Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інформаційно-вимірюіальних технологій

Мороз Юрій Васильович

(назва роботи)

УДК 535.317.2
(індекс)

МАГІСТЕРСЬКА РОБОТА

Дослідження оптичних методів і засобів контролю розмірів виробів круглої та еліптичної форми

(назва роботи)

Метрологія та інформаційно-вимірювальна техніка
(назва освітньої програми)

152 Метрологія та інформаційно-вимірювальна техніка
(шифр і назва спеціальності)

Робота містить результати власних досліджень, використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело:

Здобувач освітнього ступеня Ю.В.Мороз
(підпис, ініціали та прізвище здобувача)

Науковий керівник Біліщук В. Б., к.т.н., доцент
(підпис, прізвище, ім'я, по батькові, науковий ступінь, вчене звання керівника)

Допущено до захисту
Завідувач кафедри

О.Є.Середюк
(посада) (підпис) (дата) (ініціали та прізвище)

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу
(повне найменування закладу вищої освіти)

Інститут *інформаційних технологій*

Кафедра *інформаційно-вимірюіальних технологій*

Освітній рівень *перший (бакалаврський)*

Спеціальність *152- Метрологія та інформаційно-вимірювальна техніка*
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри ___МІВТ___

_____О.Є. Середюк

«____» _____2024 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТОВІ

Мороз Юрій Васильович
(Прізвище, ім'я, по батькові)

1. Тема роботи Дослідження оптичних методів і засобів контролю розмірів виробів круглої та еліптичної форми
Керівник роботи *Біліщук Віктор Борисович к.т.н., доц.*
(Прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від 15.12.2023 року №737/7

2. Строк подання студентом роботи 31.01.24
3. Вхідні дані до роботи тестове коло радіусом – 2 см, Aruco мітка 5х5 см, температура – 30°С, освітленість приміщення – 505 лк.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1 Аналіз досліджуваного об'єкту. 2 Аналіз алгоритмів ідентифікації об'єктів методу оптичного контролю. 3 Розроблення системи оптично-го контролю. 4 Розроблення алгоритму ідентифікації. 5 Аналіз метрологічно-го забезпечення .
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) 1 Система оптичного контролю розмірів деталей циліндричної форми. Схема структурна. 2 Система оптичного контролю розмірів деталей циліндричної форми. Алгоритм ідентифікації та вимірювання деталей. Схема структурна. 3 – Система оптичного контролю розмірів деталей циліндричної форми. Алгоритм роботи програмного забезпечення. Схема функціональна. 4. Система оптичного контролю розмірів деталей циліндричної форми. Демонстрація роботи. 5 Система оптичного контролю розмірів деталей циліндричної форми. Метрологічний аналіз.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
	Лютак З. П.		

7. Дата видачі завдання 11.12.23

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Термін виконання етапів роботи	Примітка
1.	*Аналіз досліджуваного об'єкту.*	*11.12.23*	
2.	*Аналіз алгоритмів ідентифікації*	*20.12.23*	
	об'єктів методу оптичного		
	контролю.		
3.	*Розроблення системи оптичного*	*30.12.23*	
	контролю.		
4.	*Розроблення алгоритму ідентифікації.*	*10.01.24*	
5.	*Аналіз метрологічного забезпечення.*	*16.01.24*	
6.	*Оформлення магістерської роботи.*	*20.01.24*	

Студент_____ Мороз Ю.В.
(підпис) (прізвище та ініціали)

Керівник роботи_____ Біліщук В.Б.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Магістерська робота: 88 с., 18 рис., 3 табл., 21 джерело, 5 аркушів креслень.

Об’єкт дослідження поршень двигуна внутрішнього згорання.

Мета роботи – Дослідження оптичних методів і засобів контролю розмірів виробів круглої та еліптичної форми.

Оптичний контроль лінійних розмірів дозволяє автоматично вимірювати об'єкт без контакту, без участі оператора, що усуває можливість суб'єктивних похибок у процесі вимірювання. Досліджено питання оптичних методів та засобів для контролю геометричних розмірів головки поршня двигуна внутрішнього згорання. На основі дослідження розроблений алгоритм ідентифікації та вимірювання геометричних розмірів об’єктів, а також побудована оптична система контролю діаметра деталей круглої та еліптичної форми.

ОПТИКА, АЛГОРИТМ, РАДІУС, ДІАМЕТР, КОНТРОЛЬ, КОЛО, РОЗМИТТЯ, КОНТУР, ЕЛІПС, ВИМІРЮВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, OPENCV, МОДЕЛЬ.

ABSTRACT

Master’s thesis: 88 p., 18 drawings, 3 tables, 21 sources, 5 draughts.

Research object is a piston of the internal combustion engine.

Purpose is researching optical methods and means for dimension control of products with circular and elliptical shapes

Optical control of linear dimensions allows for automatic measurement of an object without contact and without the involvement of an operator, eliminating the possibility of subjective errors in the measurement process. The issues of optical methods and means for controlling the geometric dimensions of the piston head in an internal combustion engine have been studied. Based on the research, an algorithm for identification and measurement of geometric dimensions of objects has been developed, and an optical system has been constructed.

OPTICS, ALGORITHM, RADIUS, DIAMETER, CONTROL, CIRCLE, BLUR, CONTOUR, ELLIPSE, MEASUREMENT, SOFTWARE, OPENCV, MODEL

ЗМІСТ

ВСТУП

Актуальність теми. Двигуни внутрішнього згоряння (ДВЗ) відіграють визначальну роль у технологічному розвитку та підтримці сучасного суспільства. Найбільш очевидна сфера їхнього впливу - транспорт. Автомобілі, які в основному працюють на ДВЗ, є ключовим засобом забезпечення мобільності та сприяють створенню розвинутих транспортних систем. Літаки, кораблі та інші види транспорту також використовують ДВЗ, дозволяючи здійснювати швидкі та великі переміщення як на локальному, так і на міжнародному рівні. Поршень є ключовим елементом для роботи ДВЗ, він повинен ретельно відповідати метрологічним характеристикам, адже незначне відхилення може призвести до проблем з поршневою системою в ДВЗ які можуть викликати різноманітні несправності, що мають великий вплив на його ефективність та безпеку. Слід дотримуватись високого рівня контролю об’єкта, в тому числі його геометричних параметрів. Провівши аналіз поршня ДВЗ у роботі показав, що надважливою його частиною є головка поршня, адже вона приймає на себе надзвичайний тиск та температури, відбувається термічне розширення, тому, дотримання її геометричних розмірів є вкрай важливим.

Оптичний метод контролю головок поршнів забезпечує високу точність та швидкість вимірювань. Метод використовує світлові промені для детального аналізу геометричних параметрів та виявлення можливих дефектів чи нерівності на поверхні головок поршнів. Важливою перевагою оптичного контролю є його неруйнівність, що дозволяє виконувати вимірювання без пошкодження та впливу на фізичні властивості матеріалів. Це особливо корисно для забезпечення високої якості та довговічності головок поршнів. Додатково, оптичний контроль може бути легко інтегрований в автоматизовані виробничі процеси, де він забезпечує велику швидкість вимірювань та використання в реальному часі. Це дозволяє операторам вчасно виявляти можливі дефекти та приймати відповідні корекційні заходи, щоб забезпечити високу якість продукції.

Проведений аналіз алгоритмів ідентифікації об’єктів дав змогу оцінити їх можливості, структуру та логічні послідовності роботи, переваги та недоліки, а також отримати знання які були використанні для розробки власного алгоритму.

Розроблений у рамках магістерської роботи алгоритм слугує для ідентифікації об’єктів круглої та овальної форми та вимірювання їх радіусів. Алгоритм можна розділі на декілька ключових кроків, а саме: Попередня обробка зображення, пошук країв, Пошук контурів, обчислення розміру контуру Фільтрація для виявлення контурів та калібрування. У алгоритмі використовується метод Кенні для пошуку країв об’єктів, а також визначення найменшого охопленого кола за допомогою алгоритму Велцля.

Розроблено програмне забезпечення мовою програмування Python з використання бібліотеки комп’ютерного зору «OpenCV», для контролю радіусу деталей, базоване на власному алгоритмі. Це ПЗ володіє графічним інтерфейсом для зручного використання, взаємодіє з базою даних та може ефективно визначати придатність або необхідність бракування об'єкта вимірювання.

Метрологічний аналіз показав перспективність розробленого алгоритму та системи оптичного контролю геометричних розмірів.

Мета і завдання дослідження. Аналіз методів на засобів оптичного контролю геометричних розмірів, розроблення власного алгоритму ідентифікації та вимірювання, а також розроблення власної системи оптичного контролю об’єктів круглої та еліптичної форми.

Відповідно до цієї мети необхідно вирішити такі задачі:

- Проаналізувати поршень ДВЗ та методи які використовуються для його контролю, обґрунтувати необхідність та можливість розробки автоматизованої системи оптичного контролю головки поршня ДВЗ.
- Обґрунтувати вибір оптичного методу контролю головки поршня ДВЗ.
- Проаналізувати існуючи алгоритми ідентифікації об’єктів які використовуються оптичними системами.

– Провести метрологічний аналіз розробленої системи.

Об’єктом дослідження є головка поршня двигуна внутрішнього згорання.

Предметом дослідження є методи та засоби оптичного контролю геометричних параметрів об’єктів.

Методи дослідження. В роботі проведено аналіз алгоритмів оптичної ідентифікації та їх унікальних рис, а також використані аналітичні методи при математичному описі системи оптичного контролю головки поршнів ДВЗ.

Наукова новизна отриманих результатів – вдосконалено методику оптичного контролю розмірів циліндричних об'єктів, яка передбачає попередню обробку зображення; пошук країв за допомогою алгоритму Кенні (Canny); Пошук контурів; Обчислення розміру контуру за допомогою алгоритму Велцля; Фільтрація для виявлення контурів; Калібрування за допомогою спеціальних Aruco маркерів.

Практичне значення отриманих результатів:

- Проведено аналіз методів та засобів контролю геометричних параметрів головки поршня ДВЗ та встановлено їх переваги та недоліки.
- Розроблено алгоритм роботи системи оптичного контролю розмірів деталей циліндричної форми.
- Розроблено програмне забезпечення системи оптичного контролю деталей циліндричної форми.
- Розроблено математичний опис алгоритму використаного у програмному забезпеченні для оптичного контролю.
- Розроблено метрологічний аналіз системи оптичного контролю розмірів деталей циліндричної форми.

Апробація результатів магістерської роботи. Згідно виконаної роботи було подано тези і прийнято участь у Всеукраїнській Інтернет-конференції молодих учених і студентів, ІТОТП-2023 «Інформаційні технології в освіті, техніці та промисловості». (м. Івано-Франківськ, 2023 р.) [1].

1 АНАЛІЗ ДОСЛІДЖУВАНОГО ОБ'ЄКТУ

1.1 Загальний опис

Поршень у двигуні внутрішнього згоряння (ДВЗ) виконує ключову роль у приведенні в рух цього складного механізму. Це циліндричний елемент, який рухається вздовж циліндра двигуна, і його вертикальний рух утворює циклічний процес роботи. Важливість поршня полягає в його функціях і впливі на продуктивність та ефективність ДВЗ.

В свою чергу, ДВЗ є важливим компонентом сучасного життя і суспільства. Вони використовуються для приведення в рух транспортних засобів у всіх галузях, включаючи автомобілі, літаки та судна. Ці двигуни також є вирішальним елементом в промисловості, сільському господарстві та енергетиці, забезпечуючи енергію для великої кількості машин і процесів. Використання ДВЗ в домогосподарствах і підприємствах сприяє рухливості та економічному розвитку.

Поршень відповідає за рух вгору та вниз у циліндрі під впливом вибухових газів, що виникають після запалювання пального. Цей рух поршня перетворюється на обертання колінчастого валу, що в свою чергу приводить в рух інші елементи механізму, такі як шатуни та клапани. Такий механізм дозволяє двигуну генерувати потужність та виконувати корисну роботу.

Поршень також відіграє роль у забезпеченні герметичності циліндра. Він утворює тісну паспортну пару з циліндром, запобігаючи витоку газів та забезпечуючи ефективність роботи двигуна. Крім того, поршень співпрацює з поршневими кільцями, які також важливі для контролю тиску та зменшення витрат мастила [2].

Узагальнюючи, поршень в ДВЗ виконує функції, що визначають його значущість: генерує рух, перетворюючи енергію вибухових газів, забезпечує герметичність циліндра та сприяє ефективності роботи двигуна.

1.2 Будова та принцип роботи поршня ДВЗ

Конструктивно, поршень представляє собою циліндричну деталь, яка має кріплення до шатуна, а сам шатун, у свою чергу, закріплений на кривошипі колінчастого валу. Така організація конструкції дозволяє ефективно перетворювати енергію, що виникає у циліндрі під час горіння пального, у момент, необхідний для приведення транспортного засобу в рух.

Поршень у двигуні внутрішнього згоряння (ДВЗ) має складну структуру, яка включає різні компоненти для виконання своїх функцій. Основні елементи будови поршня:

Днище (Нижня частина): Днище поршня є нижньою частиною і контактує з гарячими газами, які утворюються під час згоряння пального. Воно може мати спеціальні форми для оптимізації роботи та ефективного видалення тепла.

Ущільнюючий пояс: Це кільце або область, що оточує поршень, і взаємодіє з циліндровими стінками для утримання газів у циліндрі під час стискання суміші пального та повітря. Ущільнюючий пояс допомагає уникнути витоків газів.

Бобишки (Заготовка або бокові стінки): Це бічні стінки поршня, які з'єднують днище з верхньою частиною. Бобишки можуть мати спеціальні виступи або отвори для зменшення ваги поршня.

Спідничка: Це верхня частина поршня, яка з'єднується з шатуном. Спідничка може мати форму, що сприяє рівномірному розподілу тиску і забезпечує стабільний рух поршня в циліндрі.

Сталева терморегулююча вставка: Ця вставка зазвичай вбудована у верхню частину поршня і призначена для контролю температури. Вона допомагає забезпечити оптимальні теплові характеристики та захищає поршень від перегріву [2].

На рисунку 1.1 представлено наочний вигляд структури стандартного поршня двигуна внутрішнього згоряння.

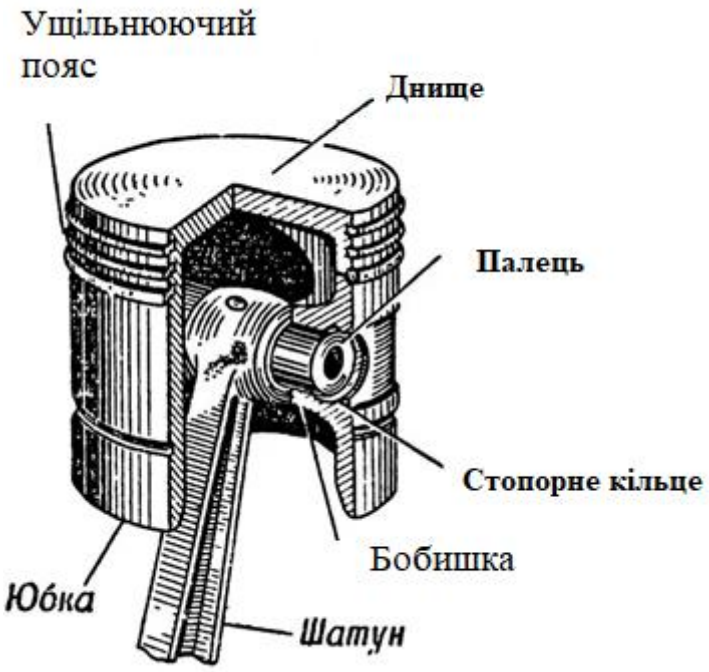

Рисунок 1.1 – Структура будови поршня ДВЗ

1.3 Наслідки несправності поршня ДВЗ

Проблеми з поршневою системою в двигуні внутрішнього згоряння (ДВЗ) можуть викликати різноманітні несправності, що мають важливий вплив на його ефективність та безпеку. Однією з основних проблем є знос або ушкодження поршнів. Знос може бути викликаний поганим мастильним обслуговуванням, низькою якістю пального або неправильними умовами експлуатації. Пошкоджені поршні можуть призвести до втрати герметичності у циліндрі, що в свою чергу може спричинити витік масла і зменшення тиску стискання. Ще однією проблемою може бути зламаний поршневий кільцевий механізм. Кільця поршня відповідають за утримання тиску газів у циліндрі, і якщо вони пошкоджені, може виникнути витік газів. Це призводить до втрати ефективності

двигуна та може викликати проблеми зі стисканням пального та повітря. Несправності у поршневій системі також можуть включати в себе проблеми з поршневою шатунною групою.

Коли поршень не відповідає своїм геометричним характеристикам і має невелике збільшення або зменшення розмірів, це може викликати ряд проблем та наслідків для роботи двигуна внутрішнього згоряння (ДВЗ). Нижче розглянуті можливі проблеми та їхні наслідки:

Зменшення розмірів поршня, навпаки, може призвести до витоку газів та втрати герметичності у циліндрі, що вплине на тиск та може знизити ефективність роботи двигуна. Крім того, зменшення розмірів поршня може зменшити об'єм циліндра, що призводить до втрати об'єму суміші пального та повітря під час роботи двигуна.

Збільшення розмірів поршня може призвести до втрати герметичності у циліндрі, оскільки поршень може неправильно взаємодіяти з циліндровими стінками. Це може стати причиною витоку газів і вплинути на ефективність двигуна. З іншого боку, збільшення об'єму циліндра може також збільшити тиск під час стискання суміші пального та повітря, що може вплинути на ефективність та роботу двигуна Крім цього, ДВЗ може заклинити та викликати серйозні поломки і суттєво знизити продуктивність двигуна. Це може трапитися з різних причин. Низький рівень мастила може призвести до недостатнього змащення, що створює умови для заклинення поршня через надмірне тертя. Пошкодження поршневих кільцевих канавок також може спричинити заклинення. Неправильний розподіл тепла або висока температура можуть призвести до розширення поршня і його заклинення у циліндрі.

