# АВТОМАТИЗАЦІЯ ТЕХНОЛОГІЧНИХ ПРОЦЕСІВ

УДК 004.93

**КОВАЛЮК Д. О.[1]*, КОВАЛЮК О. О.[2], МАЛІШЕВСЬКИЙ В. С.[1]**
**[1]Національний технічний університет України**
**«Київський політехнічний інститут імені Ігоря Сікорського»**
**[2]Вінницький національний технічний університет**

# РЕАЛІЗАЦІЯ СИСТЕМ КОНТРОЛЮ ЯКОСТІ ПРОДУКЦІЇ НА ОСНОВІ МАШИННОГО ЗОРУ ТА WEB-ТЕХНОЛОГІЙ

*Технології комп'ютерного зору отримали широке застосування в різних галузях, зокрема в медицині, виробництві, транспорті, логістиці, сільському господарстві. В статті наведено загальну схему контролю якості продукції на основі комп'ютерного зору. Як приклад розглянуто задачу виявлення дефектів друкованих плат та проаналізовано стек технологій для її розв'язання. Розпізнавання дефектів базується на індексі структурної подібності SSIM, що обчислює різницю між зображеннями еталонної і поточної плати. Незважаючи на значний прогрес у галузі машинного зору, залишається багато невирішених питань щодо якості та розмірності вхідних даних, вибору моделей, точності результатів. Інтеграція елементів систем машинного зору для обміну інформацією та прийняття рішень також становить значну проблему, тому у статті представлено один із підходів до її розв'язання. Запропоновано програмну реалізацію для виявлення дефектів у вигляді веб-додатку. Архітектура системи складається з клієнтської та серверної частини. Клієнтська частина реалізована з використанням Vue.js, на стороні сервера використано Flask фреймворк, що полегшує інтеграцію з існуючими Python-бібліотеками.*

***Ключові слова:*** *контроль якості, система керування, комп'ютерний зір, машинний зір, web-технології, ресурсоефективність*

**DOI: 10.20535/2617-9741.1.2024.300980**

*Corresponding author: dmytro.kovalyuk@gmail.com
Received 20 February 2024; Accepted 01 March 2024

**Постановка проблеми.** Розвиток технологій комп'ютерного зору (Computer Vision – CV) в останнє десятиліття знайшов широке впровадження в різних галузях: медицині, виробництві, транспорті, логістиці, аграрному секторі, хімічній промисловості [1–3]. Аналіз досліджень [4–5] показує, що Computer Vision та його практична реалізація в промисловості – машинний зір (Machine Vision) широко використовуються для розв'язання задач виявлення дефектів продукції чи обладнання. Такий підхід повністю відповідає концепції сталого розвитку, оскільки дозволяє підвищити ресурсоефективність виробництв, зменшуючи кількість задіяного персоналу і одночасно підвищуючи точність детектування. Незважаючи на різні предметні області, загальну схему контролю якості продукції на основі комп'ютерного зору (рис. 1) можна представити наступним чином:

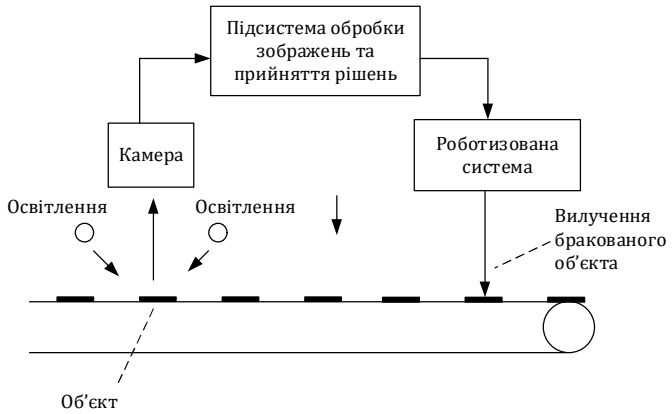


**Рис. 1 – Схема контролю якості продукції на основі комп'ютерного зору**

Хоча в останні десятиліття відбувається суттєве збільшення досліджень в даній галузі, проте існує багато невирішених проблем пов'язаних з повнотою і розмірністю вхідних даних про об'єкт, вибору моделей розпізнавання зображень та їх навчання, точність результатів [6]. Однією з важливих проблем є також інтеграція елементів системи машинного зору для обміну інформацією та здійснення висновків. В роботі продемонстрований один з підходів для розв'язання цієї проблеми.

