УДК: 620.179.16 DOI 10.36910/775.24153966.2023.75.12

В.С. Волкотруб, Л.О. Гуменюк
Луцький національний технічний університет

АВТОМАТИЗАЦІЯ ПРОЦЕСУ ВИЯВЛЕННЯ ТРІЩИН НА ОСНОВІ ОБРОБКИ ЗОБРАЖЕНЬ

Запропоновано алгоритм виявлення тріщин на зображеннях, який використовує різні методи для досягнення високої точності та ефективності. Алгоритм включає етапи попередньої обробки, такі як регулювання яскравості, підвищення контрастності та віднімання фону для зменшення шуму та підвищення контрастності зображення. Алгоритм застосовує фільтри Габора для виділення деталей на зображенні, а потім застосовує бінаризацію і виявлення країв для вилучення тріщин із зображення. Виконано його програмну реалізацію.

Ключові слова: *аналіз тріщин, комп'ютерний зір, класифікація елементів зображень, фільтр Габора, оброблення зображень, покращення зображень, Python.*

V. Volkotrub, L. Gumeniuk

AUTOMATION OF THE CRACK DETECTION PROCESS BASED ON IMAGE PROCESSING

An algorithm for detecting cracks in images is proposed that uses various methods to achieve high accuracy and efficiency. The algorithm includes preprocessing steps such as brightness adjustment, contrast enhancement, and background subtraction to reduce noise and enhance image contrast. The algorithm applies Gabor filters to extract details in the image, and then applies binarization and edge detection to extract cracks from the image. Its software implementation has been performed.

Keywords: *crack analysis, computer vision, image feature classification, Gabor filter, image processing, image enhancement, Python.*

Постановка проблеми. В сучасних умовах, де автоматизація процесів стає все більш актуальною, проблема виявлення тріщин у різних конструкціях і матеріалах є досить важливою. З одного боку, це дозволяє забезпечити безпеку та ефективність роботи різних машин, з іншого - знижує витрати на планове обслуговування та ремонт об'єктів.

Виявлення тріщин конструкцій є важливим у багатьох галузях, таких як цивільне будівництво, аерокосмічна промисловість, автомобілебудування та виробництво. У цивільному будівництві, наприклад, виявлення тріщин у мостах, будівлях та інших спорудах має вирішальне значення для забезпечення їхньої безпеки та довговічності. Автоматизація процесу виявлення тріщин за допомогою методів обробки зображень може підвищити швидкість і точність процесу, що призведе до раннього виявлення потенційних загроз безпеці.

В аерокосмічній та автомобільній промисловості виявлення тріщин в деталях літаків і автомобілів має важливе значення для забезпечення безпеки пасажирів і надійності транспортних засобів. Автоматизоване виявлення тріщин може допомогти виявити потенційні проблеми на ранніх стадіях, знизити ризик аварій і заощадити час і гроші.

На виробництві виявлення тріщин у виробах може допомогти покращити процес контролю якості та зменшити витрати на доопрацювання та ремонт. Автоматизоване виявлення тріщин може допомогти виявити дефекти в процесі виробництва і підвищити загальну ефективність і продуктивність виробничого процесу.

Використання обробки зображень для виявлення тріщин має кілька переваг.

По-перше, її можна використовувати для виявлення тріщин у важкодоступних або небезпечних для людини місцях. Наприклад, у випадку трубопроводів, які прокладені під землею або у важкодоступних місцях, може знадобитися використання роботів або дронів для зйомки зображень для аналізу.

По-друге, обробка зображень може бути використана для виявлення тріщин на ранній стадії, до того, як вони стануть більш серйозними і потенційно спричинить катастрофічні аварії. Це може допомогти запобігти дорогому ремонту або навіть катастрофам, які можуть виникнути через невиявлені тріщини.

По-третє, обробка зображень може забезпечити більш об'єктивний і послідовний підхід до виявлення тріщин. Навчені моделі машинного навчання можна використовувати для аналізу зображень з різних джерел і за різних умов, що може зменшити ризик людської помилки і варіативність результатів.

Актуальність автоматизації процесу виявлення тріщин полягає у тому, що пошук дефектів є важливим етапом в процесі діагностики машин і обладнання, а також забезпечує безпеку їх експлуатації. Неруйнівний контроль, яким є обробка зображень, є перспективною технологією, яка дозволяє збільшити точність виявлення тріщин, а також покращити швидкість і ефективність процесу.

Неавтоматизовані (ручні, візуальні) та автоматизовані методи виявлення тріщин на основі обробки зображень - це два різних підходи до виявлення та аналізу тріщин на зображеннях.

Неавтоматизовані методи передбачають, що людина-оператор вручну оглядає зображення і виявляє тріщини на основі їх візуального вигляду. Це може включати вимірювання розміру, форми і розташування тріщин, а також оцінку їхньої серйозності. Ручні методи схильні до людських помилок і можуть бути трудомісткими, особливо при роботі з великою кількістю зображень або складними зображеннями. Ці методи не передбачають використання будь-яких комп'ютеризованих алгоритмів обробки зображень.