Наслідком заклинення поршня може бути поломка поршневого пальця, пошкодження циліндрової обшивки чи самого поршня. Це може призвести до великих витрат на ремонт та вимагати розбирання двигуна для виправлення проблем. Періодична перевірка та обслуговування двигуна, а також

використання якісних мастил та палива, можуть допомогти уникнути проблем і зберегти працездатність ДВЗ.

Вихід з ладу двигуна внутрішнього згоряння (ДВЗ) через проблему поршнів може мати різноманітні негативні наслідки для водія та пасажирів, включаючи небезпеку на дорозі. Порушення роботи ДВЗ може вплинути на безпеку на дорозі, особливо якщо аварійна ситуація виникає під час руху. Втрата потужності та контролю над автомобілем може призвести до небезпечних ситуацій, таких як втрата швидкості або непередбачені рухи на дорозі.

Проблеми з поршнями також можуть створити експлозійну загрозу. Несправний поршень може викликати неконтрольоване вибухове спалаху, особливо якщо поршень вийде з ладу або відбудеться витік пального. Це може створити серйозну загрозу для безпеки пасажирів та інших користувачів дороги.

Неправильно функціонуючий поршень може призвести до викидів токсичних речовин у вихлопних газах через неповне згорання пального. Це може мати негативний вплив на якість повітря та здоров'я людей, зокрема в міських районах.

У разі проблем з поршнями, власник транспортного засобу може зіткнутися з великими витратами на ремонт. Поправка проблем може бути дорогою та вимагати значних витрат на заміну або відновлення двигуна, що може вплинути на економічні можливості власника.

Нарешті, неправильність у роботі поршнів може призвести до повної втрати мобільності транспортного засобу, що суттєво ускладнить життя власника та його можливість використовувати автомобіль для потреб щоденного життя. Усі ці аспекти підкреслюють важливість своєчасного технічного обслуговування та виявлення проблем для забезпечення безпеки та надійності автотранспорту.

Таким чином, можна визначити, що контроль розмірів головки поршня у двигуні внутрішнього згоряння (ДВЗ) є критично важливим для забезпечення оптимальної ефективності та довговічності двигуна. Розмір головки поршня має прямий вплив на динаміку горіння пального, тиск у циліндрі, а також на загальні параметри роботи двигуна. Враховуючи ці аспекти, контроль розмірів головки поршня стає важливим елементом технічного обслуговування, спрямованого на забезпечення оптимальної роботи двигуна, зменшення негативного впливу на навколишнє середовище та забезпечення безпеки експлуатації.

Агротехніка є невід'ємною частиною сучасного аграрного сектору України, який є одним із визначальних галузей економіки країни. Застосування передових технологій у виробництві сільськогосподарської продукції має ключове значення для підвищення ефективності, забезпечення стабільності врожаю та зменшення витрат. В цьому контексті, важливою складовою агротехніки є належний стан сільськогосподарської техніки, включаючи справність двигунів внутрішнього згоряння (ДВЗ). Справність поршнів у ДВЗ грає ключову роль у забезпеченні оптимальної роботи сільськогосподарської техніки. Поршні відповідають за перетворення енергії внутрішнього згоряння на механічну силу, яка використовується для приводу рухомих частин техніки. Їхній справний стан визначає ефективність роботи двигуна, споживання пального та загальну надійність обладнання.

В умовах сучасної агросфери, де технологічний прогрес відіграє важливу роль у підвищенні врожайності та економії ресурсів, важливо вдосконалювати агротехніку для оптимізації сільськогосподарських процесів. Справність поршнів в ДВЗ забезпечує ефективне використання пального, надійність рухомих механізмів та зниження витрат на обслуговування.

Враховуючи постійні зростання вимог до продуктивності та стійкості сільськогосподарської техніки, збереження справності та якості поршнів у ДВЗ стає важливим елементом агротехніки. Це не лише сприяє оптимальному

використанню ресурсів, але й впливає на підвищення виробничої потужності та забезпечення стабільності аграрного виробництва в Україні.

Двигун Perkins 6.354 TC належить до ряду двигунів Perkins 6.354, які широко використовуються у різних галузях, таких як сільське господарство, будівництво, промисловість та морський транспорт. Версія "TC" вказує на турбодизельну модифікацію цього двигуна. У такому двигуні використовуються поршневі комплекти 98.48 мм. В назві комплекту прямо вказується діаметр зовнішньої частини головки поршня, тобто, двигун Perkins 6.354 TC використовує поршні діаметром 98.48 мм. На рисунку 1.2 наведена схема поршня типурозміру 98.48 мм від виробника B&B, а нижче наведені фактичні розміри поршня [3].

Отвір (мм) 98.48 Алфін YES A (мм) 69.75 B+ / B- (мм) 0 - 18.67 C (мм) D (мм) 107.85 E (мм) 38.10 F (мм) B' (мм) 103.19 C' (мм) 227.28 D' (мм) 106.45 E' (мм) 3.86 F' (мм) 0.89 A' (мм) 98.48 мм.

Алфін (Alfin) - це технологічне покриття або вставка, яке використовується в конструкції поршнів для покращення їхньої міцності та оптимізації теплопередачі.

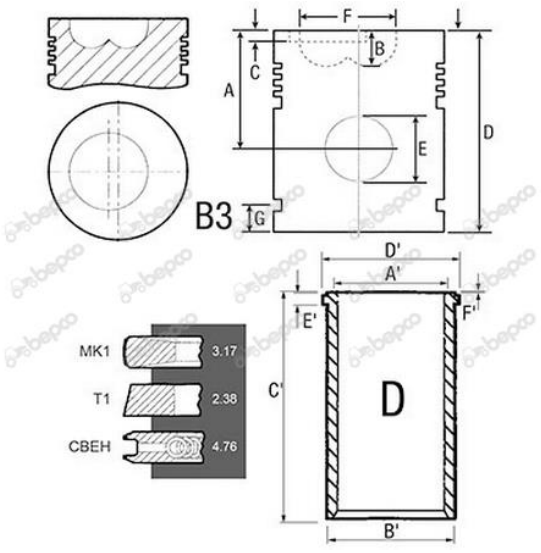

Рисунок 1.2 Схема поршня 98.48 мм від виробника B&B

Застосування технології Alfin спрямоване на поліпшення тривалості служби двигуна та забезпечення його оптимальної ефективності під час експлуатації.

Для різних типів поршнів, допустиме відхилення відрізняється, в більшості V6 це значення становить від 0.015 мм до 0.020 мм. з врахуванням теплового розширення яке виникає під час експлуатації ДВЗ.

Контроль розмірів поршня на масовому виробництві є критичним етапом у виготовленні ДВЗ та інших промислових застосувань, де використовуються поршні. На великих лініях масового виробництва важливо забезпечити стабільність та однорідність розмірів поршнів, щоб уникнути відхилень і забезпечити їхню сумісність з іншими компонентами двигуна. Сучасні технології вимірювання та автоматизовані системи контролю дозволяють досягати великої точності та швидкості при вимірюванні розмірів поршнів.

Забезпечення високої точності та однорідності розмірів поршня важливе для досягнення оптимальної ефективності двигуна та забезпечення його тривалого та надійного функціонування.

1.4 Методи контролю

Пневматичні методи - застосовують струмінь стиснутого повітря, що проходить через сопло і потрапляє на встановлену перед цим отвором деталь. Принцип дії приладу базується на залежності витрати повітря від віддалі між соплом і деталлю. Шкала приладу проградуйована в одиницях довжини при відомій віддалі між соплом і поверхнею, на яку встановлено деталь. Настроюють такі прилади за розміром зразкової деталі або кінцевими мірами довжини. В більшості своєму, пневматичні методи використовуються у машино та

приладобудуванні. Ці прилади використовуються для вимірювань внутрішніх 14 зовнішніх і розмірів, відхилень форми поверхонь (в тому числі внутрішніх), конусів і тому подібних. Пневматичні прилади мають високу точність і швидкодію. Ряд вимірювальних завдань, наприклад точні вимірювання в отворах малого діаметра, вирішується тільки приладами пневматичного типу. Однак прилади цього виду найчастіше вимагають індивідуального градуювання шкали з використанням еталонів [4].

Механічний метод контролю об’єктів застосовує механічні прилади-лінійки, штангенциркулі, пружинні прилади, мікрометричні і тому подібні. Як правило, механічні прилади та інструменти відрізняються, високою надійністю вимірювань, простотою, однак мають порівняно невисоку точність і продуктивність контролю. При проведенні вимірювання необхідно дотримуватися компоративного принципу, згідно з яким необхідно, щоб на одній прямій лінії розташовувалися вісь шкали приладу і контрольований розмір перевіряється деталі, тобто, лінія вимірювання повинна бути продовженням лінії шкали. Якщо цей принцип не дотримується, то перекіс і непаралельність напрямних вимірювального приладу викликають значні похибки вимірювання. Також, використовуючи механічний метод контролю об’єктів, як правило, повинен спеціаліст, який буде проводить цей контроль, присутність людини у вимірюванні може викликати суб’єктивну похибку, і її величина залежить насамперед від самого оператора [5].

Оптичний метод контролю об'єктів стає все більш популярним з прогресом мікроелектроніки. Розвиток як лінз, так і мікросхем та програмного забезпечення дозволяє сьогодні здійснювати вимірювання за допомогою різноманітних оптичних систем. Оптичний контроль розмірів головки поршня є важливим етапом в виробництві та технічному обслуговуванні двигунів внутрішнього згоряння. Цей процес забезпечує високу точність та ефективність у визначенні відхилень розмірів поршневої головки від заданих нормативів. Існує 2 основних методи оптичного контролю, 2D та 3D.

2D оптичний метод здійснює вимірювання на плоскості, інтерпретуючи об'єкт як двовимірне зображення. Використання камер та спеціальних об'єктивів дозволяє збирати велику кількість даних з області дослідження. Програмне забезпечення обробки зображень визначає розміри об'єктів на основі аналізу їхніх двовимірних проекцій. Цей метод ефективний для вимірювань довжин, ширин, радіусів та інших параметрів, які можуть бути оцінені на плоскості.

3D оптичний метод використовує технології для отримання тривимірного образу об'єкта. Це може бути здійснене за допомогою спеціальних камер, лазерних систем або структурованого світла. Зафіксовані дані перетворюються в точковий хмару, що представляє тривимірну модель об'єкта. Програмне забезпечення аналізує цю модель для вимірювань об'ємних параметрів, таких як висота, глибина, кутові величини тощо. 3D оптичний метод особливо ефективний для складних форм та деталей, які не можуть бути повністю описані двовимірними проекціями.

Оптичний метод вимірювань використовує різноманітні системи, такі як комп'ютерний зір, щоб отримати детальну інформацію про геометричні параметри головки поршня. Комп'ютерний зір забезпечує високу швидкість обробки даних та точність вимірювань, дозволяючи ефективно виявляти аномалії та відхилення від стандартів.

Системи оптичного контролю можуть включати в себе використання камер та спеціалізованого програмного забезпечення для аналізу отриманих зображень. Вони ретельно перевіряють розміри головки поршня, форму камери горіння та інші параметри, необхідні для оптимального функціонування двигуна.

Оптичний контроль дозволяє виявляти мікродефекти та недоліки, які можуть бути невидимі за допомогою інших методів вимірювань. Він забезпечує виробникам та технічним спеціалістам високий рівень якості та надійності у виробництві та обслуговуванні двигунів [6].

Застосування оптичного контролю розмірів головки поршня є важливим кроком у підтриманні високих стандартів якості в автомобільній промисловості та гарантує ефективну та безпечну роботу транспортних засобів.

Вибір оптичного методу контролю поршнів є обґрунтованим через декілька переваг, які цей підхід пропонує. По-перше, оптичні методи забезпечують високу точність вимірювань, що особливо важливо у масовому виробництві для дотримання заданих специфікацій. Крім того, вони є швидкими, що сприяє підтримці високого темпу виробництва. Оптичні системи можуть бути як контактними, так і безконтактними, що дозволяє їх використовувати в різних умовах виробництва. Важливою перевагою є також можливість візуалізації та аналізу поршнів. З іншого боку, недоліками оптичного контролю може бути висока вартість обладнання та його впровадження, що може стати значним фактором обмеження. Оптичні системи можуть бути чутливими до зовнішніх впливів, таких як пил, волога та інші агенти, що може вплинути на їхню ефективність в умовах виробництва. Також слід враховувати, що деякі системи можуть вимагати спеціального освітлення та бути обмеженими в здатності вимірювати певні матеріали.

Загалом, обираючи оптичний метод контролю поршнів, важливо ретельно зважити на його переваги та недоліки, враховуючи конкретні вимоги та умови виробництва.

Проаналізувавши об’єкт вимірювання, оцінивши його важливість як складову ДВЗ та небезпеки які можуть виникнути якщо не дотримуватись потрібних геометричних розмірів головки поршня, було вибрано оптичний метод контролю розміру головки поршня ДВЗ для подальшої розробки системи контролю геометричних розмірів головки поршня.

2 АЛГОРИТМИ ІДЕНТИФІКАЦІЇ ОБ’ЄКТІВ МЕТОДУ ОПТИЧНОГО КОНТРОЛЮ

2.1 Алгоритми ідентифікації об'єктів на зображеннях

В сучасному світі комп'ютерного зору та обробки зображень алгоритми ідентифікації об'єктів на зображенні грають важливу роль у розвитку автоматизованих систем та штучного інтелекту. Ці алгоритми дозволяють комп'ютерам розпізнавати, класифікувати та визначати об'єкти на візуальних зображеннях, що відкриває безліч можливостей у різних галузях, від медицини та виробництва до автономних транспортних засобів та розпізнавання обличчя.

Ці алгоритми базуються на математичних та статистичних методах, а також на використанні нейронних мереж та глибокого навчання. Вони дозволяють автоматично виявляти різні об'єкти, включаючи форми, контури та текстури, і надають можливість розпізнавання об'єктів навіть в умовах змінної освітленості та шуму на зображенні.

У цьому контексті важливо вивчати та розробляти нові методи ідентифікації об'єктів, які б не лише забезпечували високу точність, але й були ефективними у використанні та можливість застосування в реальних умовах. Алгоритми ідентифікації об'єктів на зображенні стають ключовим інструментом у вдосконаленні роботи систем відеоспостереження, медичних діагностичних засобів, роботі з автономними транспортними засобами та багатьох інших сферах, сприяючи прискоренню та автоматизації багатьох процесів.

2.2 Каскадний класифікатор Хаара

Каскадні класифікатори Хаара — це алгоритм машинного навчання для виявлення об'єктів на зображеннях, який був розроблений Павлом Віолою та Майклом Джонсом. Він отримав назву від типу особливостей зображення, які використовуються в алгоритмі, та є одним із піонерських методів у комп'ютерному зорі завдяки своїй швидкодії та ефективності. Хаарові особливості базуються на концепції хвильових перетворень і є простими прямокутними фільтрами, які розраховують різницю у сумі інтенсивностей пікселів в різних областях. Кожна особливість f є зваженою сумою пікселів у білій (w) та чорній (b) областях прямокутника:

$$f = \sum w \cdot I(w) - \sum b \cdot I(b), \tag{2.1}$$

де I - позначає інтенсивність пікселів;

Класифікатор використовує алгоритм AdaBoost, який використовується для підсилення слабких класифікаторів, перетворюючи їх у сильний класифікатор. У контексті каскадного класифікатора Хаара, кожна особливість може розглядатися як слабкий класифікатор hj, який приймає рішення на основі значення однієї особливості:

$$\begin{cases} 1 & p_j \cdot f_j(x) < p_j \cdot \theta_j \\ 0 & \text{інакше} \end{cases}, \tag{2.2}$$

де x - вхідне зображення, $fj(x)$ - значення j-ї особливості для x;

θj - порогове значення, pj - напрямок нерівності, який визначається під час тренування.

Кожен слабкий класифікатор вибирається та підсилюється шляхом мінімізації вагомої суми помилок на тренувальному наборі даних.

В каскадній структурі Сильний класифікатор $H(x)$ складається з послідовності T стадій, кожна з яких містить декілька слабких класифікаторів:

$$\begin{cases} 1 \; if \; \sum_{j=1}^{T} \alpha_j \cdot h_j(x) \geq \frac{1}{2} \sum_{j=1}^{T} \alpha_j , \\ 0 \qquad\qquad\qquad\qquad \text{Інакше} \end{cases} \tag{2.3}$$

де αj - вага, надана j-му класифікатору. Об'єкт вважається виявленим, якщо він проходить усі стадії каскаду.

Під час процесу виявлення об'єктів на зображенні, каскад застосовується до кожного можливого місцеположення x і масштабу s у зображенні. Для кожного такого випадку розраховується відповідь сильного класифікатора $H(x)$, і якщо вона дорівнює 1, то регіон класифікується як містящий об'єкт. Цей метод має деякі обмеження, зокрема сильну залежність від якості та різноманітності тренувальних даних і тенденцію до помилок при великих змінах у освітленні, масштабі чи обертанні об'єктів. Крім того, каскадні класифікатори Хаара чутливі до шуму на зображенні [7].

2.3 Гістограма орієнтованих градієнтів

Гістограма орієнтованих градієнтів (ГОГ) є одним із популярних методів для опису особливостей об'єктів для задач розпізнавання та класифікації зображень. Основна ідея методу полягає в тому, що зовнішній вигляд та форма об'єкта в зображенні можуть бути описані розподілом градієнтів або країв. моделі HOG можна розділити на кілька етапів.

Перш за все, зображення нормалізується для зменшення впливу освітлення та контрасту. Це може бути досягнуто за допомогою корекції гамми або корекції кольору.

Після чого, відбувається обчислення градієнтів, де для кожного пікселя зображення обчислюються градієнти по X та Y:

$$Gx = I(x+1, y) - I(x-1, y), \tag{2.4}$$

$$Gy = I(x, y+1) - I(x, y-1), \tag{2.5}$$

де $I(x,y)$ - інтенсивність пікселя на позиції (x, y).

Отримай результат використовується для обчислення величини та напрямку градієнта, що виражається як величина M та напрямок θ градієнта для кожного пікселя можуть бути обчислені як:

$$M(x, y) = \sqrt{G_x^2 + G_y^2}, \tag{2.6}$$

$$\theta(x, y) = atan2\left(\frac{G_y}{G_x}\right), \tag{2.7}$$

де atan2 - функція, що повертає арктангенс двох чисел, яка враховує знаки обох аргументів для визначення квадранта кута.