**Аналіз попередніх досліджень.** Процес побудови систем контролю якістю продукції може бути представлений наступним чином: отримання інформації з камер, попередня обробка цієї інформації, розв'язання однієї або декількох наступних задач: семантична сегментація, інстанс-сегментація, детектуваня та супроводження об'єктів, оцінка розміщення (pose estimation). На кожному з цих етапів використовуються відповідні технології та програмні засоби для їх реалізації, наведені в таблиці 1.

**Таблиця 1 – Стек технологій для задач обробки зображень**

| Стадії | | | | |
|---|---|---|---|---|
| **Збір даних** | **Попередня обробка** | **Генерування / Маркування / Аугментація** | **Побудова моделі** | **Розгортання** |
| Отримання даних з пристрою | фільтрація шумів, зменшення розмірності, зменшення викидів | Анотація областей або точок у випадку ML / генерація з використанням CAD моделей | Навчання, тестування, висновок | Публікація моделей для подальшого промислового використання |
| Програмні засоби (Python, C++) | | | | |
| драйвери камер, SDK розробників камер | open3d<br>opencv<br>pandas | VoTT<br>CVAT<br>Blender<br>PyBullet | PyTorch<br>TensorFlow<br>OpenCV<br>scikit-learn | TensorFlow Serving,<br>PyTorch Serve<br>AWS SageMaker |

Розглянемо більш детально останній етап розгортання моделей. Стратегії і форми розгортання залежать від конкретної задачі, але їх можна умовно поділити на online та offline. Модель висновку створена з урахуванням периферійного (офлайн) розгортання передбачає, що розроблені моделі можна запускати на пристроях ARM (наприклад, Raspberry Pi), пристроях з підтримкою CUDA (таких як NVIDIA), пристроях архітектури x86 та інших. При онлайн розгортанні здійснюються виклики відповідного API (Application programming interface), які дозволяють взаємодіяти з розробленою моделлю.

Зазначимо, що при використанні API для відносно нескладних і не трудоємних задач, доцільно використати власне програмне забезпечення на основі REST/RPC, що дозволить повністю отримати контроль над процесом розрахунків і бути незалежним в його доопрацюванні/підтриманні.

**Метою** статті є дослідження підходу реалізації систем контролю якості продукції з використанням web-технологій. Оскільки найбільш популярними мовами програмування для розробки систем CV є Python та C (це пов'язано з наявністю відповідних wrapers–обгорток для виклику методів бібліотек обробки зображень), то в роботі буде розглянуто створення web-додатку з використанням Python-фреймворків.

**Математична модель розпізнавання дефектів друкованих плат.** Задача виявлення дефектів друкованих плат зустрічається на багатьох виробництвах. Для контролю плат використовується порівняння поточного зразка плати з еталонним, що не містить дефектів. Фактично це задача порівняння двох зображень і виявлення відмінностей між ними. Сьогодні є декілька підходів порівняння зображень [7], зокрема: основаних на гістограмах (Histogram-Based Approaches), індексах подібності (Structural Similarity Index), ознаках зображень (Feature-Based Approaches) та глибокому навчанні (Deep Learning-Based Approaches). Для побудови моделі розпізнавання дефектів плат використано фотографії виробів та базовий код з роботи [8]. Розпізнавання дефектів базується на використанні індексу SSIM (Structural Similarity Index). Він обчислює деградаційну компоненту, що відображає різницю між зображеннями, і схожу компоненту. Загальний SSIM індекс вираховується як добуток цих двох компонент.

