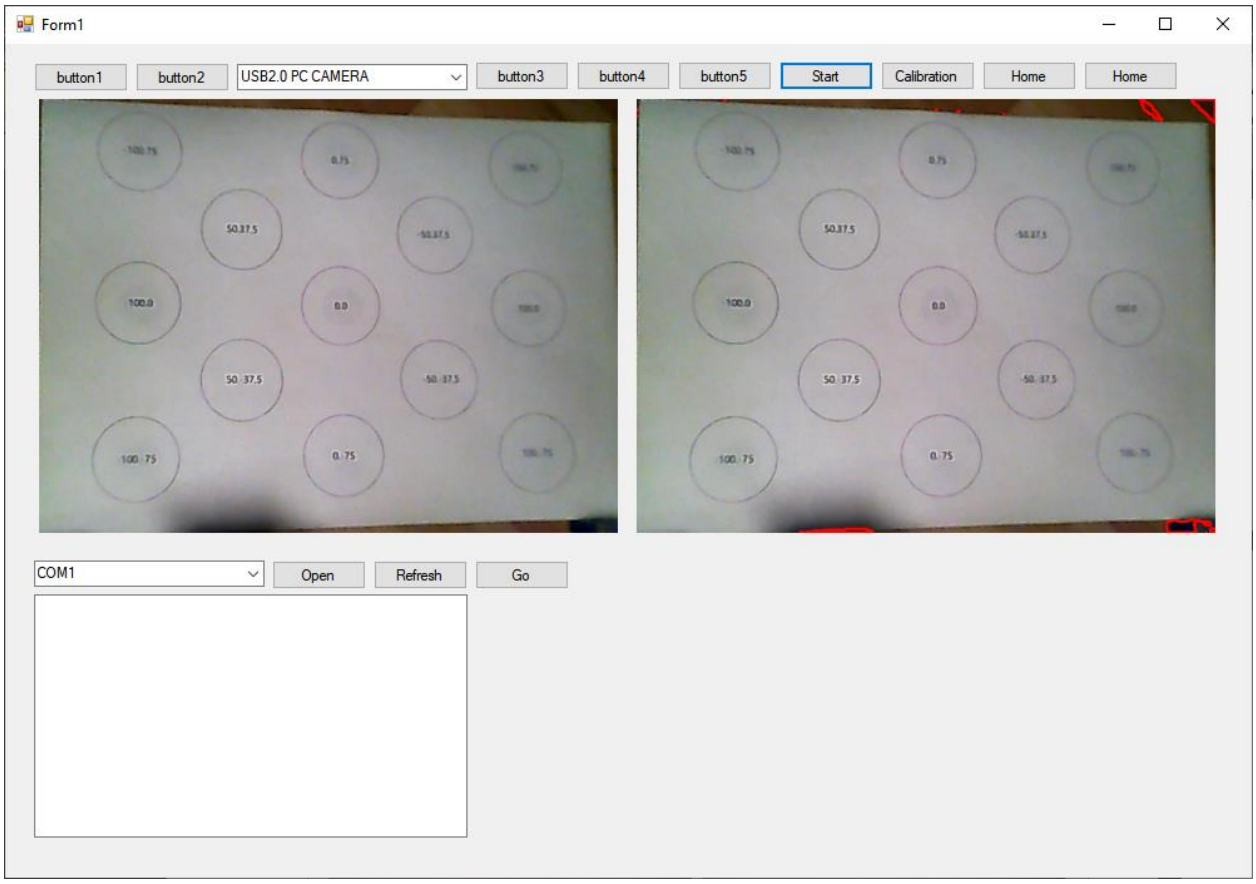


Рисунок 4.18 – Приклад тестового шаблону на екрані комп’ютера

В процесі проведення експерименту послідовно виконані дев’ять основних вимірів позиції розташування об’єкту на тестовому шаблоні. Результати експерименту показані на рисунку 4.19.