Далі, для кожної комірок гістограма орієнтованих градієнтів обчислюється шляхом підрахунку кількості градієнтів кожного напрямку. Напрямки градієнтів зазвичай розділяються на 9 до 18 інтервалів (або "бінів"), що відповідає 0-180° або 0-360°. Гістограми комірок групуються в більші блоки, і для кожного блоку гістограми нормалізуються для зниження впливу змін освітлення. Нормалізація може бути виконана, наприклад, за допомогою методу L2-norm:

$$v = \sqrt{\|v\|_2^2 + \epsilon^2}, \tag{2.8}$$

де v - вектор гістограми комірки, а ϵ - мале число, що запобігає діленню на нуль.

Після обчислення та нормалізації гістограм для всіх блоків, вони об'єднуються в один великий вектор особливостей ГОГ, який використовується

для подальшої класифікації або розпізнавання з використанням машинного навчання, такого як ПВМ (Підтримуюча векторна машина).

Цей вектор особливостей використовується як вхід для класифікатора (наприклад, ПВМ) для тренування та визначення того, чи містить зображення конкретний об'єкт, наприклад, людську фігуру [8].

2.4 Відповідність шаблону в OpenCV

Відповідність шаблону в OpenCV — це метод пошуку і знаходження місця зображення, яке найкраще відповідає шаблону (темплейту) в більшому зображенні. Основна ідея полягає в тому, щоб "просканувати" шаблон по всьому зображенню (як зазвичай в слайд-шоу) і порівняти відповідний регіон зображення з шаблоном.

Цей процес можна описати за допомогою кореляції між шаблоном T та вікном зображення I, яке має ті ж просторові розміри, що й шаблон. Це порівняння виконується для кожного можливого положення шаблону на зображенні.

Для кожного положення (x,y), функція відповідності f(x,y) може бути визначена різними способами. Однією з найпростіших є сума абсолютних різниць (Sum of Absolute Differences, SAD):

$$f(x,y) = \sum_{i=0}^{w}\sum_{j=0}^{h}|I(x+i,y+j) - T(i,J)|, \tag{2.9}$$

де w і h — ширина і висота шаблону відповідно;

$I(x+i,y+j)$ і $T(i,j)$ — інтенсивності відповідних пікселів зображення та шаблону.

Інший спосіб — це сума квадратів різниць (Sum of Squared Differences, SSD):

$$f\ (x,y) = \sum_{i=0}^{w}\sum_{j=0}^{h}[I\ (x+i,y+j) - T(i,j)]^2. \qquad (2.10)$$

Для нормалізованого перехресного кореляційного коефіцієнта (Normalized Cross-Correlation, NCC):

$$f(x,y) = \frac{\sum_{i=0}^{w}\sum_{j=0}^{h}|I(x+i,y+j) - T(i,J)|}{\sqrt{\sum_{i=0}^{w}\sum_{j=0}^{h} I(x+i,y+j)^2 \cdot \sum_{i=0}^{w}\sum_{j=0}^{h} T(i.j)^2}}. \qquad (2.11)$$

NCC є особливо корисним, оскільки він враховує яскравість та контраст шаблону й зображення.

Після обчислення $f(x,y)$ для кожної позиції, місцеположення з найкращим збігом визначається максимумом (для NCC) або мінімумом (для SAD і SSD).

В OpenCV цей процес виконується функцією **matchTemplate**, і результатом є зображення, де кожен піксель відповідає значенню функції відповідності для відповідного положення шаблону. Місцеположення найкращого збігу може бути знайдено за допомогою функції **minMaxLoc** [9].

2.5 Контурний аналіз

Контурний аналіз в комп'ютерному зорі — це метод, який використовується для виявлення та аналізу контурів на зображеннях. Контур — це просто границя або замкнена крива, яка описує форму об'єкта.

Виявлення контурів засноване на техніці, яка шукає різниці у кольорі або інтенсивності між об'єктом та його фоном. Математично це може бути виражено через градієнти зображення. Наприклад, градієнт зображення I в позиції (x,y) може бути визначений як:

$$\nabla I(x,y) = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{dy} \end{bmatrix}, \quad (2.12)$$

де Gx та Gy — це горизонтальні та вертикальні компоненти градієнту відповідно. Контур знаходиться там, де величина градієнту досягає локального максимуму, що часто відбувається на межі між об'єктами та фоном.

Після того, як контур був виявлений, він може бути описаний за допомогою різних математичних підходів:

Чотирикутник, що описує контур (Bounding Rectangle): Можна обчислити мінімальний чотирикутник, що охоплює контур. Це дає простий опис форми та розміру об'єкта.

Моменти контуру: Моменти — це середні ваги інтенсивностей пікселів, які можуть використовуватися для обчислення центру мас, площі, орієнтації та інших характеристик контуру:

$$M_{ij} = \sum \sum Xi \cdot Yi \cdot I(x,y). \quad (2.13)$$

Форму контуру можна описати за допомогою різних описувальних функцій, таких як:

1. Густина пікселів уздовж контуру.
2. Функція відстані, яка вимірює відстань від центру мас до кожної точки на контурі.
3. Фур'є-дескриптори, які представляють контур в частотному домені.

Hu's Moment Invariants: Це набір із семи чисел, обчислених на основі моментів, які є інваріантними до зміни масштабу, перенесення, обертання та відображення. Вони визначаються за допомогою нескладних комбінацій моментів низького порядку.

Використовуючи ці математичні моделі, можна провести кількісний аналіз контуру, класифікувати його та використовувати для розпізнавання об'єктів на зображеннях.

2.6 Сегментація зображень

Сегментація зображень — це процес розділення зображення на декілька частин або сегментів, які є значущими у контексті певних задач. Мета сегментації — спростити або змінити представлення зображення для полегшення його аналізу. Сегменти часто відповідають різним об'єктам або областям зображення. Математично сегментацію можна описати декількома способами:

Одним з найпростіших методів сегментації є порогова обробка вона ж сегментація На Основі Порога (Thresholding), де кожному пікселю зображення I присвоюється мітка на основі його інтенсивності:

$$\begin{cases} 1 \; if \; I(x,y) > T \\ 0 \qquad \text{Інакше}' \end{cases} \tag{2.14}$$

де T — порогове значення, а $S(x,y)$ — сегментоване зображення.

Сегментація На Основі Кластеризації (Clustering) – це методи, такі як k-means, використовуються для групування пікселів на основі їхніх характеристик, таких як колір, текстура або інтенсивність:

$$\arg\min s \sum_{i=1}^{k} \sum x \in S_i \|I(x) - \mu_i\|^2, \quad (2.15)$$

де $I(x)$ — вектор ознак пікселя x;

Si — і-й кластер, μi — центроїд і-го кластера;

k — кількість кластерів.

Також існує сегментація На основі границь (Edge-Based Segmentation). Методи границь шукають перепади інтенсивності в зображенні, які часто відповідають межам об'єктів:

$$\nabla I(x,y) = \sqrt{(\partial I_x)^2 + \left(\partial I_y\right)^2}, \quad (2.16)$$

де $\nabla I(x,y)$ — магнітуда градієнта зображення в точці (x,y);

∂xI та ∂yI — часткові похідні інтенсивності по горизонталі та вертикалі відповідно [10].

Сегментація На Основі Регіонів (Region-Based Segmentation), як наприклад алгоритм затоплення (region growing), об'єднують пікселі або підобласті зображення, які мають подібні характеристики:

$$S = U_{i=1}^{n} R_i, \quad (2.17)$$

$$\forall R_i, R_j, if\ i \neq j : R_i \cap R_j = \emptyset, \quad (2.18)$$

$$\forall R_i : P(R_i) = 1(True), \quad (2.19)$$

де S — сегментоване зображення; Ri — регіон зображення;

$P(Ri)$ — предикат, який визначає, чи має регіон Ri однорідні характеристики.

Сегментація На Основі Моделі (Model-Based Segmentation)

Методи, як активні контури або змії (snakes), використовують параметричні моделі для фітингу контурів об'єктів в зображенні:

$$E_{total} = E_{internal}\big(v(s)\big) + E_{image}\big(v(s)\big) + E_{con}\big(v(s)\big), \quad (2.20)$$

де $v(s)$ — параметрична крива на зображенні;

Etotal, Einternal, Eimage,Econ — енергетичні функції, які визначають внутрішню енергію кривої, зображення і обмежуючі умови.

Кожен із цих методів має свої переваги та недоліки, і вибір методу залежить від специфіки задачі. Сучасні підходи до сегментації часто використовують глибоке навчання, де нейронні мережі, зокрема мережі глибокої семантичної сегментації, можуть навчатися визначати складні структури на зображеннях, у таблиці 2.1 наведені переваги та недоліки методів сегментації.

Таблиця 2.1 Переваги та недоліки методів сегментації

Метод	**Переваги**	**Недоліки**
Порогова обробка (Thresholding)	Простий у реалізації; швидкий; ефективний для простих зображень.	Неадаптивний до змін освітлення; важко вибрати глобальний поріг для складних зображень.
Кластеризація (Clustering)	Не потребує заздалегідь визначених порогів; може виявляти більш складні структури.	Обчислювально більш вимогливий; може бути чутливим до шуму; потребує визначення кількості кластерів.

Границі (Edge-Based Segmentation)	Добре виявляє краї об'єктів; корисний для форми аналізу.	Може бути чутливим до шуму; не завжди добре справляється з перекриттями об'єктів.
Регіони (Region-Based Segmentation)	Хороший для зображень із поступовими змінами інтенсивності; може об'єднувати суміжні області з однорідними властивостями.	Може бути вимогливим до обчислювальних ресурсів; результати залежать від вибору початкових точок.
Модель (Model-Based Segmentation)	Гнучкий; може інтегрувати попередні знання про форму об'єктів; хороший для слідкування рухомих об'єктів.	Вимагає складних обчислень; ініціалізація моделі може значно впливати на результати.

Провівши аналіз алгоритмів ідентифікації об’єктів методу оптичного контролю, було досліджено унікальність та ефективність кожного алгоритму вирішення конкретних завдань контролю розмірів головки поршня. Впровадження відповідних алгоритмів у оптичну систему, що розробляється, може значно поліпшити точність, швидкість та автоматизацію процесу вимірювань, зменшуючи при цьому вплив людського фактору на результати.

3 РОЗРОБЛЕННЯ СИСТЕМИ ОПТИЧНОГО КОНТРОЛЮ

3.1 Структура оптичної системи контролю

Комп'ютер - це електронний пристрій, що виник для полегшення математичних операцій людини, результати яких можна було подати у зрозумілій формі для користувача. Сьогодні важко уявити сучасне життя без персонального комп'ютера (ПК), який дозволяє опрацьовувати та створювати значну кількість інформації, а також отримувати та засвоювати її за допомогою ПК. На рисунку 3.1 наведено структурну схему роботи одноплатного комп'ютера Raspberry PI, до якого підключена відеокамера для отримання відеосигналу, необхідного для правильної роботи системи оптичного контролю.

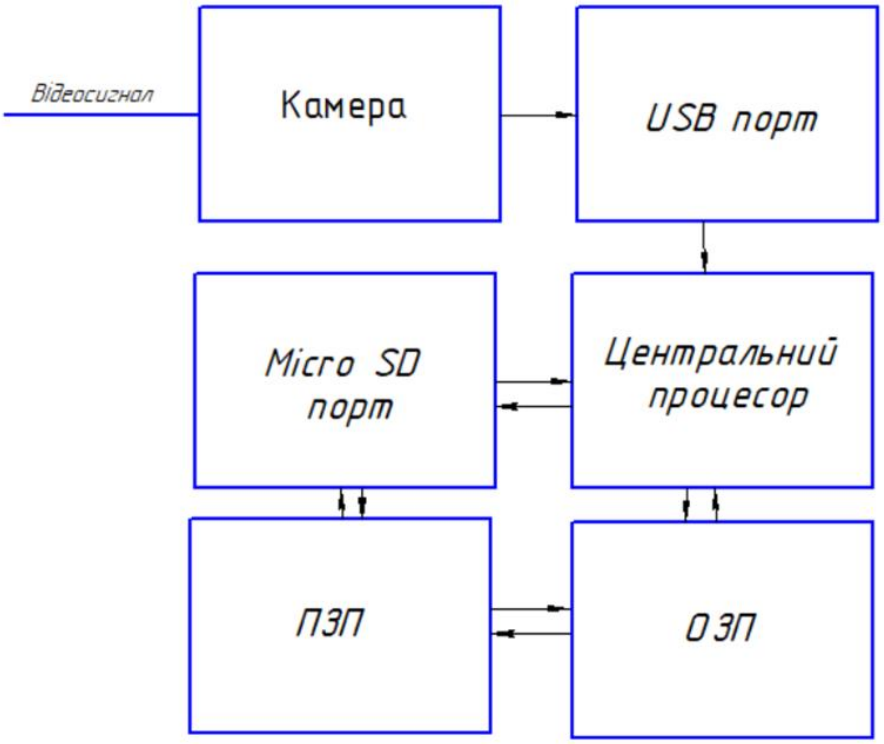

Рисунок 3.1 - Структурна схема системи оптичного контролю

Відеосигнал у систему потрапляє за допомогою відеокамери A4Tech PK-910H, яка надає можливість отримувати зображення в високій роздільній здатності, стандарт FHD (1920 на 1080 пікселів), що є оптимальним для системи оптичного контролю. З'єднання відеокамери з системою здійснюється через USB-порт версії 2.0, чия пропускна здатність достатня для оброблення інформації з відеокамери. Після цього зображення потрапляє до центрального процесора (ЦП) Broadcom BCM2837 архітектури ARM, який містить чотири ядра Cortex-A53 і працює на частоті 1,2 ГГц. Потужності процесора достатньо для оброблення програмного коду, описаного у розділі 2.1. Система працює стабільно протягом усього робочого процесу системи. У свою чергу, всі компоненти системи спілкуються з ЦП, включаючи постійний та оперативний запам'ятовуючі пристрої, ПЗП та ОЗП відповідно.

Постійна пам'ять реалізована у вигляді карти пам'яті, яка підключена до системи через відповідний порт для карт пам'яті. Її об'єм залежить від вибору самої карти пам'яті; у даному випадку об'єм ПЗП становить 32 ГБ. Карта пам'яті містить операційну систему Raspbian OS, а також файли з програмним кодом системи оптичного контролю. У оперативну пам'ять об'ємом 1 ГБ типу LPDDR2 поміщаються дані, з якими комп'ютер працює у даний момент для більшої швидкодії, оскільки об'єм ОЗП значно більший, ніж ПЗП [11]. Структурна схема подана на рис. МР.МТТм-19.00.00.000 Е1.

3.2 Вибір мови програмування

Вибір мови програмування становить початковий етап у розробці програмного забезпечення, оскільки це визначає подальший розвиток проєкту. На ринку існує безліч мов програмування, які можуть відповісти вимогам для реалізації концепції такої системи. Однак основними лідерами можна вважати

C++ та Python, оскільки саме в цих мовах можна використовувати бібліотеку комп'ютерного зору - OpenCV. Мова програмування Python приваблює простотою освоєння завдяки лаконічному синтаксису, що робить її доступною для початківців у програмуванні. Додатковою перевагою Python є його активне спільнота, яка постійно розробляє безліч корисних бібліотек для використання у різних проектах. З урахуванням досвіду використання мови Python було вирішено обрати саме її для написання системи оптичного контролю радіусу об'єктів.

OpenCV (Open Source Computer Vision Library) - це відкрита бібліотека комп'ютерного зору та машинного навчання. OpenCV була створена для надання загальної інфраструктури для застосунків комп'ютерного зору та для прискорення використання машинного сприйняття у комерційних продуктах. Як продукт з ліцензією BSD, OpenCV полегшує компаніям використання та модифікацію коду. Бібліотека має понад 2500 оптимізованих алгоритмів, які включають у себе як класичні, так і сучасні алгоритми комп'ютерного зору та машинного навчання. Ці алгоритми можуть використовуватися для виявлення та розпізнавання облич, ідентифікації об'єктів, класифікації дій людини на відео, відстеження рухів камери, відстеження рухомих об'єктів, створення 3D моделей об'єктів, виробництва 3D хмар точок зі стереокамер, складання зображень для виробництва зображення високої роздільної здатності цілого сценарію, пошуку подібних зображень з бази даних зображень, усунення ефекту червоних очей з зображень, зроблених за допомогою спалаху, відстеження рухів очей, розпізнавання сценерії та встановлення маркерів для їх накладання з доповненою реальністю тощо.

Розробка програмного забезпечення (ПЗ) розпочинається з проектування майбутнього функціоналу. В першу чергу важливо, щоб ПЗ працювало в реальному часі та могло відображати відеосигнал у вікні. З метою уникнення проблем з оптичним ефектом, програма повинна бути здатна взаємодіяти з

калібрувальною міткою. Крім того, біля вимірювального об'єкта ПЗ повинно виводити його розмір, вимірювати абсолютну та відносну похибки.

Невід'ємною частиною ПЗ є робота з базою даних, де зберігаються дані про вимірювальні об'єкти. Ці дані включають розмір, максимальну допустиму похибку, назву та коментар оператора до об'єкта.

Окрім того, для зручності взаємодії з ПЗ передбачено створення графічного інтерфейсу. Це надасть можливість користувачеві взаємодіяти з базою даних, додавати та видаляти об'єкти вимірювання, а також обирати конкретний об'єкт для вимірювання системою.

3.3 Опис алгоритму роботи програмного забезпечення

Виконання програми розпочинається із налаштування графічного інтерфейсу. Після його налаштування перед користувачем відкривається інтерфейс програми.

За допомогою полів для введення інформації, користувач може внести дані про об'єкт до бази даних. Програма зчитує дані, введені у комірки, та створює об'єкт у базі даних із відповідною інформацією. Кожному об'єкту присвоюється унікальний порядковий номер для зручності в програмному коді.