Перед зіставленням зображень друкованих плат їх необхідно відформатувати: перевести у відтінки сірого та зменшити розмір. Зменшення розміру має декілька причини:

- *обчислювальна ефективність*: зменшення розміру зображення зменшує обчислювальне навантаження, оскільки SSIM вимагає порівняння структури зображення на рівні пікселів.
- *зменшення шуму*: при зменшенні розміру зображення деяка втрата деталей може приховати шум або артефакти, що можуть вплинути на значення SSIM.
- *зниження впливу непомітних змін*: зменшення розміру зображення може допомогти знизити вплив незначних змін, що є невидимими для людини, але впливають на значення метрики SSIM.

З урахуванням вищесказаного наведемо фрагмент коду, який виконує ці перетворення [8].

```
imageA = cv2.resize(imageA,(1000,500),interpolation=cv2.INTER_LINEAR)
imageB = cv2.resize(imageB,(1000,500),interpolation=cv2.INTER_LINEAR)
grayA = cv2.cvtColor(imageA, cv2.COLOR_BGR2GRAY)
grayB = cv2.cvtColor(imageB, cv2.COLOR_BGR2GRAY)
```

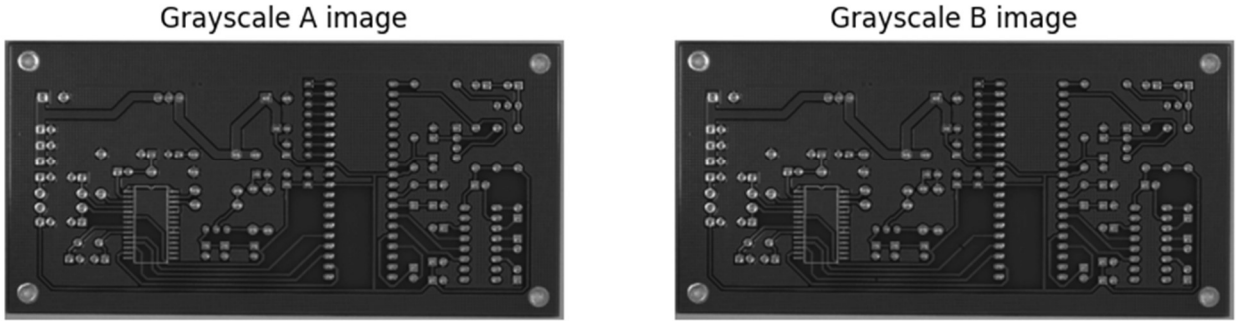


**Рис. 2 – Результат перетворення вхідних зображень у відтінки сірого**

Метрика SIIM уже реалізована в бібліотеках обробки зображень. В роботі використано підхід для обчислення SIIM на основі бібліотек OpenCV та scikit-image [9]. Фактично потрібно викликати вбудований метод і подати йому на вхід зображення. Отримавши SSIM зображення, виконаємо його порогову обробку.

```
(ssim, ssim_img) = structural_similarity(grayA, grayB, full=True)
thresh = cv2.threshold(ssim_img, 0, 255, cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
```

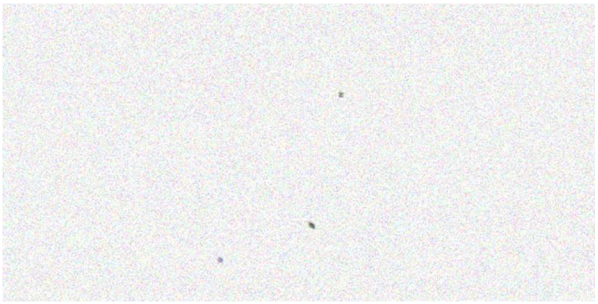

**Рис. 3 – Результуюче SSIM зображення після зіставлення**

**Рис. 4 – Результуюче зображення SSIM після порогової обробки**

Після отримання результату порогової обробки слід визначити координати об'єктів та обвести їх на вхідних зображеннях.

**Написання веб-додатку**

На першому етапі контролю якості камера здійснює знімок друкованої плати. Далі виконується порівняння отриманого зображення зі зразком за допомогою метрики SSIM. На основі результатів аналізу скрипт робить висновок, чи містить зображення друкованої плати дефекти, і визначає, чи потрібно бракувати плату. Для автоматизації процесу порівняння плат, вирішення задач розгортання та інтеграції на рівень SCADA створено веб-додаток, архітектура якого наведена на рис. 5. Цей веб-додаток дає змогу користувачеві вибрати еталонне зображення друкованої плати та завантажувати поточне зображення друкованої плати, яку система перевірятиме на наявність пошкоджень.