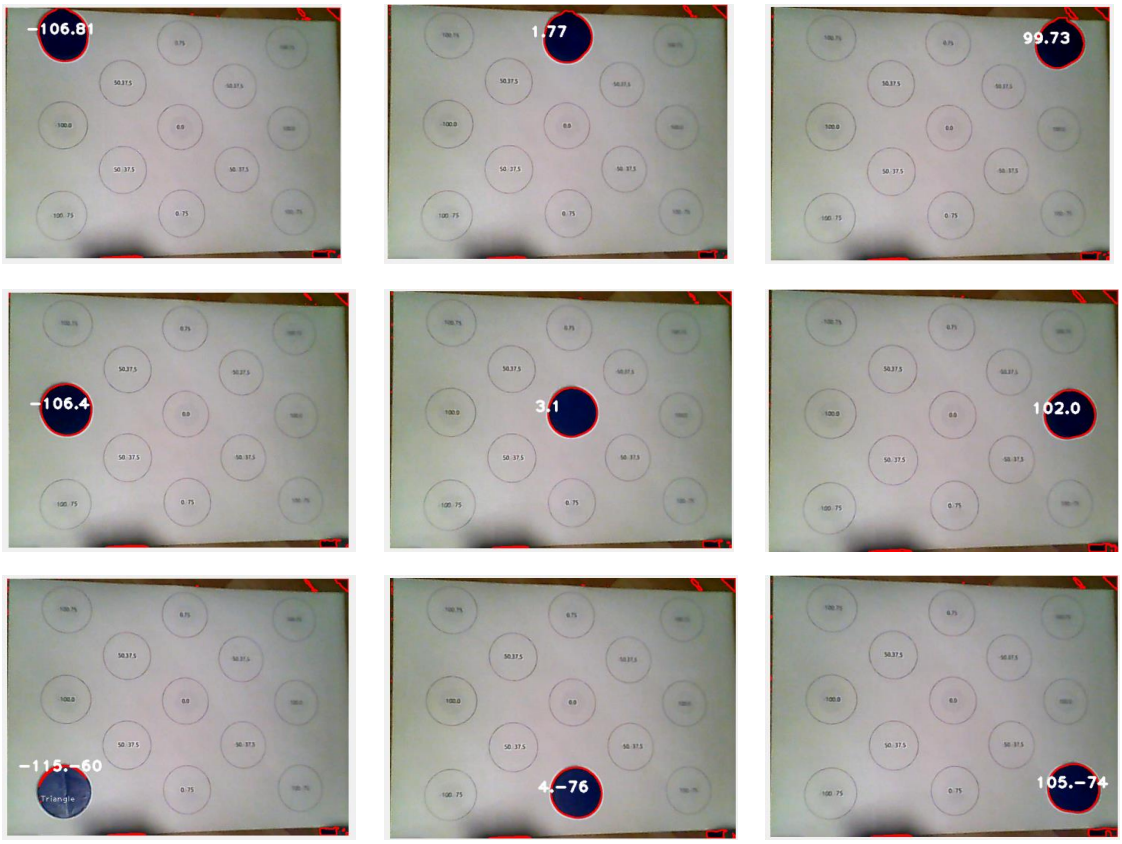


Рисунок 4.19 – Базові координати розташування об’єкту

Після визначення базових координат було проведено ще чотири експерименти для допоміжних координат розташування об’єкту рисунок 4.20.

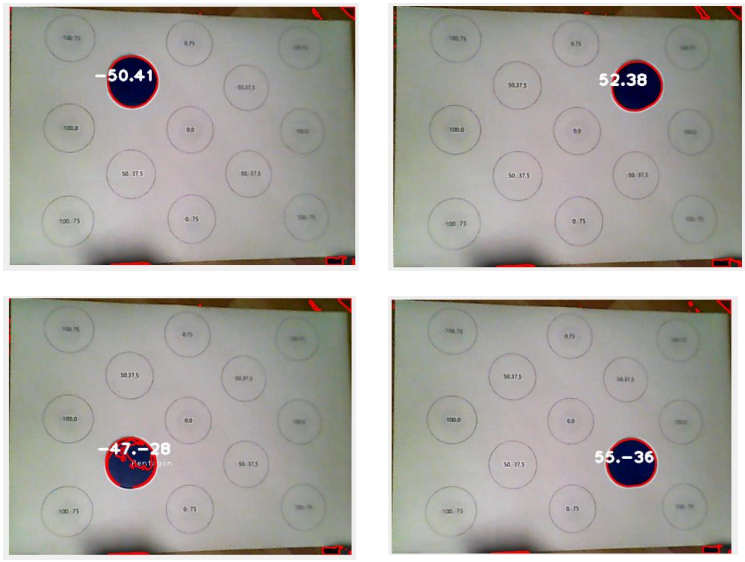


Рисунок 4.20 – Допоміжні координати розташування об’єкту

Результати експериментальних досліджень зведені в таблицю 4.2.