Крім того, користувач може використовувати поля для введення даних для видалення об'єкта з бази даних у разі потреби. Після введення порядкового номера об'єкта програма видаляє його із бази даних, якщо об'єкт існує. У випадку відсутності об'єкта виводиться повідомлення про помилку.

Для вказівки програмі на об'єкт, який слід виміряти, користувач вказує його порядковий номер. Дані про об'єкт потім відображаються у відповідних

комірках, визначених для активного об'єкта. Система отримує статус 1, що вказує на вибір об'єкта для вимірювання. У випадку невибору об'єкта статус дорівнює 0.

Робота програми з вимірювання розпочинається натисканням кнопки із зображенням камери. Після натискання починається ініціалізація відеосигналу. Алгоритм програмної бібліотеки OpenCV сканує систему наявність відеокамери, і якщо вона знайдена, відкривається вікно із зображенням з камери.

Після відкриття камери наступним кроком є пошук об'єктів у кадрі, які мають форму кола. Це вдається завдяки використанню бібліотеки OpenCV та нейронних мереж. Система вже знає, як повинні виглядати круги, і намагається знайти схожі об'єкти.

Паралельно з пошуком об'єктів відбувається ініціалізація Aruco-маркерів на кадрі. Вони необхідні для калібрування системи та усунення оптичних ефектів. При ініціалізації завантажується список розмірів маркерів. Після цього алгоритм починає пошук маркерів у кадрі з використанням завантажених розмірів. Якщо маркер не знайдено у кадрі, на екран виводиться графічне відображення не знайденого маркера у вигляді тексту "ANF" (Aruco not found), а також текст фарбується жовтим кольором. У подальшому всі негативні позначення отримують жовтий колір тексту. В програмі встановлюється коефіцієнт Aruco, який дорівнює 1.

У разі знаходження маркера на кадрі, знаючи його фізичний розмір, визначається його розмір у пікселях. Ця інформація дозволяє встановити відношення пікселя до сантиметра, що надає розмір об'єкта у сантиметрах. Значення записується в змінну "aruco", яка в даному випадку дорівнює 29,3 пікселя на один сантиметр.

У випадку знаходження об'єкта відбувається його морфінг - це спеціальний ефект, який створює плавний перехід між об'єктами. У цьому випадку алгоритм згладжує різкі краї кола для більш точної обробки.

Після виявлення об'єкта на кадрі та його згладжування, алгоритм фіксує значення x, y та r. Тут x та y вказують на координати центру кола, а r - радіус об'єкта, тобто відстань від центру до його краю. Усі ці значення вимірюються в пікселях. Далі алгоритм перетворює значення радіуса об'єкта в сантиметри, розділяючи його на коефіцієнт Aruco.

Після цього алгоритм здійснює постпроцесінг - виведення інформації на екран. Розмір отриманого об'єкта відображається на ньому, враховуючи його координати центру, а також графічно відображується центр об'єкта. Об'єкт, який вимірюється, обводиться червоною лінією.

Далі алгоритм перевіряє статус програми, звертаючись до бази даних і витягуючи відповідну комірку. Якщо ця комірка дорівнює 0, це вказує, що система працює у режимі лінійки (ruler mode), і не виконує вимірювання похибок. У цьому випадку дані про останній вимірювальний об'єкт залишаються на екрані і позначаються жовтим кольором.

Якщо відповідна комірка в базі даних має значення 1, розпочинається процес розрахунків похибок. Програма витягує з комірок активного об'єкту в базі даних його розмір та максимально допустиму похибку, а також розмір об'єкта, який вимірюється. Проводиться розрахунок абсолютної похибки, і після цього система перевіряє об'єкт на його валідність.

Якщо абсолютна похибка перевищує максимально допустиму, відбувається бракування деталі, і на екрані з'являється червоний напис "STATUS". Це дозволяє оператору візуально визначити бракований об'єкт. У випадку, якщо об'єкт знаходиться в межах допустимого, на екрані також

з'являється напис "STATUS", але зеленого кольору, що свідчить про те, що все в порядку.

Одночасно з цим розраховується відносна похибка вимірювального об'єкту, і її значення виводиться на екран, позначаючи фіолетовий колір. Це полегшує оператору ідентифікацію похибок серед іншої інформації.

Для зручності оператора програмний алгоритм виводить на екран певну інформацію, взяту з бази даних:

1. Інформація про розмір вимірювального об'єкту;

2. Інформація про максимально допустиму похибку об'єкту;

3. Інформація про режим роботи програми;

4. Інформація про статус об'єкту.

Увесь програмний код, починаючи з ініціалізації відеосигналу, виконується в безкінечному циклі, забезпечуючи реальний час роботи програми. Функціональна схема роботи алгоритму подана на рис. МР.МТТм-19.00.00.000 Е2.

3.4 Розроблення програмного забезпечення

Для зручної взаємодії з програмним забезпеченням (ПЗ), необхідно розробити графічний інтерфейс (далі - ГІ), який спростить роботу оператора та надасть ПЗ привабливий вигляд. Існує різноманітні методи створення ГІ для ПЗ, починаючи від повністю ручних підходів до використання напівавтоматичних конструкторів, які дозволяють створювати інтерфейс за кілька хвилин. Для мови програмування Python доступна потужна бібліотека для створення графічних

оболонок - PyQt. Вона включає в себе велику кількість елементів для створення різноманітних ГІ, що дозволяє використовувати її в різних сценаріях.

Створення ГІ з використанням PyQt може відбуватися кількома способами, вручну або за допомогою конструктора графічних інтерфейсів PyQt - Qt Designer. У разі використання Qt Designer не існує потреби вручну створювати елементи ГІ, що спрощує процес. Таким чином, для даного ПЗ було обрано створення ГІ за допомогою конструктора Qt Designer.

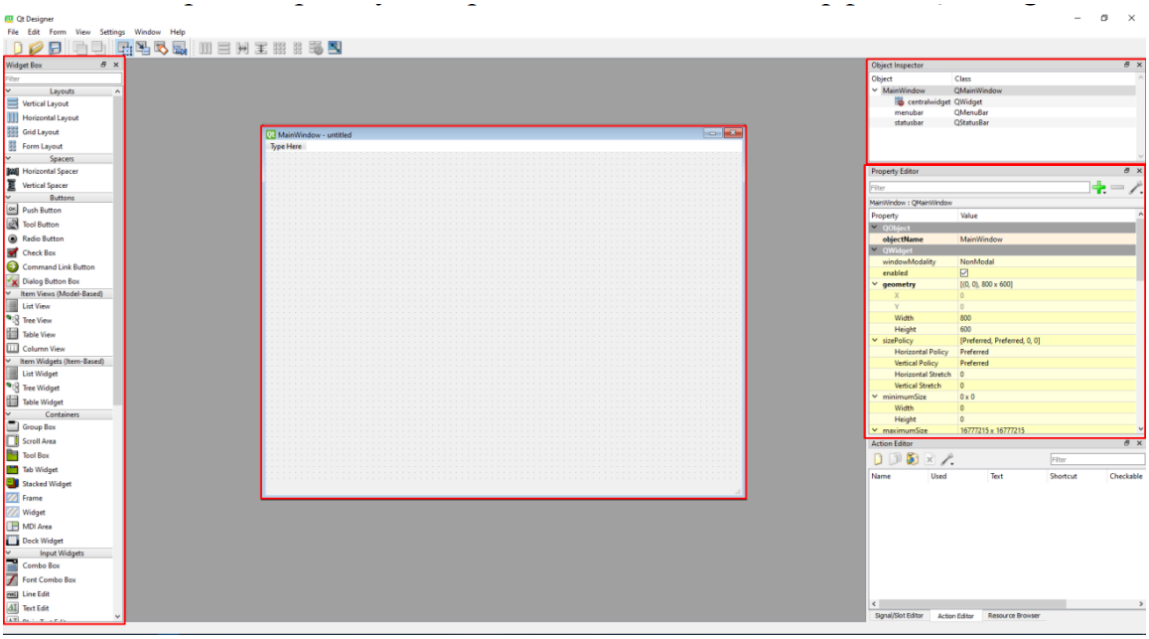

Рисунок – 3.2 Інтерфейс Qt Designer

На зображенні 3.2 представлений графічний інтерфейс, створений за допомогою Qt Designer, який включає наступні компоненти:

1. Widget box - Знаходиться у лівій частині програми і представляє собою блок-список графічних елементів, які можна використовувати для створення користувацьких графічних інтерфейсів.
2. Центральна область екрану - Тут розташований поточний вигляд користувацького графічного інтерфейсу, де можна розміщувати та редагувати елементи.

3. Object inspector - Розташований у правій частині екрану, цей блок є списком об'єктів інтерфейсу, які були додані до проєкту з Widget box.
4. Property Editor - Цей блок містить параметри об'єктів, які можна редагувати, щоб налаштувати їхні властивості [12].

Після вивчення інструменту Qt Designer був створений користувацький графічний інтерфейс для системи оптичного контролю об'єктів. В результаті цього графічний інтерфейс системи набув вигляду, представленого на рисунку 3.2.

Рисунок 3.3 - Графічний інтерфейс систему оптичного контролю

Графічний інтерфейс системи контролю об'єктів має такі компоненти:

1. Вікно бази даних - Показує список об'єктів вимірювання, які знаходяться в базі даних;

2. Поле для вводу порядкового номера – Служить для введення порядкового номера об'єкта в базі даних;
3. Кнопка вибору об'єкта – Активує об'єкт вимірювання, вибраний з введеного номера;
4. Поле статусу бази даних – Відображає інформацію про стан роботи бази даних;
5. Поля для введення інформації про об'єкт вимірювання – Включають різні характеристики об'єкта, які можна ввести;
6. Кнопка збереження інформації – Додає введену інформацію в базу даних;
7. Кнопка видалення об'єкта – Видаляє вимірювальний об'єкт, враховуючи введений порядковий номер;
8. Кнопка запуску камери – Запускає процес вимірювання;
9. Кнопка скасування вибору об'єкта – Скасовує вибір вимірювального об'єкта і переводить систему у режим звичайної лінійки.

3.5 Розроблення програмного забезпечення ідентифікації та вимірювання головки поршня

Впроваджено за допомогою потужної бібліотеки комп'ютерного зору - OpenCV. Ця бібліотека використовується для проведення пошуку, ідентифікації та отримання розмірів об'єктів у кадрі за допомогою нейронних мереж, які є частиною OpenCV. Це значно полегшує завдання розробки системи.

Перед розробкою основної програми важливо написати функцію для створення бази даних. Ця функція перевіряє наявність бази даних з вказаною назвою в директорії. Якщо база даних не існує, вона створюється за допомогою бібліотеки баз даних для Python - sqlite3. База даних включає дві таблиці: "objects", в якій містяться всі об'єкти вимірювання, додані користувачем, та

"current", яка містить один запис інформації про вибраний об'єкт вимірювання та статус системи.

```
def create_db(db_file):
    conn = None
    try:
        conn = sqlite3.connect(db_file)
    except Error as e:
        pass
    finally:
        if conn:
            sql = conn.cursor()
            conn.execute("CREATE TABLE IF NOT EXISTS objects (ID INTEGER, Name
TEXT, Size REAL, Tol REAL, Comment TEXT)")
            conn.execute("CREATE TABLE IF NOT EXISTS current (status INTEGER,
Name TEXT, Size TEXT, Tol TEXT, Ind INTEGER)")
            conn.commit()
            sql.execute('SELECT status FROM current')
            index = sql.fetchone()
            if index == None:
                conn.execute(f"INSERT INTO current VALUES (?, ?, ?, ?, ?)",
(int(25), str(0), str(0), str(0), int(0)))
                conn.commit()
                conn.close()
            else:
                conn.close()
```

Код алгоритму обробки зображення знаходиться у файлі з назвою «_OpenCV.py» , де «_OpenCV» - це назва файлу, а «.py» - формат файлу, скорочений від назви мови програмування Python. Де. Для зручності весь наступний код розміщений у класі "Output". У цьому класі відбуваються всі процеси, пов'язані з обробкою зображення. Основною функцією є "frame", яка відповідає за отримання зображення від джерела. Спочатку у функції оголошується змінна "cap", в яку фактично завантажується зображення з відеокамери. Після цього відкривається безкінечний цикл для формування відеопотоку.

```
class Output:

    @staticmethod
    def frame():
        cap = cv2.VideoCapture(0)
```

Для запису зображення в змінну image, ми ініціалізуємо процес read для змінної яка звертається до відеосигналу.

```
# Select method video or image
ret, image = cap.read()
#image = cv2.imread('test image.jpg')
```

Крім цього є можливість вводу тестового зображення, для цього потрібно закоментувати змінну відеосигналу, та розкоментувати змінну з тестовим зображенням.

Маркери Aruco є інструментом, що дозволяє системі орієнтуватися у просторі за допомогою визначеної мітки або трекеру. Ця мітка також відома як калібрувальна мітка або точка відліку. Використовуючи Aruco мітку, система оптичного контролю може вимірювати об'єкти у стандартних одиницях СІ. Цей маркер також допомагає уникнути зміни візуальних розмірів об'єкта при зміні відстані до нього, запобігаючи оптичному ефекту.

Маркери Aruco насправді представляють собою QR-коди у системі OpenCV і існують у фіксованих розмірах, таких як 4x4, 5x5, 6x6, 7x7. Це відповідає їх фізичним розмірам, і система розпізнає розмір мітки відповідно до її зображення.

```
try:
    # Aruco setup
    parameters = cv2.aruco.DetectorParameters_create()
    aruco_dict = cv2.aruco.Dictionary_get(cv2.aruco.DICT_5X5_50)
    corners, _, _ = cv2.aruco.detectMarkers(image, aruco_dict, parameters =
parameters)

    # Aruco draw border
    int_corners = np.int0(corners)
    #cv2.polylines(image, int_corners, True, (0, 255, 0), 1)

    # Aruco get perimeter
    aruco_perimeter = cv2.arcLength(corners[0], True)

    # Text Label for Aruco status
    cv2.putText(image, "ACF", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(107,255,53), 3)
except(IndexError):
    aruco_perimeter = 1
    cv2.putText(image, "ANF", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1,
(53,255,236), 3)

# Pixel graduation
pixel_cm_ratio = aruco_perimeter / 20

# Get origin aruco size on cm
aruco_size = (((aruco_perimeter / pixel_cm_ratio) / 4))
aruco_size = (aruco_size * sqrt(2)) / 2
aruco_size = round(aruco_size, 3)
```

Під час початкової настройки Aruco у систему завантажується список маркерів розміром 5x5 у змінну "aruco_dict". Периметр маркера записується у змінну "aruco_perimeter". Знайдені маркери заносяться у масив "corners". Система читає дані з цього масиву і виводить на екран повідомлення "ACF", що свідчить про успішне знаходження маркера. У випадку, якщо мітка не знайдена, масив буде порожнім, і система виведе помилку "IndexError", яку перехоплює та виводить текст "ANF", що означає відсутню мітку. Також периметр дорівнює 1, щоб не порушити структуру роботи програми. Після цього отримуємо коефіцієнт, який дозволяє переводити пікселі у сантиметри, і поміщаємо його у змінну "pixel_to_cm_ratio".

Вхідне зображення піддається обробці, а саме, перетворенню в градієнт сірого, розмиття та перетворення в бінарне зображення, це подано у змінних «gray», «blur», «thresh» відповідно.

```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (3,3), 0)
thresh = cv2.threshold(blur, 0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
```

Наступним кроком є пошук та згладжування країв вимірювального об’єкту для більш коректної роботи системи, цей процес називається морфінгом.

```
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5,5))
opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=2)
```

Відповідно, наступним кроком у програмі є отримання контурів кіл та занесення всіх можливих об’єктів, які потрапили до кадру, у масив, після чого, цикл for перебирає усі елементи масиву та до кожного з них рисує поверх зображення його розмір, який записаних у змінну «object_radius» відповідно з застосуванням коефіцієнту.

```
cnts = cv2.findContours(opening, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]
for c in cnts:
    cv2.putText(image, f"  ", (10, 150), cv2.FONT_HERSHEY_SIMPLEX, 1, (53, 255,
236), 3)

    # Get Width and Height of the Objects by applying the Ratio pixel to cm
    (x, y), r = cv2.minEnclosingCircle(c)
    r = round(r, 3)
```

```
    # Convert px to radius in cm
    object_radius = round(((r / pixel_cm_ratio) / 2.1), 2)
    cv2.putText(image,
                f"R{str(object_radius)} cm",
                (int(int(size_checker(int(y), int(r), int(x))[1])),
                 int(size_checker(int(y), int(r), int(x))[0])),
                cv2.FONT_HERSHEY_SIMPLEX,
                text_size(r)[0], (0, 0, 255),
                text_size(r)[1])
```

Якщо відбувається процес вимірювання певного об’єкта, з бази даних записуємо розмір та максимальну похибку у змінні «obj_size» - розмір об’єкта та «obj_tol» - відхилення об’єкта.

```
# Init current data about object
obj_size = db_connect()[1]
obj_tol = db_connect()[2]
```

Для визначення поточного режиму роботи системи була створена функція "target_name", яка представлена на рисунку 3.11. Ця функція аналізує статус системи в базі даних і, відповідно до отриманої інформації, повертає відповідний статус у формі тексту. "TNF" вказує, що об'єкт вимірювання не вибраний, і система працює у режимі лінійки. "TCF" вказує, що об'єкт вимірювання вибрано, і можливий процес вимірювання.

```
def target_name():
    index = db_connect()[3]
    if index == 0:
        status_name_not_found = "TNF"
        return status_name_not_found
    if index == 1:
        status_name_found = "TCF"
        return status_name_found
    else:
        return "Error"
```

Функція «tolerance_measurement» відповідає за включення об'єкта вимірювання. Вона приймає розмір об'єкта у сантиметрах і перевіряє статус системи. Якщо статус дорівнює 1 (вибраний об'єкт), функція отримує дані об'єкта, такі як розмір і максимально допустима похибка. Після виконання математичних операцій визначається величина похибки. Якщо вона знаходиться в межах допустимого діапазону, повертається статус об'єкта, який відображається зеленим кольором, надаючи операторові зрозуміти, що все в порядку. У випадку перевищення похибки, статус об'єкта маркується червоним

кольором, і ведеться журнал, повідомляючи про дефектний об'єкт і вимагаючи його аналізу. Якщо статус системи рівний 0, статус об'єкта змінюється на "RULLER MODE" - це вказує на те, що система працює у режимі лінійки.