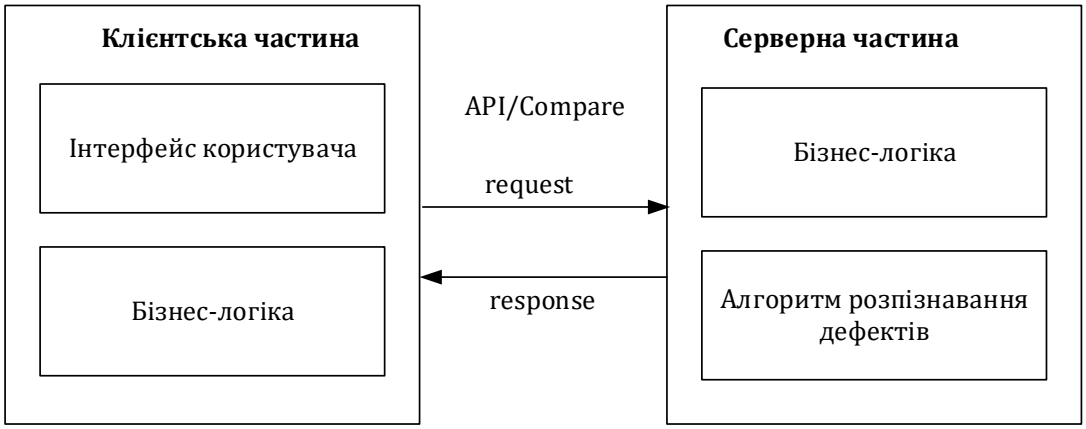


**Рис. 5 – Архітектура web-додатку для порівняння плат**

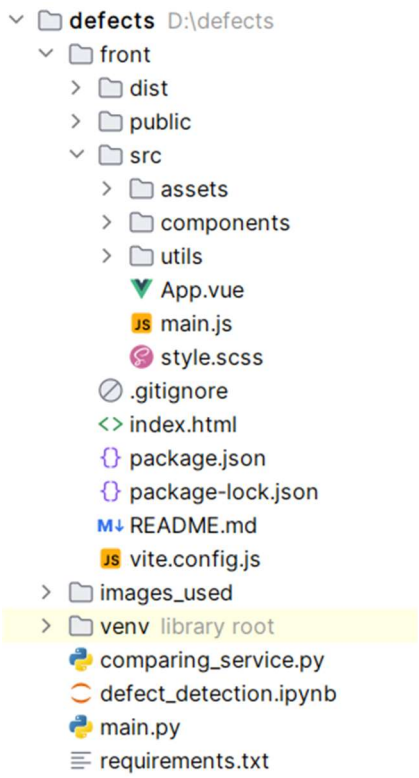


**Рис. 6 – Структура проекту**

Для реалізації клієнтської частини був обраний Vue.js, що є одним з найбільш використовуваним фреймворком JavaScript, спрямованим на розробку користувацьких інтерфейсів веб-додатків. Однією з переваг використання Vue.js є можливість побудови ефективних та динамічних односторінкових додатків (SPA). Завдяки реактивній системі Vue.js, зміни в стані даних автоматично відстежуються і відображення оновлюються відповідно.

Для реалізації серверної частини був використаний Flask – фреймворк написаний на мові програмування Python. Це полегшує інтеграцію з алгоритмом порівняння плат, оскільки обидва використовують Python. На рис. 6 наведено структуру проекту.

Оператор має можливість завантажити два зображення: зображення зразка друкованої плати і зображення друкованої плати, яке потрібно порівняти зі зразком. Після завантаження, зображення друкованих плат відображаються в інтерфейсі системи. Для початку процесу порівняння, оператор має натиснути кнопку "Compare". Після цього зображення друкованих плат перетворюються в формат base64 і надсилаються запитом на сервер.