Таблиця 4.2 – Результати експерименту

№ експерименту	очікувані дані	отримані дані	№ експерименту	очікувані дані	отримані дані
1	-100.75	-100.81	8	0.-75	4.-76
2	-100.0	-100.4	9	50.37,5	52.38
3	-100.75	-115.-60	10	50.-37,5	55.-36
4	-50.37,5	-50.41	11	100.75	99.73
5	-50.-37,5	-47.-28	12	100.0	102.0
6	0.75	1.77	13	100.-75	105.-74
7	0.0	3.1			

В таблиці представлені очікувані дані, що відповідають тестовому шаблону та отримані в результаті розпізнавання дані. На рисунку 4.21 графічно показано динаміку зміни значень координати X.

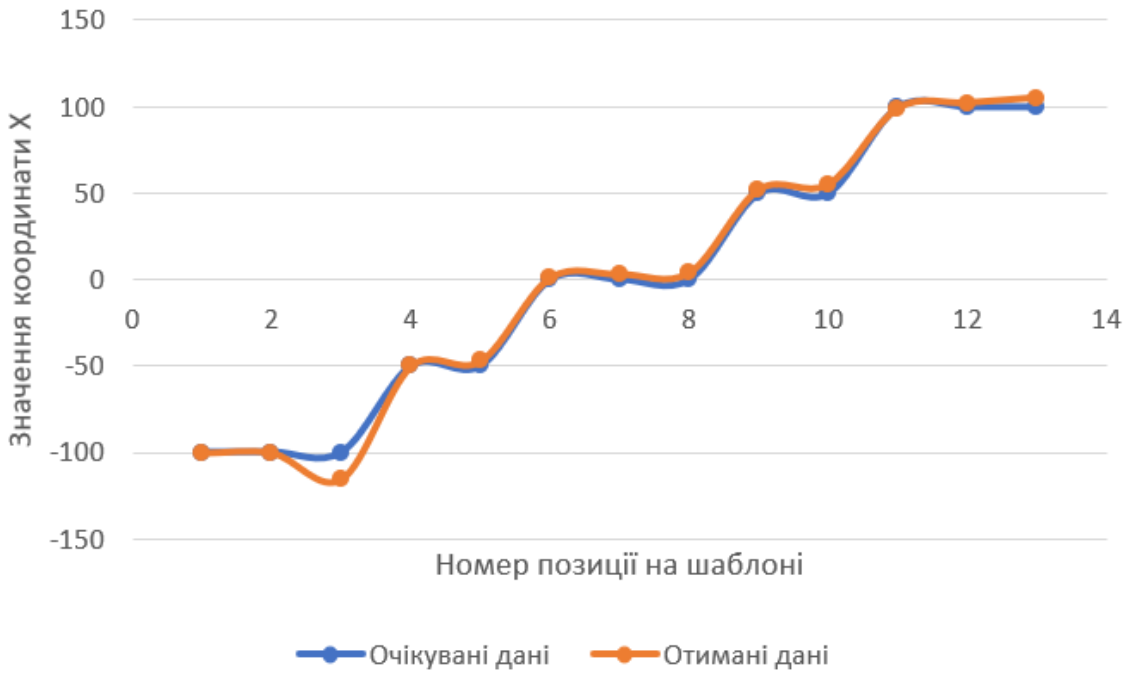


Рисунок 4.21 – Динаміка зміни значень координати X

На рисунку 4.22 показано результати виміру значень координати Y.