RULLER MODE - це особливий режим системи, в якому вона виконує звичайне вимірювання об'єктів, не проводячи при цьому контроль. Цей режим дозволяє виміряти радіус будь-якого об'єкта у кадрі.

```
def tolerance_measurement(measure_size):
    index = db_connect()[3]
    if index == 1:
        origin_name = db_connect()[0]
        origin_size = db_connect()[1]
        origin_tol = db_connect()[2]
        if float(measure_size) > float(origin_size) + float(origin_tol) or
float(measure_size) < float(origin_size) - float(origin_tol):
            status_time = datetime.datetime.now()
            status_color_red = 75, 63, 255
            text_status = "STATUS"
            log_text = f"{origin_name} A DEFECT AT {status_time}"
            #print(f"{float(measure_size) > float(origin_size) +
float(origin_tol)}, {float(measure_size) < float(origin_size) -
float(origin_tol)}")
            return text_status, status_color_red, log_text, 0
        elif float(measure_size) < float(measure_size) + float(origin_tol) or
float(measure_size) > float(measure_size) - float(origin_tol):
            status_time = datetime.datetime.now()
            status_color_green = 107,255,53
            text_status = "STATUS"
            #print(measure_size)
            return text_status, status_color_green, status_time, 1
    if index == 0:
        status_time = datetime.datetime.now()
        status_yellow_color = 53, 255, 236
        text_status = "RULLER MODE"
        return text_status, status_yellow_color, status_time, 2
    else:
        return "error"
```

Функція «percent» - визначає відносну похибку, У системі вона використовує розмір об'єкту як вхідний параметр X. Потім отримує дані з бази даних щодо розміру цього об'єкту і проводить математичні операції відповідно до формули для визначення відносної похибки.

```
def percent(x):
    if x != 0:
        origin_size = db_connect()[1]
        origin_size = float(origin_size)
        mistake = (x - origin_size)
        tolerance_percent = (mistake / origin_size) * 100
        tolerance_percent = round(tolerance_percent, 3)
        tolerance_percent = str(tolerance_percent)
```

```
        return tolerance_percent, 1
    else:
        return " "
```

Останнім етапом є відображення певної інформації на екрані для полегшення взаємодії з програмним забезпеченням. По завершенні алгоритму виконується наступне: малюється конкретна інфографіка, така як центр вимірювального об'єкту, лінія-радіус та контур об'єкта. На екран виводиться інформація щодо статусу об'єкта - чи був вибраний для вимірювання, його розміру та похибки. Додатково виводиться відносна похибка для зручності спостереження.

3.6 Демонстрація роботи системи оптичного контролю

На рисунку 3.4 представлений знімок екрану, демонструючи функціонал системи контролю. В якості об'єкту для тестування було використано нарисоване на листі А4 коло з радіусом 2 см. Система працює в режимі контролю, при цьому в базу даних завантажено об'єкт з назвою «Circle» розміром 2 см і максимально допустимою похибкою 0,3 см.

У верхньому лівому куті знаходяться індикатори системи, які розфарбовані у зелений колір, свідчачи про те, що система належним чином функціонує і перебуває в режимі контролю обраного об'єкта, чиї дані також відображені. Відносна похибка вимірювання виведена фіолетовим кольором в лівому нижньому куті. Статус програми також представлений в цьому куті, і в даному випадку він зелений, що вказує на те, що програма працює в режимі контролю, а об'єкт успішно пройшов перевірку. Розмір виміряного об'єкта становить 2,03 см, що входить в межі допустимої похибки 1,5%. Також, приклад роботи системи оптичного контролю поданий на рисунку МР.МТТм-19.00.00.001.

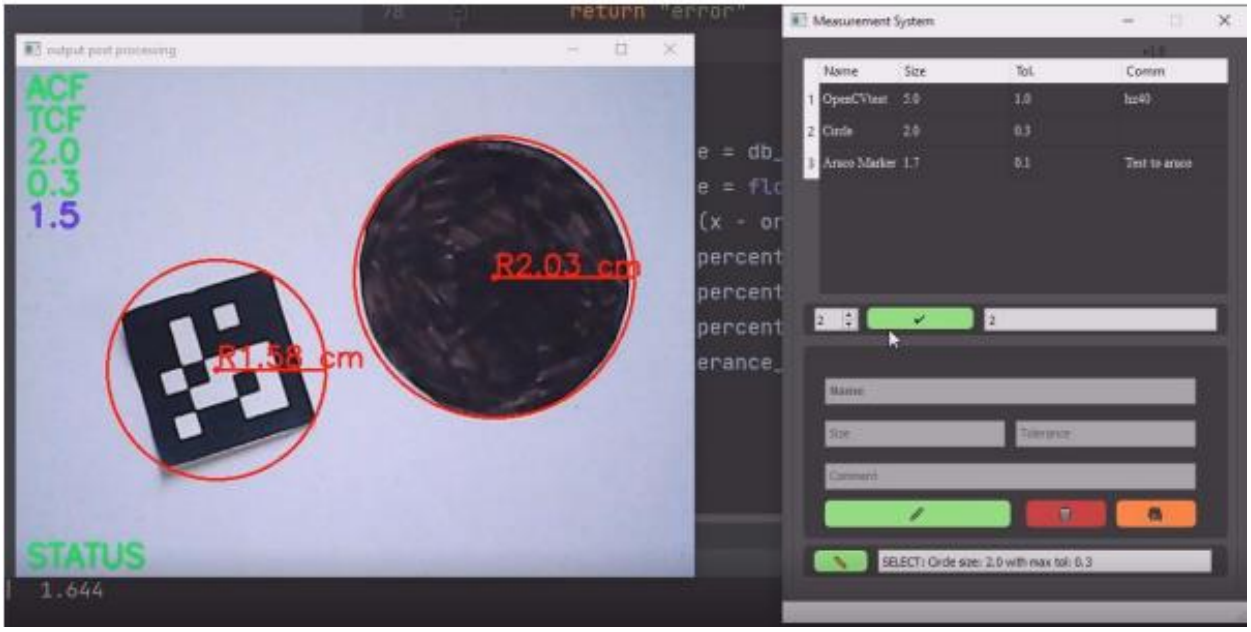

Рисунок 3.4 – Демонстрація роботи системи

У випадку, коли об’єкт для контролю не обрано, система автоматично перейде в режим лінійки, приймаючи форму, яку можна побачити на рисунку 3.5.

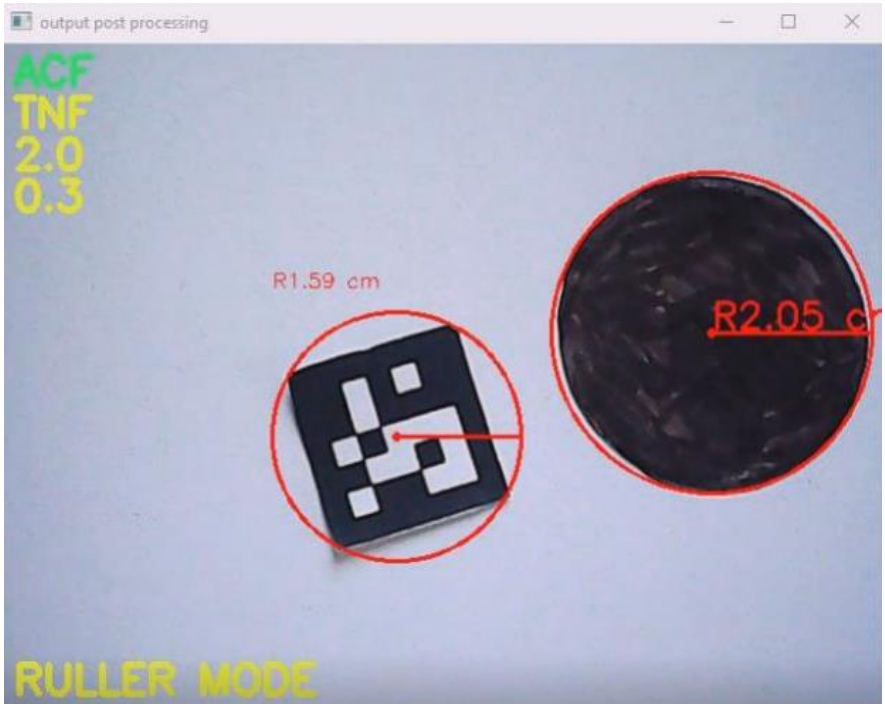

Рисунок 3.5 – Система у режимі лінійки

Основними технічними характеристиками системи оптичного контролю є наступні:

1. Роздільна здатність камери – Ця характеристика визначає, наскільки детально камера може передавати образ. Чим вища роздільна здатність, тим більше деталей буде зафіксовано на вихідному зображенні або відео. Зазвичай, камери, які використовуються в системах контролю, мають роздільну здатність від 640x480 до 1280x720 пікселів;
2. Максимальна кількість одночасних операцій – Цей параметр вказує на кількість об’єктів, які система може одночасно визначати. Значення 1, наприклад, означає, що система може виявляти лише один об’єкт на зображенні одночасно. Залежно від потужності системи, деякі продуктивні моделі можуть обробляти десятки або сотні об’єктів одночасно;
3. Пам’ять – Кількість доступної флеш-пам’яті визначається обсягом даних, які система може зберігати. Пам’ять використовується для зберігання бази даних об’єктів та іншої інформації. Більша кількість пам’яті дозволяє записувати більше об’єктів та інших даних.

Дана система завдяки кросплатформі OpenCV може працювати на різних операційних системах та платформах, таких як Windows (включаючи Windows 10, 8, та 7), Linux (Ubuntu, Fedora, Debian, тощо), macOS, Android, iOS, Raspberry Pi, а також вбудованих системах. OpenCV розширює свої можливості в області обробки зображень та комп’ютерного зору, забезпечуючи гнучкість для використання на різних пристроях та платформах. В моєму випадку, система розробляється та тестується на ОС Windows 10/11, приклади роботи якої подані на протязі магістерської роботи. Після проходження тестування, система буде розміщена на платформі Raspberry Pi на базі ОС Linux.

Розроблена система надає можливість вимірювати діаметри об'єктів за допомогою камери та програмних алгоритмів. Система може функціонувати як у режимі контролю об'єктів, так і у режимі звичайної лінійки для вимірювання діаметрів об'єктів на кадрі. Зручність використання системи підкреслюється її

взаємодією з базою даних, що спрощує збереження інформації про об'єкти вимірювання та уникає повторного введення даних. Програма оснащена зрозумілим інтерфейсом, що дозволяє використовувати її без необхідності спеціалізованих знань. Система автоматично працює без участі оператора, проте може виводити необхідну інформацію на екран для оцінки процесу оператором у разі потреби. Під час контролю в кадрі повинен бути присутній лише об'єкт контролю, але в режимі лінійки система може вимірювати декілька об'єктів одночасно.

4 РОЗРОБЛЕННЯ АЛГОРИТМУ ІДЕНТИФІКАЦІЇ І ВИМІРЮВАННЯ ГОЛОВКИ ПОРШНЯ ОПТИЧНИМ МЕТОДОМ

4.1 Аналіз кроків алгоритму

Для пошуку та вимірювання розмірів об’єкту в кадрі за допомогою OpenCV потрібно виконати певні кроки, які подані на рисунку 4.1.

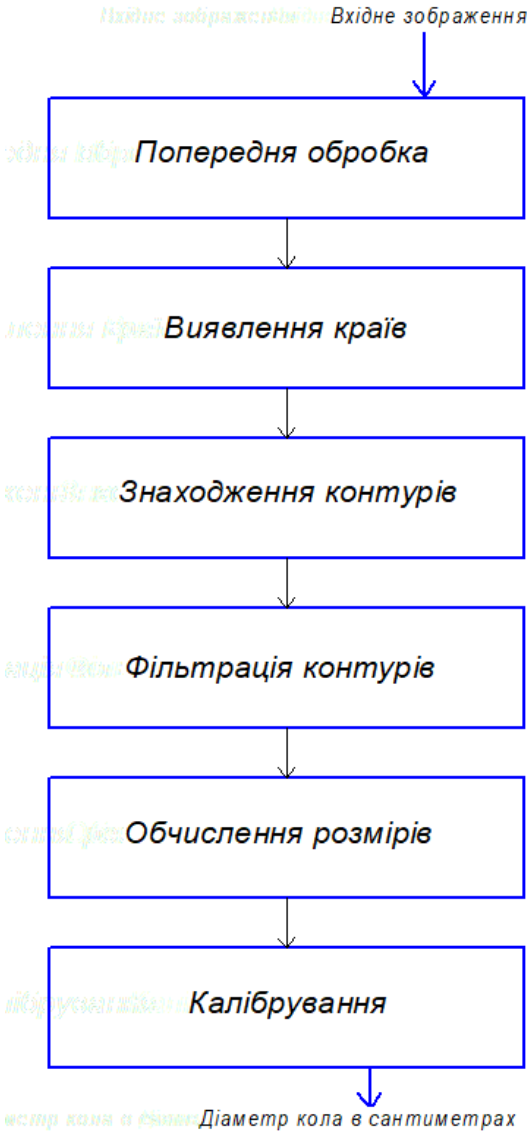

Рисунок 4.1 Алгоритм пошуку та вимірювання об’єкта системи

Попередня обробка зображення — це важливий етап в комп’ютерному зорі, який підготовляє зображення для подальшої обробки та аналізу. Метою попередньої обробки є підвищення якості зображення шляхом видалення шумів, покращення контрасту та виділення важливих особливостей.

Для початку, відбувається перетворення в градації сірого, його мета Зменшити складність обробки, перетворивши кольорове зображення в шкалу сірого, що зменшує кількість каналів з трьох (червоний, зелений, блакитний) до одного. Інтенсивність кожного пікселя на сірому зображенні — це зважене середнє кольорових каналів оригінального пікселя, яке враховує людську чутливість до різних кольорів.

Наступним пунктом попередньої обробки є розмиття зображення, метою якого є зменшити вплив шуму та несуттєвих деталей на зображенні, що спрощує процес виявлення країв та контурів. Цього можна досягти з використанням гаусового ядра, що дає менше розмиття на краях і більше у центрі.

Заключним єтапом попередньої обробки є порогове перетворення, метою якого є виділити об’єкти на зображенні, перетворивши зображення на бінарне, де пікселі з значеннями вище певного порогу встановлюються як білі, а нижче — як чорні, цього можна досягти методом глобального порогово перетворення, де відбувається встановлення одного порогового значення для всього зображення.

Виявлення країв з використанням Алгоритму Кенні (Canny) — це популярний алгоритм виявлення країв, який був розроблений Джоном Кенні у 1986 році. Він вважається одним з найефективніших алгоритмів виявлення країв і часто використовується в комп’ютерному зорі для виявлення лінійних країв у зображеннях. Зображення спочатку розмивається за допомогою Гаусового фільтра для зменшення шуму та несуттєвих деталей. Це робить краї більш чіткими та зменшує кількість помилкових крайових пікселів внаслідок шуму. Після розмиття обчислюється градієнт зображення для визначення напрямку та

інтенсивності зміни яскравості. Це робиться за допомогою операторів Собеля, які виявляють горизонтальні та вертикальні зміни. Немаксимальне придушення застосовується для “проріджування” країв. Для кожного пікселя перевіряється, чи є він максимумом у напрямку градієнта. Якщо не є, піксель придушується (його значення встановлюється в нуль), що призводить до тонких ліній (країв). Після чого, алгоритм застосовує два порогові значення для визначення справжніх країв. Сильні краї з інтенсивністю вищою за високий поріг визнаються як справжні краї, слабкі краї нижче низького порогу відкидаються, а всі інші краї, що знаходяться між двома порогами, позначаються як слабкі краї. В останньому кроці алгоритм вирішує, чи слід з’єднати слабкі краї з сильними. Якщо слабкий край з’єднаний з сильним краєм, він вважається справжнім краєм; інакше, він відкидається [13].

Знаходження контури на зображенні, використовуючи функцію findContours. Після знаходження контуру, відбувається фільтрування знайдених контури на основі різних критеріїв, таких як площа, форма або розмір, щоб ізолювати контур цільового об’єкту.

Для Вимірювання розмірів об’єкту, використовують функції, такі як «inEnclosingCircle» або «cv2.fitEllipse», для отримання висоти та ширини. Для системи контролю використовується функція «inEnclosingCircle».

Останнім кроком є калібрування. Щоб отримати реальні фізичні розміри, потрібно калібрувати систему за допомогою відомого посилання або масштабу, це реалізовано за допомогою Aruco Marker 5X5. Структурна схема роботи алгоритму пошуку та вимірювання об’єкта системи подана на рис. МР.МТТм-19.00.00.001 Е1.

Aruco маркери використовуються для позиціонування та орієнтування у 3D просторі, особливо в області доповненої реальності та робототехніки. Вони забезпечують швидке та надійне виявлення кутів та ідентифікацію.

Опис алгоритму буде здійснюватися на основі статичного тестового зображення, подане на рисунку 4.2, з колом радіусом 2 сантиметри, та 4 сантиметри діаметру відповідно. Доповнює його маркер Aruco програмованим розміром 5X5, та фізичним 2.5 см. Приклад тестового зображення наведено на рисунку 4.2.