На сервері отримані дані відповідно декодуються і порівнюються з використанням відповідних алгоритмів. Після здійснення порівняння, результати кодуються в формат base64 і надсилаються назад на клієнтську частину системи. Клієнтська частина декодує отримані дані та виводить зображення результатів порівняння в інтерфейсі системи, що дозволяє оператору оцінити подібність чи розбіжності між друкованою платою і зразком.

Додаток може працювати в автоматичному і ручному режимі. В першому – інформація має зчитуватися з камери і пересилатися на вказану url-адресу у формі POST запиту HTTP-протоколу.

В другому випадку фото завантажується оператором вручну. Нижче наведений код, що відповідає за порівняння двох зображень.

```
@app.route('/compare', methods=['POST'])
def compare_images():
    try:
        data = request.get_json()
        image_a = base64_to_cv2_image(data["sample_image"])
        image_b = base64_to_cv2_image(data["compared_image"])
        if image_a.shape[0] != image_b.shape[0] or
            image_a.shape[1] !=   image_b.shape[1]:
            raise ValueError('Images not same')
        result_image_a, result_image_b, image_defect_count, image_ssim =
             compare(image_a, image_b)
        response = {
            "image_a": cv2_image_to_base64(result_image_a),
            "image_b": cv2_image_to_base64(result_image_b),
            "image_defect_count": image_defect_count,
            "image_ssim": image_ssim,
        }
        return jsonify(response)
    except ValueError as e:
        return jsonify({"error": str(e)}), 400
```

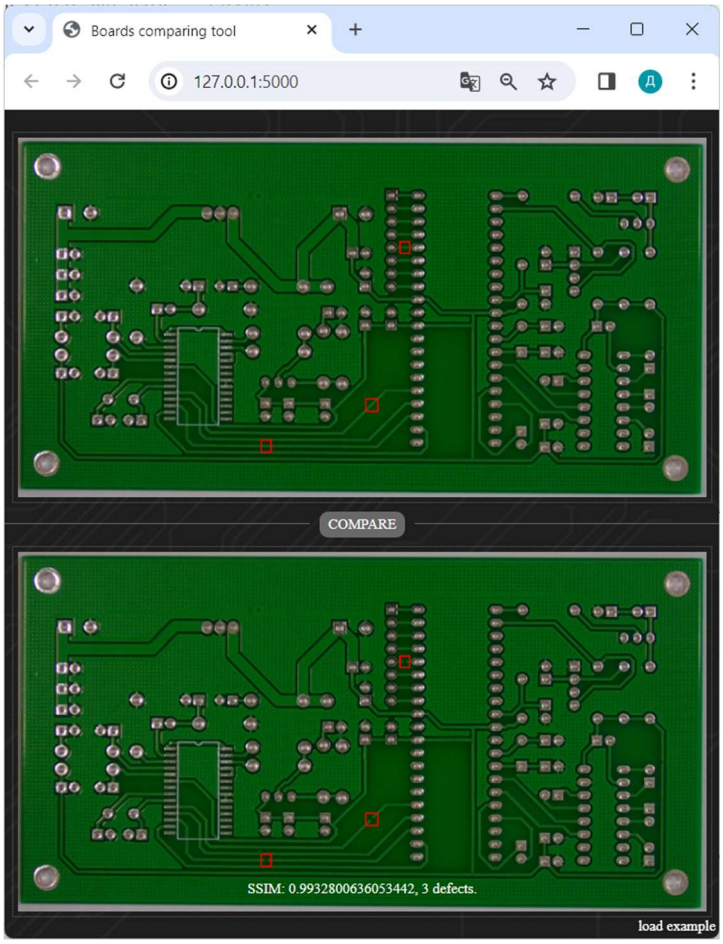


**Рис. 7 – Результат порівняння плат в web-додатку**

**Висновки.** Проведено аналіз проблем, що виникають при реалізації та розгортанні систем контролю якості продукції на основі методів комп'ютерного зору. Запропоновано використовувати web-технологій зокрема Python-фреймворків для інтеграції складових таких систем. Даний підхід продемонстровано на прикладі задачі розпізнавання дефектів друкованих плат.