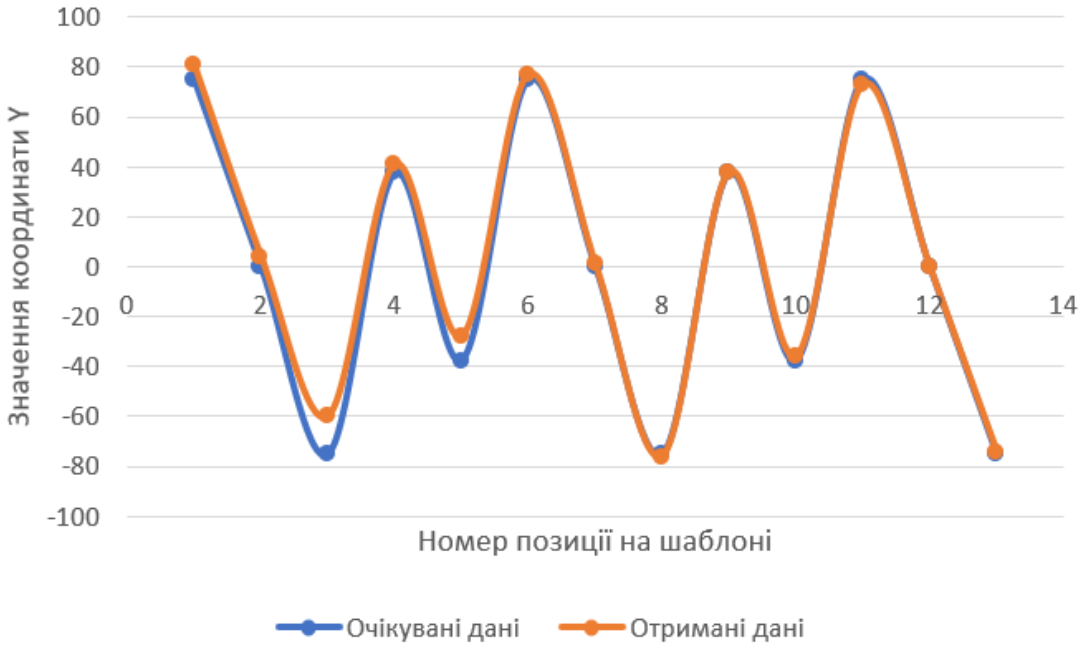


Рисунок 4.22 – Результати виміру значень координати Y

За результатами проведених експериментів розраховано похибку визначення координат розміщення об’єктів з використанням програми MS Excel. Отримані результати показані на рисунку 4.23.

	x	y	x	y	dx	dy
1	-100	75	-100	81	0	6
2	-100	0	-100	4	0	4
3	-100	-75	-115	-60	-15	-15
4	-50	37,5	-50	41	0	3,5
5	-50	-37,5	-47	-28	-3	-9,5
6	0	75	1	77	1	2
7	0	0	3	1	3	1
8	0	-75	4	-76	4	1
9	50	37,5	52	38	2	0,5
10	50	-37,5	55	-36	5	-1,5
11	100	75	99	73	1	-2
12	100	0	102	0	2	0
13	100	-75	105	-74	5	-1

Рисунок 4.23 – Похибка визначення координат розміщення об’єктів

На рисунку 4.24 показано графік зміни похибки визначення координат в залежності від номеру експерименту а значить і від місця проведення вимірювання.

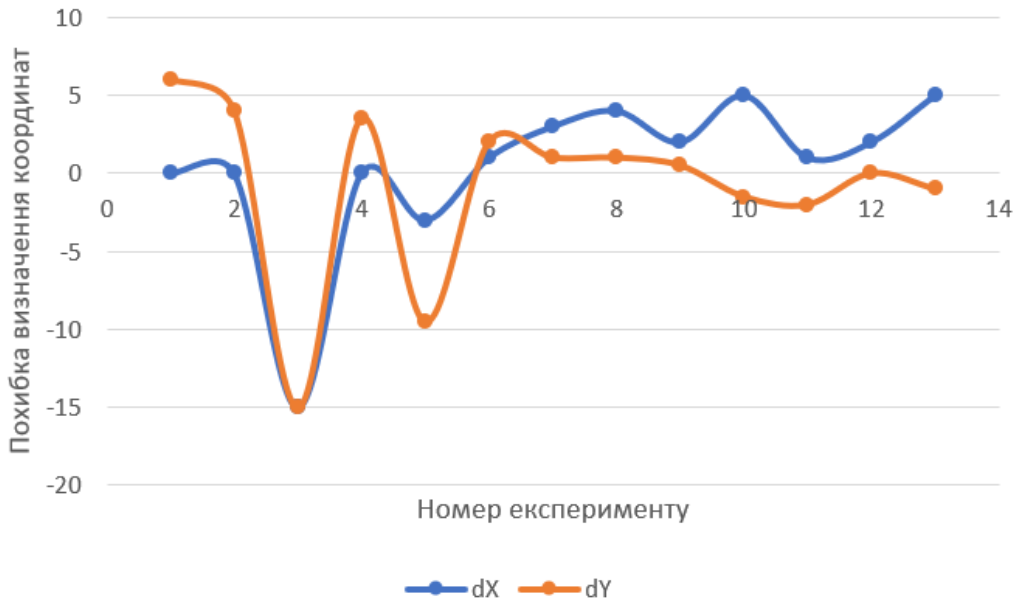


Рисунок 4.24 – Графік зміни похибки визначення координат в залежності від місця проведення вимірювання

4.4 Охорона праці

Роботи в лабораторії відносяться до робіт категорії 1а – легка фізична робота, яка виконується сидячи.