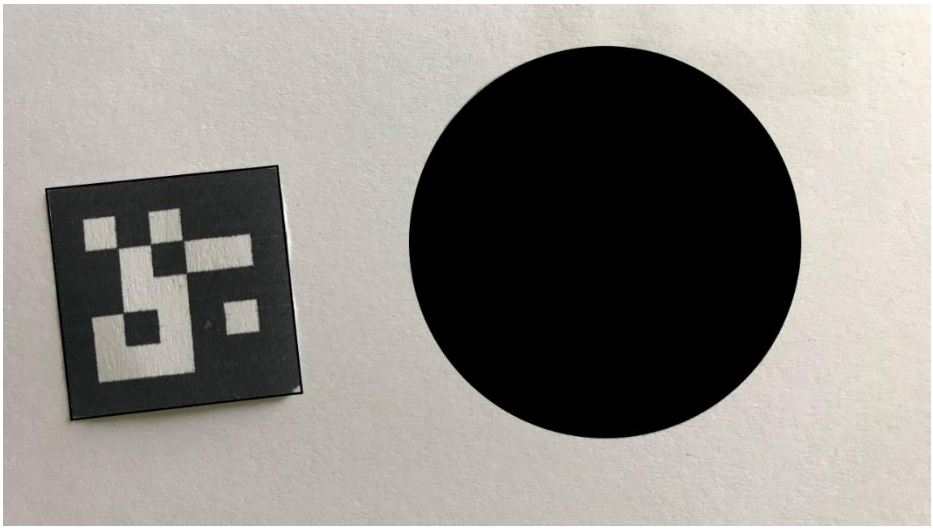

Рисунок 4.2 – Тестове зображення

4.2 Конвертація в градації сірого

У змінну gray кодується зображення, перетворене в градаціях сірого, це досягається за допомогою використання функції cvtColor, яка приймає зображення на прапорець кольору, в даному випадку «COLOR_BGR2GRAY». «COLOR_BGR2GRAY» є константою в бібліотеці OpenCV, яка використовується як параметр у функції cvtColor. Вона позначає тип перетворення кольорового простору із BGR (Blue, Green, Red – синій, зелений, червоний) до відтінків сірого (GRAY), виражається це, як:

$$Y = 0.299R + 0.587G + 0.114B, \quad (4.1)$$

де Y відповідає інтенсивності сірого кольору;

R, *G*, і *B* — інтенсивності червоного, зеленого, і синього каналів відповідно.

Коефіцієнти відображають важливість кожного кольору для сприйняття яскравості людським оком [14]. Результат процесу градації сірого наведено на рисунку 4.3.

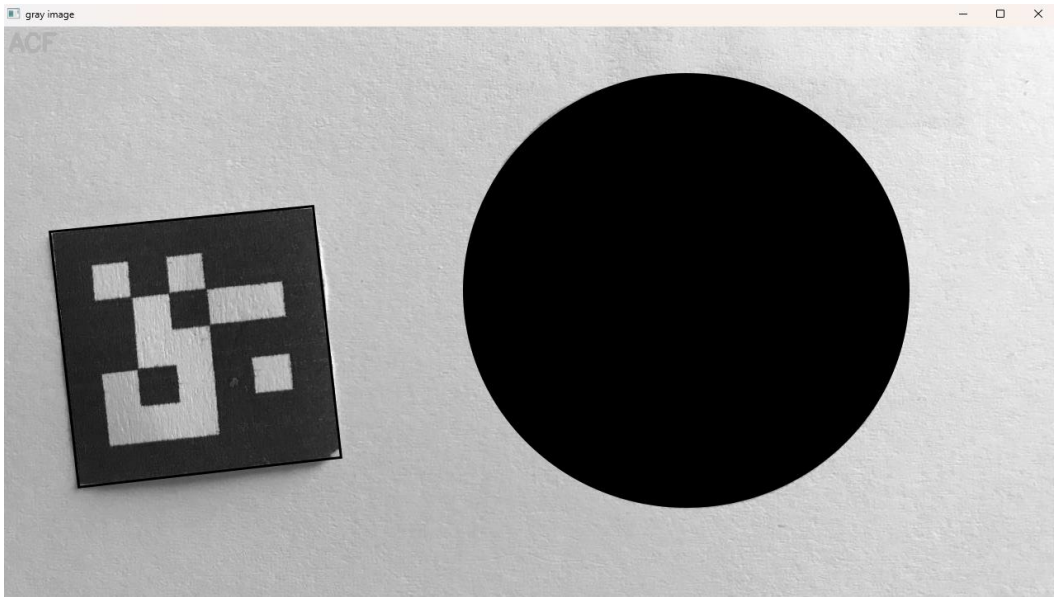

Рисунок 4.3 – зображення з використанням перетворення в градації сірого.

4.3 Процес розмиття

Функція «cv2.GaussianBlur» в OpenCV застосовує гаусівське розмиття до зображення. Гаусівське розмиття — це метод згладжування зображень, що зменшує шум та деталі. Вона приймає зображення, у моєму випадку це змінна gray, яка містить закодоване зображення у відтінках сірого. Розмір ядра ksize визначає висоту та ширину гаусівського фільтра. Більший розмір ядра призводить до сильнішого розмиття. Значення sigmaX та sigmaY визначають стандартні відхилення ядра по осях X та Y. Якщо sigmaY не вказано, воно приймається рівним sigmaX. Якщо обидва вказані як нулі, вони обчислюються з

розміру ядра. Гаусівське ядро застосовується до кожного пікселя зображення. Для кожного пікселя обчислюється взважена сума його сусідів, де ваги визначені гаусівським ядром. Операція згортки (конволюції) з ядром застосовується до зображення, що призводить до ефекту розмиття.

Гаусівське розмиття використовує гаусівську функцію (ядро) для генерування розмитих зображень. Гаусівське ядро формується за допомогою наступної двовимірної гаусівської функції:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{4.2}$$

де:

x та y — відстані від центру ядра по горизонталі та вертикалі відповідно;

σ — стандартне відхилення розподілу, яке контролює рівень розмиття.

Ядро нормалізується так, щоб сума всіх його елементів дорівнювала 1, забезпечуючи те, що загальна яскравість зображення не змінюється після розмиття [15]. Результат подано на рисунку 4.4.

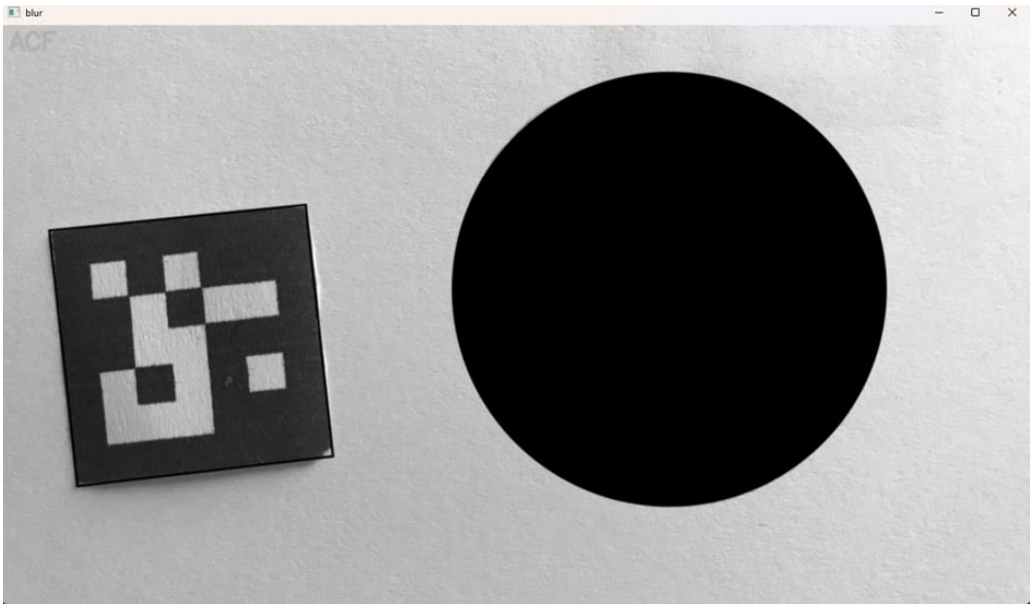

Рисунок 4.4 - Зображення з використанням розмиття.

4.4 Порогове перетворення

Функція cv2.threshold в OpenCV використовується для застосування порогової обробки до зображення. Порогова обробка – це простий, але ефективний метод сегментації зображень, який перетворює зображення в бінарний вигляд.

Обернена бінарна порогова обробка (cv2.THRESH_BINARY_INV). Якщо значення пікселя менше порогового значення, йому присвоюється maxVal, інакше – 0.

Параметри 0 та 255 вказують на колір візуального опису бінарного вигляду зображення, де 0 – це чорний, 255 – білий.

Процес порогової обробки можна розбити на наступні пункти:

1. Вибір порогового значення (threshold value) T.
2. Вибір максимального значення maxVal, яке буде присвоєно пікселям, що перевищують поріг.
3. Проходження через всі пікселі зображення та їх порівняння з пороговим значенням T.
4. Пікселі, які перевищують поріг, отримують значення maxVal, інші пікселі стають нулями (або залишаються незмінними залежно від типу порогової обробки).

Порогову обробку можна описати наступною функцією:

$$\begin{cases} maxVal & \text{якщо } x > T \\ 0 & \text{інакше'} \end{cases} \quad (4.3)$$

де *x* це значення пікселя зображення;

T – порогове значення, та maxVal – значення, що присвоюється пікселям, які перевищують поріг [16]. Порогова обробка тестового зображення подана на рисунку 4.5.

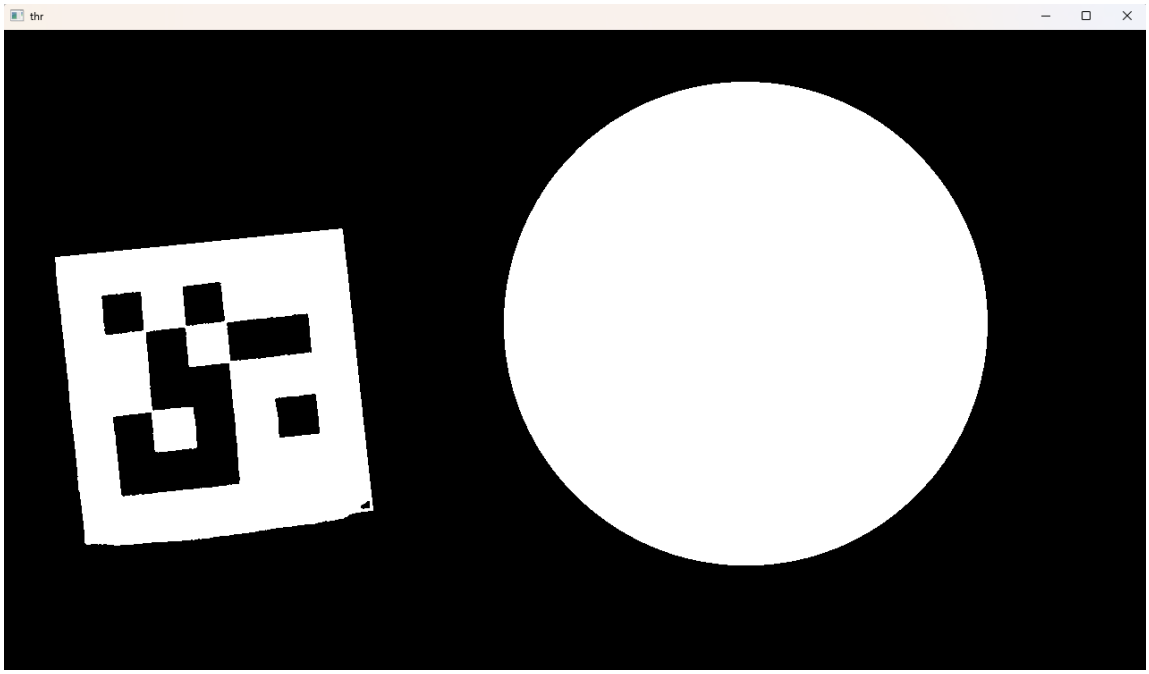

Рисунок 4.5 – результат порогової обробки

4.5 Створення структурного єлементу

Функція cv2.getStructuringElement в OpenCV використовується для створення структурного елементу, який потім може бути використаний в морфологічних операціях, таких як ерозія, диляція, відкриття, закриття тощо. Структурний елемент є ядром, яке визначає форму та розмір впливу морфологічних операцій на зображення.

Коли викликається cv2.getStructuringElement з cv2.MORPH_ELLIPSE та розміром (5,5), OpenCV створює еліптичний (або овальний) структурний елемент 5x5 пікселів. Це означає, що структурний елемент матиме форму еліпсу, вписаного в прямокутник 5x5 пікселів.

Структурний елемент у формі еліпса є корисним, коли необхідно застосувати більш «м'яку» морфологічну операцію, яка не вводить різких кутів або прямих ліній, як це може бути в структурному елементі прямокутної форми. Еліптичні структурні елементи часто використовуються для обробки зображень з круглими об'єктами або для зменшення артефактів, які можуть виникнути при використанні прямокутних або квадратних структурних елементів.

Еліпс у математичному сенсі може бути визначений за допомогою рівняння, що відображає пікселі, які попадають в межі еліпса. Для еліпса, вписаного в прямокутник, центр еліпса знаходиться у точці,

$$(\frac{width-1}{2}, \frac{height-1}{2}) \tag{4.4}$$

відповідають розмірам прямокутника.

Структурний елемент з формою еліпса визначається як множина точок (*x*,*y*), які задовольняють наступне рівняння:

$$\left(\frac{x-c_x}{r_x}\right)^2 + \left(\frac{y-c_y}{r_y}\right)^2 \leq 1, \tag{4.5}$$

Де *cx*,*cy* — координати центру еліпса;

rx,*ry* — радіуси еліпса по осям x та y (в цьому випадку буде $\frac{width}{2}$ та $\frac{height-1}{2}$ відповідно).

Для структурного елементу розміром 5×5, центр буде (2,2), і радіуси будуть $\frac{5}{2}$ по обох осях, оскільки розмір ядра нечітний, і центр еліпса точно в центрі ядра. Таким чином, було отримано структурний єлемент, який має вигляд, що подано на рисунку 4.6.

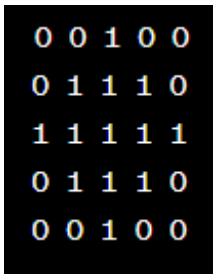

Рисунок 4.6 – Структурний елемент форми еліпса.

Тут центральний рядок повністю заповнений одиницями, що відображає найширшу частину еліпса, а верхній та нижній рядки мають по три одиниці, що відображає звуження еліпса ближче до його полюсів.

В результаті, структурний елемент буде мати форму матриці з бінарними значеннями, де 1 вказує на належність до еліпса, а 0 — на відсутність належності. Коли цей структурний елемент застосовується до зображення в рамках морфологічної операції, лише пікселі зі значенням 1 впливають на вихідну операцію.

4.6 Проведення морфологічних операцій

Функція **cv2.morphologyEx** в OpenCV застосовує морфологічні операції до зображень. Морфологічні операції — це прості підходи, які базуються на формі, і вони використовуються в основному для обробки бінарних зображень.

Морфологічні операції використовують структурний елемент (часто у формі невеликого прямокутника або диска), який «проводиться» по зображенню (подібно до згортки) і виконує операції, засновані на порівнянні сусідів пікселів.

В даному випадку функція «cv2.morphologyEx» використовується для операції відкриття (MORPH_OPEN) на зображенні thresh з використанням ядра kernel, ми робимо дві ітерації.

Програмно, функція морфологічної операції, приймає наступні змінні:

1. Thresh – Вхідне зображення, яке зазвичай є бінарним зображенням, отриманим після застосування порогової функції (cv2.threshold()), це було зроблено у формулі tresh.
2. cv2.MORPH_OPEN – Операція морфологічного відкриття, яка є послідовністю ерозії, за якою слідує диляція.
3. Kernel – Структурний елемент, який визначає форму операції. Це може бути, наприклад, квадрат або коло, або еліпс. У випадку даної робити – єліпс, отриманий у формулах № (kernel).
4. iterations=2 – Кількість ітерацій, які буде застосовано операцію відкриття. Більше ітерацій посилюють ефект морфологічного відкриття.

Процес починається з ерозії. Ерозія (Erosion) – Видаляє границі об’єктів. Теоретично, вона «стискає» передній план, зменшуючи розміри об’єктів.

$$A \ominus B = z \in E \mid Bz \subseteq A, \tag{4.6}$$

де A — вхідне зображення,

B — структурний елемент, а z — елемент вхідного зображення.

Після чого виконується Диляція (Dilation). Вона збільшує розмір об’єктів і заповнює діри і тріщини в передньому плані і виражена наступним рівнянням:

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\}, \tag{4.7}$$

де A — вхідне зображення після ерозії;

B — структурний елемент;

B^ — віддзеркалення структурного елемента B.

Відкриття (Opening) – Ерозія, за якою слід диляція. Використовується для видалення шуму:

$$A \circ B = (A \ominus B) \oplus B, \tag{4.8}$$

де A — вхідне зображення після диляції;

B — структурний елемент .

Коли вказано iterations=2, операція відкриття виконується двічі. Це означає, що вся послідовність (ерозія, за якою слідує диляція) буде застосована двічі. Це може бути особливо корисним для видалення більшого шуму або для відкриття. Загалом, морфологічне відкриття ефективно видаляє шум (малі об’єкти або переривчасті частини) з переднього плану об’єкта розділення) об’єктів, які є близькими один до одного на зображенні [17].