**Перспективи подальших досліджень.** В подальшому доцільним є продовження досліджень для створення автоматичної системи контролю якості, в якій би зображення з камер автоматично передавалося в веб-додаток, результати порівняння відображалися на екрані оператора, виконувалася відповідна імітація робота-маніпулятора у випадку виявлення дефектів.

**Список використаної літератури**

1. Trinh C., Meimaroglou D., Hoppe S. Machine Learning in Chemical Product Engineering: The State of the Art and a Guide for Newcomers. Processes 2021, 9, 1456. URL: https://doi.org/10.3390/pr9081456 (date of access 24.02.2004)
2. Divyanshu Tirkey, Kshitiz Kumar Singh, Shrivishal Tripathi Performance analysis of AI-based solutions for crop disease identification, detection, and classification, Smart Agricultural Technology, Volume 5, 2023, 100238, ISSN 2772-3755. URL: https://doi.org/10.1016/j.atech.2023.100238 date of access 24.02.2004)
3. Aditya Akundi, Mark Reyna A Machine Vision Based Automated Quality Control System for Product Dimensional Analysis, Procedia Computer Science, Volume 185, 2021, Pages 127-134, ISSN 1877-0509. URL: https://doi.org/10.1016/j.procs.2021.05.014. (date of access 24.02.2004)
4. Yang, J., Li, S., Wang, Z., Dong, H., Wang, J., Tang, S. Using Deep Learning to Detect Defects in Manufacturing: A Comprehensive Survey and Current Challenges. Materials 2020, 13, 5755. URL: https://doi.org/10.3390/ma13245755 (date of access 24.02.2004)
5. Villalba-Diez, J., Schmidt, D., Gevers, R., Ordieres-Meré, J., Buchwitz, M., Wellbrock, W Deep Learning for Industrial Computer Vision Quality Control in the Printing Industry 4.0. Sensors 2019, 19, 3987. URL: https://doi.org/10.3390/s19183987 (date of access 24.02.2004)
6. Тимчишин Р.М., Волков О.Є., Господарчук О.Ю., Богачук Ю.П. Сучасні підходи до розв'язання задач комп'ютерного зору, Control systems and computers, 2018, № 6 Article 4. URL: https://doi.org/10.15407/usim.2018.06.046 (дата звернення 24.02.2004)
7. Vasista Reddy Exploring Image Similarity Approaches in Python. ScrapeHero, Sep 5, 2023. URL: https://medium.com/scrapehero/exploring-image-similarity-approaches-in-python-b8ca0a3ed5a3 (date of access 24.02.2004)
8. How to detect defects on a Printed Circuit Board using OpenCV? March 7, 2021. URL: https://blog.ldtalentwork.com/2021/03/07/how-to-detect-defects-on-a-printed-circuit-board-using-opencv (date of access 24.02.2004)
9. Adrian Rosebrock Image Difference with OpenCV and Python. June 19, 2017. URL: https://pyimagesearch.com/2017/06/19/image-difference-with-opencv-and-python/ (date of access 24.02.2004)

---

***Dmytro Kovaliuk, Oleh Kovaliuk, Volodymyr Malishevskyi***

**IMPLEMENTATION OF QUALITY CONTROL SYSTEMS BASED ON MACHINE VISION AND WEB TECHNOLOGIES**

*The development of Computer Vision technologies has seen widespread adoption in various fields over the last time, including medicine, manufacturing, transportation, logistics, agriculture, and chemical industry. Research analysis indicates that Computer Vision and its practical implementation, known as Machine Vision, are widely used for defect detection in production and equipment. Despite significant research progress in this field in recent years, many unresolved issues remain concerning the quality and dimensions of input data, selection of image recognition models and their training, and the accuracy of results. Integration of machine vision system elements for information exchange and decision-making poses a significant challenge. This paper presents one approach to addressing this issue.*

*The aim of the paper is to explore the implementation approach of quality control systems using web technologies. The task of detecting defects in printed circuit boards (PCBs) is common in many manufacturing processes. PCB inspection involves comparing the current PCB sample with a defect-free reference. Essentially, this task involves comparing two images and identifying differences between them. Defect recognition is based on the Structural*