Оптимальні норми мікроклімату для холодного та теплого періоду року згідно і ДСН 3.3.6.042-99:

– температура від 22 °С до 25 °С;

– відносна вологість від 40 % до 60 %;

– швидкість руху повітря не більше 0,1 м/с.

Забезпечуються припливно-витяжна загальнообмінна вентиляція.

В приміщенні використовується сумісне освітлення: природне та штучне. Згідно з ДБН В.2.5-28-2006 категорія зорових робіт, що проводяться у

приміщенні – III В. Штучне освітлення виконано як загальне, за допомогою світильників з люмінесцентними лампами.

Шум в приміщенні відповідає нормативним значенням – 50 дБ згідно з ДСН 3.3.6-037-99.

Так як домінуючим шкідливим фактором є пари свинцю зробимо розрахунок повітрообміну в робочому приміщенні.

В лабораторних умовах найбільш доцільним буде встановити припливно-витяжну загальнообмінну вентиляцію.

У лабораторії об’ємом V=213,5 м3 проводиться паяння та лудіння м'яким припоєм ПОС-40 (до його складу входить р – 40% свинцю). За 1 годину роботи витрачається т = 105 мг припою. Кількість випаровуємого припою q = 0,3 %. Число працюючих n = 2 особи.

У приміщенні лабораторії знаходяться тверді горючі матеріали, тому за вибухопожежної та пожежної небезпеки приміщення слід віднести до категорії В, згідно НАПБ Б.03.002-2007. Приміщення належить до класу П - IIа згідно ПУЕ-2011.

Лабораторія розташована в будівлі II ступеня вогнестійкості за ДБН В 1.1.7-2002.

У приміщенні знаходяться ПЕОМ, які представляють собою пожежну небезпеку, тому що при підвищенні температури окремих вузлів можливо оплавлення ізоляції сполучних проводів, яке веде до замикання, що супроводжується в свою чергу іскрінням.

Причиною пожежі в лабораторії можуть бути несправність електрообладнання, руйнування ізоляції провідників, порушення правил пожежної безпеки.

Пожежна безпека в лабораторії забезпечується відповідно до ГОСТ 12.1.004-91, системою запобігання пожежі, протипожежного захисту та організаційно-технічними заходами.

Згідно ДБН В.2.5.56-2010 в приміщенні встановлено точковий димовий пожежний сповіщувач ДИП-1, який контролює площу до 86 м^2.

Згідно НАПБ Б03.001-2004 в приміщенні розміщені первинні засоби пожежогасіння – вуглекислотний вогнегасник ВВК-1,4 з розрахунку 1 вогнегасник на 3 ПК, але не менше 1 на приміщення.

Організаційні заходи: проводиться інструктаж персоналу з пожежної безпеки; розроблено заходи щодо дій адміністрації на випадок виникнення пожежі; на видному місці розміщений план евакуації при пожежі.

4.5 Висновки за результатами експериментальних досліджень

За результатами експериментальних досліджень можна зробити висновок, що похибка визначення координат зростає у нижній частині шаблону, тобто у ближній зоні роботи маніпулятора.

Проведений аналіз показав, що така поведінка системи розпізнавання пов’язана з нерівномірністю освітлення на робочому місці. Об’єкти у нижній частині предметного столу знаходились під кутом до джерела світла і тому вони відкидали більшу тінь, ніж об’єкти у верхній частині столу. Система розпізнавання визначала тінь, як продовження контуру об’єкту і таким чином його площа зростала, а координати центру змінювались, що і призводило до вказаної похибки.

ВИСНОВКИ

В результаті виконання першого розділу атестаційної роботи проведено аналіз методів застосування машинного зору для промислових роботів. Виконано аналіз об’єкту керування, в якості якого виступає учбовий макет робота маніпулятора. Виконано аналіз кінематичної схеми маніпулятора для подальшої можливості створення програми керування переміщенням його суглобів в залежності від визначеної координати розташування об’єкту.

В результаті виконання другого розділу атестаційної роботи проведено аналіз основних задач, що виконуються системами комп'ютерного зору, розглянуті основні методи обробки зображень. Виконано аналіз методів і алгоритмів обробки зображень системами комп'ютерного зору. Проаналізовані переваги та недоліки використання основних методів для обробки зображень. В якості основного методу обрано контурний аналіз.

В результаті виконання третього розділу атестаційної роботи розроблено структурну схему системи позиціонування суглобів маніпулятора. В якості центрального модуля керування можна використовувати одноплатний комп’ютер Raspberry, або персональний комп’ютер. Розроблено функціональну схему системи позиціонування маніпулятора.