4.7 Пошук контурів

Функціонально, пошук контурів у OpenCV має форму, подану на рисунку 4.7, та приймає декілька змінних, а саме:

1. Вхідне зображення;
2. Режим витягування контуру;
3. Метод наближення контуру.

```
cnts = cv2.findContours(opening, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

Рисунок 4.7 Функція пошуку контурів

Нехай I – вхідне зображення, яке представлене як матриця. Попередня обробка включає функцію виявлення країв E, таку як Детектор країв Кенні. Вихід цього кроку, P, може бути визначено як:

$$P = E(I, \sigma l, \sigma h), \tag{4.9}$$

де E – це значення порогу;

σl – нижнє порогове значення для процедури гістерезису в Canny;

σh – верхнє порогове значення.

Функціональна форма детектора країв Кенні подана на рисунку 4.8:

```
edges = cv2.Canny(blurred, threshold1=50, threshold2=150)
```

Рисунок 4.8 Функціональна форма детектора країв Кенні

де blurred(I) - вхідне зображення, яке може бути одноканальним зображенням у відтінках сірого або похідними по x (dx) та y (dy);

Threshold1(σl) - нижній поріг для процесу гістерезису – це мінімальна інтенсивність, яка вважається досить сильною, щоб бути визначеною як край. Якщо зміни яскравості пікселів нижчі за цей поріг, вони відкидаються;

Threshold2(σh) - верхній поріг для процесу гістерезису. Визначає мінімальну інтенсивність для пікселів, які можуть бути визнані як край у випадку, якщо вони з’єднуються з сильними краями (що пройшли поріг threshold1).

Правильний вибір цих порогів визначає ефективність алгоритму детектування країв. Зазвичай, експериментують з цими значеннями для досягнення найкращих результатів відповідно до конкретного зображення та завдання. Низькі пороги можуть виводити багато шуму, тоді як високі пороги можуть пропускати деякі краї.

Градієнт зображення обчислюється для виявлення областей з високою похідною інтенсивності, що відповідають краям. Використовуючи оператори Собеля, обчислюються приблизні значення градієнтів:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \cdot I, \tag{4.10}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \cdot I, \tag{4.11}$$

де I — це зображення у відтінках сірого, а * позначає операцію згортки.

Величина та напрямок градієнта:

$$G = \sqrt{G_x^2 + G_y^2}, \tag{4.12}$$

$$\theta = \text{atan}\left(\frac{G_y}{Gx}\right), \tag{4.13}$$

де *Gx* та *Gy* — градієнти в горизонтальному та вертикальному напрямках відповідно [13].

Немаксимальне придушення:

$$G_{non\ max}(x,y) = \begin{cases} G(x,y) \text{ якщо } G(x,y) & \text{є локальним максимумом} \\ 0 & \text{інакше} \end{cases}. \tag{4.14}$$

На цьому етапі визначаються «справжні» краї за допомогою двох порогів:

- Сильні краї: $G_{non\ max}(x,y) > threshold2$; (4.15)
- Слабкі краї: $threshold1 < G_{non\ max}(x,y) \leq threshold2$; (4.16)
- Країв немає: $G_{non\ max}(x,y) \leq threshold1$. (4.17)

Прогове перетворення з гістерезисом:

$$G_{non\ max}(x,y) = \begin{cases} 255 & \text{для слабких країв} \\ 0 & \text{для слабких, які не з'єднаніз сильними,} \\ \text{Не визначено} & \text{для слабких, що чекають перевірки} \end{cases} \tag{4.18}$$

де mode – це ціле число, яке визначає режим витягування контуру. Це впливає на структуру та взаємозв'язок контурів.

У якості параметра Mode, буде вказано «cv2.RETR_EXTERNAL», Цей параметр визначає, які контури будуть знайдені та як вони будуть організовані у відповідь. Конкретно cv2.RETR_EXTERNAL вказує функції findContours знаходити тільки зовнішні контури, тобто ті, які не містяться всередині інших контурів. Нехай M позначає режим виявлення контурів, який може впливати на ієрархію контурів. Вибір режиму може бути представлений як функція, яка модифікує набір усіх можливих контурів C на основі M:

$$C' = F(C, M), \tag{4.19}$$

де F представляє функцію фільтрації режиму, а ′C′ - набір контурів після застосування фільтра режиму;

M – це ціле число, яке визначає метод наближення контуру. Це визначає, як апроксимується крива контуру. «cv2.CHAIN_APPROX_SIMPLE» (параметр method у findContours): метод апроксимації. Це означає, що контур буде апроксимовано лініями між сусідніми точками.

Нехай A позначає алгоритм апроксимації контуру, який зменшує кількість точок у контурі. Це може бути представлено як функція G, яка бере контур c ∈ C′ та метод m, що генерує апроксимований контур c′:

$$c' = G(c, m), \tag{4.20}$$

де G представляє функцію апроксимації;

m – метод апроксимації;

c′ - апроксимований контур.

Результатом є список, що містить N елементів, де кожний елемент Ci представляє контур i у вигляді послідовності точок. Кожний контур Ci складається з Mi точок. Contours може містити один або кілька контурів, залежно від параметрів пошуку.

$$Contours = \{ C_1, C_2 \dots C_N \}, \quad (4.21)$$

$$C_i = \{ p_{i1}, p_{i2}, \dots p_i, M_i \}, \quad (4.22)$$

де $P_{ij} = (\text{x}, \text{y})$є координатами $j -$ тої точки у контурі C_i.

Кожна точка *pi,j* в контурі *Ci* відповідає пікселю з координатами (*x,y*) у вихідному зображенні. Ці точки утворюють форму або границю об'єкта, яка може бути використана для аналізу та обробки. *N* – кількість контурів, що були знайдені на зображенні. Кожний контур *Ci* може містити різну кількість точок, що представляють границю об'єкта.

Крім цього, утворюється ієрархія кіл, вона ж hierarchy – це матриця або вектор, що містить інформацію про ієрархічні зв'язки між контурами на зображенні.

Ієрархія визначає, як контури пов'язані між собою: які з них є дитячими, батьківськими, а також наступними та попередніми на даному рівні ієрархії hierarchy використовується для аналізу структури об'єктів на зображенні, визначення та інтерпретації взаємозв'язків між контурами. Ця інформація допомагає в розумінні та визначенні ієрархічних відносин між контурами об'єктів на зображенні. Функціонально вона має наступний вигляд:

$$hierarchy[i] = (next, prev, first_{child}, parent), \quad (4.23)$$

де *next* – це індекс наступного контуру на цьому рівні ієрархії. Якщо немає наступного контуру, *next* дорівнює -1;

Prev – це індекс попереднього контуру на цьому рівні ієрархії. Якщо немає попереднього контуру, *prev* дорівнює -1;

First_child – це індекс першого «дитячого» контуру. Якщо немає дитячого контуру, *first_child* дорівнює -1;

Parent – це індекс «батьківського» контуру. Якщо немає батьківського контуру, *parent* дорівнює -1.

Таким чином, функція findCountours повертає кортеж *(contours,hierarchy*):

$$\Phi(I, \theta, \sigma l, \sigma h, M, m) = (S, H). \quad (4.24)$$

Функція генерує два основних вихідних результати: послідовність контурів S та ієрархію H. Кожен контур c’ у S може бути визначений як послідовність точок $\{p_1, p_2, \ldots, p_n\}$, а ієрархія H може бути представлена у вигляді матриці або деревовидної структури, яка описує взаємозв’язки між контурами [18].

Таким чином, вихідна інформація буде подана як зображено на рисунку 4.9:

```
[[376, 223]]], dtype=int32), array([[[814,  58]],
```

Рисунок 4.9 Результат розрахунку контуру

Де перший масив – контури, а другий – ієрархія H

4.8 Отримання розмірів

Функціонально, функція отримання розмірів кола подана на рисунку 4.10, що повертає радіус кола, а також координати його центру.