*Similarity Index (SSIM), which computes the degradation component reflecting the difference between images and the similarity component. The SSIM metric is already implemented in image processing libraries. This work utilizes calculations using OpenCV and scikit-image image libraries. To automate the PCB comparison process and address deployment and integration challenges at the SCADA level, a web application has been developed. This web application allows users to select a reference PCB image and upload the current PCB image for inspection.*

*For the client-side implementation, Vue.js was chosen due to its popularity and efficiency in developing web interfaces, particularly Single Page Applications (SPAs). Vue.js's reactive system ensures automatic updates to reflect changes in data state. For the server-side implementation, Flask – a Python framework, was used, facilitating integration with the PCB comparison algorithm as both utilize Python.*

*According to the available features, operators can upload two images – a reference PCB image and the PCB image to be compared with the reference. After uploading, the images are displayed in the system interface. On the server side, the received data is decoded and compared using the appropriate algorithms. After comparison, the results are encoded in base64 format and sent back to the client-side for display, allowing the operator to assess the similarity or differences between the PCBs. The application can operate in both automatic and manual modes.*

*Therefore, an analysis of the challenges encountered in implementing and deploying quality control systems based on computer vision methods has been developed. The use of web technologies, particularly Python frameworks, for integrating developed models is proposed. This approach is demonstrated using the example of PCB defect recognition tasks.*

***Keywords**: quality control, control systems, computer vision, machine vision, web technologies, resource efficiency*

***References***

1. Trinh, C., Meimaroglou, D., & Hoppe, S. (2021). Machine Learning in Chemical Product Engineering: The State of the Art and a Guide for Newcomers. Processes, 9, 1456. [Online] Available: https://doi.org/10.3390/pr9081456 (Accessed: 24.02.2004).
2. Tirkey, D., Singh, K. K., & Tripathi, S. (2023). Performance analysis of AI-based solutions for crop disease identification, detection, and classification. Smart Agricultural Technology, 5, 100238. [Online] Available: https://doi.org/10.1016/j.atech.2023.100238 (Accessed: 24.02.2004).
3. Akundi, A., & Reyna, M. (2021). A Machine Vision Based Automated Quality Control System for Product Dimensional Analysis. Procedia Computer Science, 185, 127-134. [Online] Available: https://doi.org/10.1016/j.procs.2021.05.014 (Accessed: 24.02.2004).
4. Yang, J., Li, S., Wang, Z., Dong, H., Wang, J., & Tang, S. (2020). Using Deep Learning to Detect Defects in Manufacturing: A Comprehensive Survey and Current Challenges. Materials, 13, 5755. [Online] Available: https://doi.org/10.3390/ma13245755 (Accessed: 24.02.2004).
5. Villalba-Diez, J., Schmidt, D., Gevers, R., Ordieres-Meré, J., Buchwitz, M., & Wellbrock, W. (2019). Deep Learning for Industrial Computer Vision Quality Control in the Printing Industry 4.0. Sensors, 19, 3987. [Online] Available: https://doi.org/10.3390/s19183987 (Accessed: 24.02.2004).
6. Tymchyshyn, R.M., Volkov, O.Ye., Hospodarchuk, O.Yu., & Bohachuk, Yu.P. (2018). Suchasni pidkhody do rozvyazannia zadach kompiuternoho zoru. Control systems and computers, № 6 Article 4. [Online] Available: https://doi.org/10.15407/usim.2018.06.046 (data zvernenia 24.02.2004)
7. Reddy, V. (2023). Exploring Image Similarity Approaches in Python. ScrapeHero. [Online] Available: https://medium.com/scrapehero/exploring-image-similarity-approaches-in-python-b8ca0a3ed5a3 (Accessed: 24.02.2004).
8. (2021). How to detect defects on a Printed Circuit Board using OpenCV? [Online] Available: https://blog.ldtalentwork.com/2021/03/07/how-to-detect-defects-on-a-printed-circuit-board-using-opencv (Accessed: 24.02.2004).
9. Rosebrock, A. (2017). Image Difference with OpenCV and Python. [Online] Available: https://pyimagesearch.com/2017/06/19/image-difference-with-opencv-and-python/ (Accessed: 24.02.2004).