Наведено відомості про метод перетворення координат розміщення об’єктів на предметному столі в координатну систему робота для використання його в розділі експериментальних досліджень.

Виконано вибір засобів програмної реалізації методів комп’ютерного зору. В якості основної бібліотеки обрано OpenCV. Наведено приклад реалізації архітектури бібліотеки OpenCV та розглянуті основні функції, що використовуються для обробки зображень.

За результатами експериментальних досліджень можна зробити висновок, що похибка визначення координат зростає у нижній частині шаблону, тобто у ближній зоні роботи маніпулятора.

Проведений аналіз показав, що така поведінка системи розпізнавання пов’язана з нерівномірністю освітлення на робочому місці. Об’єкти у нижній частині предметного столу знаходились під кутом до джерела світла і тому вони відкидали більшу тінь, ніж об’єкти у верхній частині столу. Система розпізнавання визначала тінь, як продовження контуру об’єкту і таким чином його площа зростала, а координати центру змінювались, що і призводило до вказаної похибки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. ДСТУ 3008–2015. Звіти у сфері науки і техніки. Структура і правила оформлення. Документація. – Введ. 2015-06-22. - К.: Держстандарт України, 2015. - 31 с.

2. Методичні вказівки з «Розробки й оформлення магістерської атестаційної роботи» для студентів другого (магістерського) рівня вищої освіти галузі знань 15 Автоматизація та приладобудування за спеціальністю 151 Автоматизація та комп'ютерно-інтегровані технології освітні програми: «Автоматизоване управління технологічними процесами», «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / Упоряд. І.Ш. Невлюдов, В.В. Косенко, В.В. Євсєєв. – Харків: ХНУРЕ, 2019. – 55 с.

3. Основи наукових досліджень: Навч. посібник / І.Ш. Невлюдов, Ю.М. Олександров, А.О. Андрусевич, О.О. Чала. – Кривий Ріг: Криворізький коледж НАУ, 2019. – 396 с.

4. Borisov O.I., Gromov V.S., Pyrkin A.A., Vedyakov A.A., Petranevsky I.V., Bobtsov A.A., Salikhov V.I., Manipulation Tasks in Robotics Education // IFAC-PapersOnLine. 2016, V. 49, N. 6, P. 22–27.

5. L. Zhang, A. Dehghani, Z. Su, T. King, B. Greenwood and M. Levesley, "Development of a Mechatronic Sorting System for Removing Contaminants From Wool", IEEE/ASME Transactions on Mechatronics, vol. 10, no. 3, pp. 297-304, 2005.

6. Кинематические схемы манипуляторов [Електронний ресурс] URL: — http://stroy-technics.ru/article/kinematicheskie-skhemy-manipulyatorov – Дата звернення: 02.10.2021 р.

7. Md. Hazrat Ali, Aizat K., Yerkhan K., Zhandos T., Anuar O., Vision-based Robot Manipulator for Industrial Applications, Procedia Computer Science, Volume 133, 2018, Pages 205-212

8. OpenCV шаг за шагом. Нахождение контуров и операции с ними [Електронний ресурс] URL: https://robocraft.ru/blog/computervision/640.html – Дата звернення: 14.09.2021 р.

9. A. Djajadi, F. Laoda, R. Rusyadi, T. Prajogo and M. Sinaga, "A model vision of sorting system application using robotic manipulator", Journal.uad.ac.id, 2017. [Online]. Available: http://journal.uad.ac.id/index.php/TELKOMNIKA /article/view/615/424. [Accessed: 21- Sep- 2021].

10. Juang, Jih-Gau, Yi-Ju Tsai, and Yang-Wu Fan. "Visual recognition and its application to robot arm control." Applied Sciences 5, no. 4 (2015).

11. Rege Sanket. 2D geometric shape and color recognition using digital image processing International journal of advanced research in electrical, electronics and instrumentation engineering, 2 (6) (2013), pp. 2479-2487.

12. Kumar Rahul. Object detection and recognition for a pick and place Robot Computer Science and Engineering (APWC on CSE), 2014 Asia-Pacific World Congress on. IEEE (2014).

13. Chhaya Shambhavi Vijay, Sachin Khera, Pradeep Kumar. Basic geometric shape and primary color detection using image processing on Matlab IJRET: International Journal of Research in Engineering and Technology, 4 (2015), pp. 1163-2319.