```
(x, y), r = cv2.minEnclosingCircle(c)
r = round(r, 3)
```

Рисунок 4.10 – Програмний код отримання розмірів кола контуру c

Функція minEnclosingCircle в OpenCV призначена для знаходження найменшого кола, яке може повністю охопити заданий набір точок. Це корисно для визначення найменшого можливого кола для групи точок, які можуть бути контуром об’єкта на зображенні [18].

Завдання визначення найменшого охоплюючого кола (Minimum Enclosing Circle, MEC) для набору точок є класичною задачею геометричної оптимізації.

Нехай маємо множину точок:

$$P = \{p_1, p_2, \dots, p_n\}, \tag{4.25}$$

де кожна точка *pi* має координати (*xi*,*yi*). Потрібно знайти центр $C=(x_c,y_c)$ та радіус *r* кола, так що кожна точка *pi* лежить всередині або на межі цього кола.

Це можна виразити через наступну задачу мінімізації:

$$min_{C,r} r \text{ під умовою } \| pi - C \| \leq r, \forall pi \in P, \tag{4.26}$$

де $\|pi - C\|$ є Евклідовою відстанню між точкою *pi* та центром кола *C*;

r – радіус кола.

OpenCV імплементує ефективний метод для знаходження MEC, але деталі внутрішньої реалізації не відкриті. Зазвичай такі алгоритми використовують ітераційні методи або методи найменших квадратів для наближення ідеального розв’язку. Існують відомі алгоритми, такі як алгоритм Велцля.

Алгоритм Велцля (Welzl’s algorithm) — це алгоритм для розв’язання задачі пошуку найменшого охоплюючого кола для заданої множини точок у двовимірному просторі.

Алгоритм Велцля є рекурсивним, ймовірнісним, та має очікувану часову складність O(n), де n – кількість точок. Він використовує концепцію

рекурсивного зворотнього виклику, що зменшує простір пошуку на кожному кроці, виключаючи точку, яка не є частиною MEC.

Для будь-якої точки *pi*, відстань до центра кола *C* задається як:

$$d(P_iC) = \sqrt{(x_i x_c)^2 + (y_i y_c)^2}. \quad (4.27)$$

Радіус r найменшого охоплюючого кола повинен задовольняти умову: $r = max_{pi \in P} d(Pi, C)$ [19].

Коло *C* та радіус *r* повинні задовольняти нерівності:

$$(x_i - x_c)^2 + (y_i - y_c)^2 \leq r^2, \forall p_i \in P. \quad (4.28)$$

Отримавши дані про коло, а саме його радіус, можна визначити розмір у сантиметрах.

4.9 Калібрування системи

Aruco маркери використовуються для позиціонування та орієнтування у 3D просторі, особливо в області доповненої реальності та робототехніки. Вони забезпечують швидке та надійне виявлення кутів та ідентифікацію. Для вирішення питання з калібруванням системи, було використано Aruco Marker 5x5, де 5x5 означає його програмний розмір у сантиметрах.

Для подальших розрахунків, потрібно знайти периметр маркера, це можливо за допомогою функції cv2.arcLength, яка визначає довжину контура або кривої в зображенні. В контексті виразу cv2.arcLength(corners[0], True), цей вираз використовується для обчислення довжини контура, який визначений точками маркера ArUco (представлених як corners[0]),

де corners[0] – координати вершин контура (маркера ArUco) на зображенні;

True – параметр, що вказує, що контур є замкненим (зацикленим).

Ця функція використовується для обчислення довжини контуру та корисно для визначення відношення кількості пікселів до фізичної величини [20].

Тоді:

$$PixelRatio = \frac{aruco\ perimeter}{20}, \tag{4.29}$$

де aruco_perimeter – периметр маркера ArUco (в пікселях).

Таким чином:

$$D(cm) = \frac{r}{PixelRatio} \div 2.1, \tag{4.30}$$

де r – радіус об’єкта на зображенні (в пікселях), отриманий у функції «minEnclosingCircle»;

pixel_cm_ratiopixel_cm_ratio – коефіцієнт, що визначає відношення кількості пікселів до сантиметрів;

2.1 – значення яке досягнуте експериментальним шляхом, потрібне через відхилення фізичного розміру маркера Aruco так як, програмно розмір маркера 5х5 см, а фізично він в 2.1 рази менший чим потрібно.

Таким чином, розроблений в рамках даного розділу алгоритм ідентифікації і вимірювання головки поршня оптичним методом складається з декількох основних кроків:

1. Попередня обробка – зображення піддається корекції перед тим, як система почне роботу, розмиття за гаусом для більш гладких границь об’єктів, перетворення в градації сірого, та останнім важливим кроком є перетворення вхідного зображення у бінарний вигляд;

2. Пошук країв – за допомогою Алгоритму Кенні (Canny) здійснюється пошук країв об’єктів. Цей алгоритм вважається одним з найефективніших для виявлення країв і широко використовується в галузі комп’ютерного зору для визначення лінійних країв на зображеннях;

3. Пошук контурів – знаходження контури на зображенні, використовуючи функцію findCountours;

4. Обчислення розміру контуру – обчислення розміру знайдених контурів за допомогою алгоритму Велцля;

5. Фільтрація – після виявлення контуру проводиться фільтрація знайдених контурів на основі різних критеріїв, таких як площа, форма або розмір, з метою виокремлення контуру цільового об’єкту;

6. Калібрування – для перетворення результатів в зручні для людини одиниці, відбувається калібрування системи за допомогою спеціальних Aruco маркерів.

5 АНАЛІЗ МЕТРОЛОГІЧНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ КОНТРОЛЮ РОЗМІРІВ ПОРШНІВ ДВИГУНІВ ВНУТРІШНЬОГО ЗГОРЯННЯ

5.1 Теорія невизначеності

Невизначеність вимірювань визначає характеристику недостовірності вимірювань на міжнародному рівні. Слово «невизначеність» походить від англійського «Uncertainty». Це поняття відображає відсутність точного знання (істинного) значення вимірюваної величини Y та висловлює сумнів в тому, наскільки точно результат вимірювання представляє Y. Згідно з визначенням, невизначеність – це параметр, пов’язаний із результатом вимірювань, що характеризує розкид значень, які можна обґрунтовано пов’язати із вимірюваною величиною Y [21].

У визначенні цього параметра використовується перша літера слова «uncertainty» (невизначеність), що наочно демонструється стандартним форматом представлення вимірювальних результатів.

$$У = y \pm U,\ p = 0{,}95 \qquad (5.1)$$

З даного виразу очевидно, що можливий розкид значень Y знаходиться в межах діапазону ± U відносно результату вимірювання, як показано на рисунку 5.1. Ступінь впевненості в тому, що значення Y знаходиться в цьому інтервалі, визначається ймовірністю (рівнем довіри) p = 0,95.

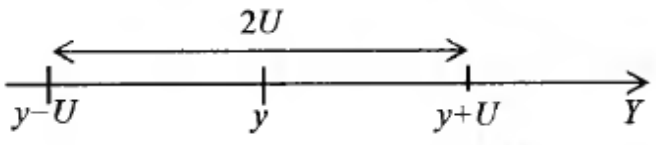

Рисунок 5.1 –Визначення невизначеності вимірювання

Всі фактори невизначеності вхідних величин можна розподілити на дві категорії відповідно до методів їх оцінювання:

1. Категорія А – фактори, що оцінюються за допомогою статистичних методів, які включають обробку результатів многоразових вимірювань;
2. Категорія В – фактори, що оцінюються іншими методами, такими як характеристики, взяті з паспортів приладів, методики вимірювань, дані з попередніх експериментів, відомості з довідників і т. д [21].

5.2 Розрахунок невизначеності системи оптичного контролю за типом А

Для перевірки точності роботи системи проведемо оцінку невизначеності вимірювань за типом А. Для отримання вихідних даних для розрахунку невизначеності використовувалася додаткова функція в програмі, яка автоматично заповнювала масив даних та зберігала їх у файлі для зручності взаємодії. Вхідні дані представлені у вигляді таблиці 5.1 і містять 30 результатів вимірювань, проведених для об’єкта розміром 2 см. Враховуючи особливості системи, були здійснені розрахунки невизначеності за типом А.

Таблиця 5.1 – Вхідні дані для розрахунку

№ Екс.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
R, см	2,10	2,07	2,03	2,03	2,03	2,03	2,03	2,04	2,03	2,02	2,03	2,03	2,03	2,03	2,05
№ Екс.	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
R, см	2,03	2,03	2,02	2,03	2,03	2,03	2,03	2,03	2,03	2,03	2,05	2,03	2,03	2,04	2,03

Для отримання результатів спостережень було використане програмне забезпечення, що представляє собою модифіковану версію основної програми для отримання вимірювальних результатів об’єктів. Основною відмінністю є додана функція, яка повертає та записує до файлу певну кількість спостережень.

Після цього був проведений ряд вимірювань, результати яких були представлені у таблиці 5.1. Для розрахунку невизначеності визначимо середнє арифметичне значення радіусів з усіх проведених вимірювань.

$$\bar{R} = \frac{1}{n}\sum_{i=1}^{n} R_i, \tag{5.2}$$

$$\begin{aligned}\bar{R} = \frac{1}{30}(&2.10 + 2.07 + 2.03 + 2.03 + 2.03 + 2.03 + 2.0 + 2.04 + 2.03\\ &+ 2.02 + 2.03 + 2.03 + 2.03 + 2.03 + 2.05 + 2.03 + 2.03\\ &+ 2.02 + 2.03 + 2.03 + 2.03 + 2.03 + 2.03 + 2.03 + 2.03\\ &+ 2.05 + 2.03 + 2.03 + 2.04 + 2.03) = 2.035 \text{ см}.\end{aligned}$$

Середнє арифметичне значення радіусів об’єкта складає 2.035 см.

Для визначення невизначеності за типом А використовуємо наступну формулу:

$$U_A(R) = \sqrt{\frac{\sum_{i=1}^{n}(E_i - E)^2}{n(n-1)}}. \tag{5.3}$$

Тоді, невизначеність вимірювань радіусу за типом А дорівнюватиме:

$$U_A(R) = \sqrt{\frac{\begin{array}{c}-0{,}0650^2 + (-0{,}0350^2) + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 +\\ +(-0{,}0050^2) + 0{,}0050^2 + 0{,}0150^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 +\\ (-0{,}0150)^2 + 0{,}0050_2 + 0{,}0050^2 + 0{,}0150^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 \quad +\\ +0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 + 0{,}0050^2 + (-0{,}0150)^2 + 0{,}0050^2 + 0{,}0050^2 +\\ +(-0{,}0050)^2 + 0{,}0050^2\end{array}}{30\,(30-1)}}$$

$$=0{,}00052 \text{ см}. \tag{5.4}$$

Отже, невизначеність вимірювань у системі оптичного вимірювання складає 0,00052 см.

5.3 Розрахунок невизначеності системи оптичного контролю за типом В

Для оцінки невизначеності величини Xi, яка не була отримана шляхом повторних спостережень, використовують оцінену дисперсію $u^2(Xi)$ або стандартну невизначеність u(Xi). Ці оцінки базуються на науковому розсуді, що ґрунтується на всій доступній інформації про можливу змінність Xi. Стандартну невизначеність типу В визначають з передбачуваної функції щільності вірогідності, основаної на рівні впевненості в тому, що подія обов’язково відбудеться [21].

Вимірювання проводилися у приміщенні з температурою повітря t = 30 °C та освітленістю стенду Ev=505 лк.. Абсолютна похибка радіуса кола становить ΔR = 0,00052 см.

Отже, загальна невизначеність, обумовлена основною похибкою вимірювань, визначається за формулою:

$$U_B(R) = \frac{\Delta R}{2 \cdot \sqrt{3}}, \tag{5.5}$$

$$U_B(R) = \frac{0{,}00052}{2 \cdot \sqrt{3}} = 0{,}0001 \text{ см}. \tag{5.6}$$

При цьому закон розподілу їхніх ймовірностей, якщо він невідомий, приймають рівномірним.

Вплив температури на зміну фокусної відстані лінзи може виявити значний вплив на оптичну систему та точність вимірювань. Зміни в фокусній відстані можуть впливати на роботу оптичної системи, включаючи її здатність розрізняти деталі та вимірювати розміри об’єктів. Для цього розрахуємо додаткову невизначеність, зумовлену зміною фокусної відстані під впливом зміни температури. Оскільки вимірювання проводилися при температурі t = 30 °C, а

температура за нормальних умов t = 30 °C , додаткова невизначеність виникає таким чином:

$$U_B(\Delta f \Delta t) = \frac{R_f - R}{R_p}, \quad (5.7)$$

Де R_f – Виміряне значення радіусу головки поршня після зміни фокусної відстані під впливом зміни температури;

$\bar{R}$ – Середнє арифметичне виміряних значень;

R_p – Радіус головки поршня.

$$R_f = R_p \cdot \left(\frac{f}{f + \Delta f}\right), \quad (5.8)$$

Де R_p – Радіус головки поршня;

f – фокусна відстань;

Δf – відхилення фокусної відстані під впливом температури.

Для розрахунку впливу температури на зміну фокусної відстані лінзи Δf, використана наступна формула:

$$\Delta f = f_0 \cdot \alpha \cdot \Delta t, \quad (5.9)$$

Де f_0 – початкова фокусна відстань лінзи при температурі за нормальних умов t – $20°C$;

α – коефіцієнт термічного розширення матеріалу лінзи;

Δt – Зміна температури.

Таким чином, $f_0 = 12$ см, при температурі $20°C$, у якості матеріалу лінзи, в веб-камері A4Tech PK-910H використовується акриловий пластик, коефіцієнт

термічного розширення якого становить $\alpha = 7 \cdot 10^{-5}$, а зміна температури $\Delta T = 10°C$.

Тоді:

$$\Delta f = 12 \cdot (7 \cdot 10^{-5}) \cdot 10 \approx 0{,}00084 \text{ см.} \quad (5.10)$$

Отже, при зміні температури на 10 градусів по Цельсія, фокусна відстань лінзи може змінитися приблизно на 0,00084 см.

Таким чином, підставляючи отримані значення, було розраховано R_f:

$$R_f = 2 \cdot \left(\frac{12}{12+0{,}00084}\right) \approx 1{,}99987 \text{ см.} \quad (5.11)$$

Тоді, додаткова невизначеність зумовлена впливом температури на зміну фокусної відстані $U_B(\Delta f \Delta t)$ дорівнює:

$$U_B(\Delta f \Delta t) = \frac{|\, 1{,}99987 - 2{,}035 \,|}{2} = 0{,}01756 \text{ см.} \quad (5.12)$$

Вплив освітлення на якість оптичних вимірювань може бути значущим у зв'язку з рядом оптичних та фізичних ефектів. Наприклад, матеріали об'єктів різним чином відбивають світло, що може впливати на точність вимірювань, особливо при використанні оптичних систем для визначення геометричних розмірів. Високий коефіцієнт відбиття матеріалу може підсилювати вплив освітлення.

Також важливими факторами є тіні, що можуть виникати внаслідок нерівномірного освітлення. Такі тіні можуть ускладнити розпізнавання контурів та впливати на точність вимірювань. Крім того, необхідно належним чином налаштовувати та калібрувати оптичну систему для врахування характеристик освітлення та зменшення систематичних помилок.

Для розрахунку додаткової невизначеності, пов'язаної з впливом освітлення на вимірювання, використано метод регресії. Зазвичай

використовують метод найменших квадратів для побудови лінійної регресійної моделі. У нашому випадку ми маємо дані освітлення (незалежна змінна) і вимірювання (залежна змінна), які подані у таблиці 5.2 як результат експерименту.

Таблиця 5.2 Результати вимірювання за різного рівня освітлення

№ вим.	**Рівень освітлення (лк)**	**Результат вимірювання**
1	400	2.0102
2	410	1.9990
3	420	2.0353
4	430	1.9952
5	440	2.0208
6	450	2.0104
7	460	2.0333
8	470	2.0691
9	480	1.9850
10	490	2.0003
11	500	2.0278
12	510	2.0890
13	520	2.0251
14	530	1.9864
15	540	2.0158
16	550	2.0311
17	560	1.9638
18	570	1.9630
19	580	1.9702
20	590	2.0163

Був взятий рівень освітлення в діапазоні від 400 до 590 лк, з кроком 10 лк, який можна досягти за допомогою LED ламп у домашніх умовах, рівень освітлення яких регулюється за допомогою пульта. Вимірювання рівня освітлення робочої поверхні здійснювався за допомогою люксметра BENETECH GM1010.

У випадку невідомого закону розподілу їх ймовірностей припускають рівномірний розподіл, тоді:

$$U_{\text{В}}(\Delta E_v) = \frac{10\% * \Delta E_v}{2 \cdot \sqrt{3}}. \quad (5.13)$$

Застосуємо лінійну регресійну модель:

$$\Delta \text{E}_v = a \cdot E + b, \quad (5.14)$$

Де a – коефіцієнт нахилу;

E – Освітлення;

b– вільний член регресії.

Коефіцієнт нахилу (a) в лінійній регресійній моделі вказує на зміну залежної змінної (вимірювань) від незалежної змінної (освітлення). Досягнутий коефіцієнт нахилу визначає нахил лінії регресії та показує, наскільки змінюється середнє значення вимірювань при зміні одиниці незалежної змінної.

Коефіцієнт нахилу розраховується за наступною формулою:

$$a = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2} = 0{,}0025 \text{ см/лк}, \quad (5.15)$$

де n – кількість спостережень;

x – значення незалежної змінної (освітлення);

y – значення залежної змінної (вимірювання);

Метод найменших квадратів дозволяє знайти b за формую:

$$b = \frac{(\sum y) - a(\sum x)}{n} = 1{,}98 \text{ см}, \quad (5.16)$$

Де n – кількість спостережень;

a - коефіцієнт нахилу;

x – значення незалежної змінної (освітлення);

y – значення залежної змінної (вимірювання).

Розрахуємо середню абсолютну похибку *E* під впливом зміни освітлення:

$$E = \frac{1}{n}\sum_{i=1}^{20} E_i^2 = 1{,}29 \text{ см.} \tag{5.17}$$

Таким чином, ΔE_v дорівнює:

$$\Delta E_v = 0{,}0025 \cdot 1{,}29 + 1{,}98 = 1{,}983 \text{ см.} \tag{5.18}$$

Звідси, додаткова невизначеність зумовлена впливом освітлення складає:

$$U_B(\Delta E_v) = \frac{10\% * 1{,}983}{2 \cdot \sqrt{3}} = 0{,}0573 \text{ см.} \tag{5.19}$$

Стандартну невизначеність типу В розраховується за формулою:

$$U_B(R) = \sqrt{U_B(\Delta f \Delta t)^2 + U_B(\Delta E_v)^2}, \tag{5.20}$$

$$U_B(R) = \sqrt{0{,}01756^2 + 0{,}0573^2} = 0{,}0599 \text{ см.}$$

Сумарну стандартну невизначеність розраховують за формулою:

$$U_c(R) = \sqrt{U_A^2(R) + U_B^2(R)}, \tag{5.21}$$

$$U_c(R) = \sqrt{0{,}0001^{\,2} + 0{,}0599^2} \approx 0{,}0599 \text{ см.}$$

Розширена невизначеність вимірювань для довірчого рівня 80% складає:

$$U(R) = U_c(R) \cdot k, \tag{5.22}$$

де k – коефіцієнт охоплення і становить 1,3 для 30 вимірювань, звідси:

$$U(R) = 0{,}0599 \cdot 1{,}3 = 0{,}07. \quad (5.23)$$

Результат вимірювання записується у вигляді:

$$R = R_{\text{в}} \pm U_R \text{ см}, \qquad P = 0{,}80, \quad (5.24)$$

$$R = 2{,}035 \pm 0{,}070 \text{ см}, \qquad P = 0{,}80.$$

Також, розраховані результати невизначеності представлені у документі МР.МТТм-19.00.00.002. Бюджет невизначеності, який зазвичай подається у вигляді таблиці, є конденсованим висновком з розрахунків невизначеності типу В. В цій таблиці включені всі чинники, що впливають на невизначеність, для полегшення подальшого аналізу. Бюджет невизначеності для оптичної системи контролю розмірів поршнів двигуна внутрішнього згорання представлено у таблиці 1 документа МР.МТТм-19.00.00.002.

ВИСНОВКИ

Отже, в магістерській роботі досліджено методи та засоби оптичного контролю об’єктів круглої та еліптичної форми. Вибрано оптимальні алгоритми для побудови власного алгоритму оптичного контролю, та на основі якого, в подальшому розроблена власна система оптичного контролю деталей круглої та еліптичної форми, що дає змогу проводити неруйнівний контроль з відповідною точністю. У якості об’єкта дослідження, вибраний поршень двигуна внутрішнього згорання, а саме його головку. Програма ґрунтується на використанні програмного коду, написаного високорівневою мовою програмування Python. Програмне забезпечення створене на основі відкритої бібліотеки OpenCV для мови програмування Python.

Розроблений в рамках магістерської роботи алгоритм ідентифікації і вимірювання головки поршня оптичним методом складається з декількох основних кроків:

1. Попередня обробка зображення;
2. Пошук країв за допомогою алгоритму Кенні (Canny);
3. Пошук контурів;
4. Обчислення розміру контуру за допомогою алгоритму Велцля;
5. Фільтрація для виявлення контурів;
6. Калібрування за допомогою спеціальних Aruco маркерів.

Алгоритм повертає результати вимірювання у сантиметрах, проте у будь-який момент одиниці вимірювання можна змінити на міліметри.

Розроблена система дозволяє вимірювати діаметри об’єктів, використовуючи камеру та програмні алгоритми. Система може працювати як у режимі контролю об’єктів, так і у режимі звичайної лінійки для вимірювання діаметрів об’єктів у кадрі. Зручність використання системи підкреслюється її взаємодією з базою даних, що дозволяє зберігати інформацію про об’єкти

вимірювання та уникнути повторного введення даних. Програма обладнана зрозумілим інтерфейсом, що дозволяє користуватися нею без спеціалізованих знань. Система працює у автоматичному режимі, не вимагаючи присутності оператора, але здатна виводити необхідну інформацію на екран для оцінки процесу оператором, якщо це необхідно. Під час контролю в кадрі повинен знаходитися лише об’єкт контролю, але у режимі лінійки система може вимірювати декілька об’єктів одночасно.

У першому розділі магістерської роботи був проведений аналіз об’єкта контролю – поршнів двигуна внутрішнього згорання. Досліджено їхню будову, функціональне призначення у двигуні, а також висвітлено проблеми, які можуть виникнути при невідповідності розмірів головки поршня. У рамках аналізу було детально розглянуто засоби та методи контролю розмірів поршнів двигуна внутрішнього згорання.

У другому розділі був проведений аналіз існуючих алгоритмів ідентифікації об’єктів на зображенні. Проаналізовано їх слабкі та сильні сторони.

У третьому розділі було розроблено та описано структуру схему оптичної системи контролю об’єктів круглої та еліптичної форми. Обґрунтовано вибір мови програмування. Розроблено та описано алгоритм роботи програмного забезпечення. На основі алгоритму було розроблено власне програмне забезпечення для контролю радіусу деталей. Програмне забезпечення отримало графічний інтерфейс для зручного користування, взаємодіє з базою даних, має можливість контролювати розмір об’єкта вимірювання та робити висновки щодо його придатності для використання або необхідності його бракування.

У четвертому розділі був розроблений алгоритм ідентифікації і вимірювання деталей круглої чи еліптичної форми з його подальшим математичним описом. Алгоритм є основною складовою оптичної системи контролю, даючи можливість отримувати розміри круглих чи еліптичних

об’єктів у реальному часі у звичних для нас одиницях – сантиметрах або міліметрах.

У п’ятому розділі був проведений метрологічний аналіз розробленої системи, в ході якого використовувалися практичні знання для розрахунку невизначеності типу А та В. Невизначеність типу А отримана на основі експерименту який базується на 30 вимірюваннях та складає $U_A(R) = 0{,}00052$ см. Невизначеність типу Б було отримано на основі розрахунку впливу зміни освітлення та зміни фокусної відстані лінзи під впливом різниці температури від нормальних умов, таким чином $R = 2{,}035 \pm 0{,}070$ см, $P = 0{,}80$.

Таким чином, практичне значення отриманих результатів у магістерській роботі:

- Проведено аналіз методів та засобів для контролю геометричних параметрів головки поршня ДВЗ, де були визначені їх переваги та недоліки.
- Спроектовано алгоритм ідентифікації і вимірювання деталей круглої чи еліптичної форм та проведено математичний опис, використаного у програмному забезпеченні для оптичного контролю.
- На основі отриманих висновків розроблений алгоритм для оптичного контролю розмірів деталей циліндричної форми, а також відповідне програмне забезпечення для його втілення.
- Завершуючи дослідження, був проведений метрологічний аналіз системи оптичного контролю розмірів деталей циліндричної форми, використовуючи практичні знання для розрахунку невизначеності типу А та В.

ПЕРЕЛІК ПОСИЛАНЬ НА ДЖЕРЕЛА

1. Мороз Ю. В, Біліщук В. Б. Контроль розмірів головки поршня оптичним методом: тези, Всеукраїнська Інтернет-конференція молодих учених і студентів, ІТОТП-2023 «Інформаційні технології в освіті, техніці та промисловості». 12 жовтня. (м. Івано-Франківськ, 2023 р.) С 98-100.
2. Урок: Призначення, конструкція,матеріал поршнів,поршневих кілець,поршневих пальців та шатунів. Вимоги до них. URL: https://vseosvita.ua/lesson/pryznachennia-konstruktsiiamaterial-porshnivporshnevykh-kiletsporshnevykh-paltsiv-ta-shatuniv-vymohy-do-nykh-57579.html (дата звернення 06.08.2023)
3. ПОРШНЕКОМПЛЕКТ 98.48ММ ДВИГУНА PERKINS 6.354. URL: https://agroviktor.com/ua/p1375224183-porshnekomplekt-9848mm-dvigatelya.html? (дата звернення 13.08.2023)
4. Alessandro Anzalone, Ph.D. Pneumatic Measurement. URL: https://etshare.pbworks.com/f/Chapter%2011%20Pneumatic%20Measurement.pdf (дата звернення 15.08.2023)
5. Сусліков Л.М., Студеняк І.П. Неруйнівні методи контролю: Навчальний посібник. – Ужгород: Видавництво УжНУ, 2016. – 192 с.
6. Білинський Й. Й, Животівський С. М. Огляд методів 3d-контролю геометричних розмірів деталей / Й. Й. Білинський, С. М. Животівський. – 2022. – 114 – 122 с. Вінницький національний технічний університет. ISSN 1997-9266.
7. OpenCV. Open Source Computer Vision. Cascade Classification Haar Feature-based Cascade Classifier for Object Detection.URL:https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html (дата звернення 26.08.2023)
8. HOG (Histogram of Oriented Gradients): An Overview. Mathematics, Paper Explanation and Code from scratch for beginners. URL:

https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f (дата звернення 29.08.2023)

9. OpenCV. Open Source Computer Vision. Template Matching. URL: https://docs.opencv.org/3.4/de/da9/tutorial_template_matching.html (дата звернення 03.09.2023)
10. OpenCV. Open Source Computer Vision. K-Means Clustering in OpenCV. URL: https://docs.opencv.org/4.x/d1/d5c/tutorial_py_kmeans_opencv.html (дата звернення 06.09.2023)
11. Raspberry p I 3b+ compute module 3. URL: https://docs.rs-online.com/a608/A700000007750677.pdf (дата звернення 08.09.2023)
12. Qt Designer Manual. URL: https://doc.qt.io/qt-6/qtdesigner-manual.html (дата звернення 14.09.2023)
13. OpenCV. Open Source Computer Vision. Canny Edge Detection. URL: https://docs.opencv.org/3.4/d7/de1/tutorial_js_canny.html (дата звернення 03.10.2023)
14. OpenCV. Open Source Computer Vision. Color Space Conversions. URL: https://docs.opencv.org/3.4/d8/d01/group__imgproc__color__conversions.html (дата звернення 07.10.2023)
15. Image Filters: Gaussian Blur. URL: https://aryamansharda.medium.com/image-filters-gaussian-blur-eb36db6781b1 (дата звернення 13.10.2023)
16. OpenCV Thresholding (cv2.threshold). URL: https://pyimagesearch.com/2021/04/28/opencv-thresholding-cv2-threshold/ (дата звернення 18.10.2023)
17. OpenCV. Open Source Computer Vision. Morphological Transformations. URL: https://docs.opencv.org/4.x/d4/d76/tutorial_js_morphological_ops.html (дата звернення 24.10.2023)
18. OpenCV. Open Source Computer Vision. Structural Analysis and Shape Descriptors. URL: https://docs.opencv.org/3.4/d3/dc0/group__imgproc__shape.html#ga17ed9f5d79ae97bd4c7cf18403e1689a (дата звернення 31.10.2023)

19. Collision, Randomization and Welzl’s Algorithm. URL: https://blog.karthickshiva.dev/2023/05/06/welzl-s-algorithm/ (дата звернення 10.11.2023)
20. OpenCV. Open Source Computer Vision. ArUco Marker Detection. URL: https://docs.opencv.org/3.4/d9/d6a/group__aruco.html (дата звернення 15.11.2023)
21. Василевський, О. М.В19 Основи теорії невизначеності вимірювань: навчальний посібник/ О. М. Василевський, В. Ю. Кучерук. – Вінниця : ВНТУ, 2012. – 172 с. ISBN 978-966-641-454-3.