Автоматизовані методи, з іншого боку, використовують алгоритми обробки зображень для автоматичного виявлення та аналізу тріщин на зображеннях. Ці методи можуть включати такі техніки, як виявлення країв, встановлення порогових значень і виділення ознак для виявлення і локалізації тріщин на зображенні. Автоматизовані методи також можуть включати алгоритми машинного навчання і глибокого навчання, які навчені розпізнавати і класифікувати тріщини на зображеннях. Автоматизовані методи, як правило, швидші, точніші і послідовніші, ніж ручні методи, і можуть обробляти велику кількість зображень без втручання людини.

Автоматизовані методи виявлення тріщин на основі обробки зображень мають великий потенціал порівняно з неавтоматизованими методами, оскільки вони забезпечують точнішу та швидшу ідентифікацію тріщин на зображеннях. Також важливим фактором є те, що автоматизовані методи виявлення тріщин можуть бути використані для обробки великої кількості зображень в автоматичному режимі.

Загалом, автоматизовані методи виявлення тріщин, засновані на обробці зображень, є більш ефективними, точними і надійними, ніж неавтоматизовані методи, але ступінь їх використання також залежить від складності зображення і типу тріщин.

Аналіз останніх досліджень і публікацій. Розробка методів дослідження та алгоритмів для автоматизації виявлення тріщин на зображеннях була постійним процесом, який розвивався протягом тривалого часу. Ранні методи передбачали ручний огляд і вимірювання тріщин, що займало багато часу і було схильне до людської помилки. З появою технології цифрових зображень почали з'являтися автоматизовані методи виявлення тріщин.

У 1990-х роках дослідники почали використовувати методи обробки зображень, такі як виявлення країв і встановлення порогових значень, для виявлення і локалізації тріщин на зображеннях [1-3]. Ці методи, хоч і були ефективними, але обмежувалися якістю зображень та умовами освітлення, за яких вони були зроблені.

У 2000-х роках для виявлення тріщин на зображеннях почали застосовувати алгоритми машинного навчання, такі як нейронні мережі та машини опорних векторів [4-6]. Ці методи показали підвищену точність порівняно з традиційними методами обробки зображень, але все ще мали обмеження в роботі з реальними зображеннями з варіаціями освітлення і текстур поверхні.

Останні дослідження були зосереджені на розробці алгоритмів глибокого навчання, таких як згорткові нейронні мережі (CNN), для виявлення тріщин на зображеннях [7-8]. Ці методи показали багатообіцяючі результати, з високою точністю і стійкістю до змін в освітленні і текстурі поверхні. Крім того, дослідники також використовували генеративні змагальні мережі (GAN) для створення синтетичних зображень тріщин для навчання ШНМ.

Загалом, розробка методів дослідження та алгоритмів для автоматизації виявлення тріщин на зображеннях була поступовим процесом, який включав інтеграцію різних методів з обробки зображень, машинного навчання та глибокого навчання.

В даний час розробка методів дослідження та алгоритмів для автоматизації виявлення тріщин на зображеннях зосереджена на підвищенні точності, надійності та ефективності процесу виявлення.

Одним з напрямків сучасних досліджень є використання алгоритмів глибокого навчання, таких як згорткові нейронні мережі (CNN), для виявлення тріщин на зображеннях [7-8]. Ці методи показали високу точність і стійкість до змін освітлення і текстури поверхні, і дослідники

продовжують покращувати їхню ефективність за допомогою таких методів, як навчання з перенесенням і ансамблеве навчання.

Іншою сферою сучасних досліджень є використання методів комп'ютерного зору для покращення виявлення тріщин на зображеннях [4, 6]. Ці методи включають використання 3D-зображень, стереозображень і структурованого світла для отримання більш детальних і точних зображень тріщин.

Дослідники також вивчають використання методів машинного навчання, таких як навчання з глибоким підкріпленням і активне навчання, щоб підвищити ефективність і швидкість процесу виявлення.

Крім того, існують методи, що базуються на техніці покращення зображень, такі як денойзинг зображень, надвисока роздільна здатність зображень та відновлення зображень, які використовуються для покращення якості зображень перед процесом виявлення тріщин, таким чином, тріщини можуть бути виявлені більш точно.

Таким чином, сучасні дослідження в галузі автоматизації виявлення тріщин на зображеннях зосереджені на підвищенні точності, надійності та ефективності процесу виявлення за рахунок використання методів глибокого навчання, комп'ютерного зору, машинного навчання та покращення якості зображень.

Загалом, сфера автоматизації виявлення тріщин на зображеннях є активною галуззю досліджень з багатьма потенційними напрямками для подальшого розвитку. З постійним розвитком технологій цілком ймовірно, що в майбутньому будуть розроблені нові та більш ефективні методи виявлення тріщин на зображеннях.

Постановка завдань. Метою роботи є розробка системи автоматизованого виявлення тріщин на основі обробки зображень.

Викладення основного матеріалу. Автоматизація виявлення тріщин на зображеннях може бути досягнута за допомогою методів обробки зображень, таких як встановлення порогових значень, виявлення країв і виділення ознак. Процес зазвичай включає попередню обробку зображення для покращення видимості тріщин, наприклад, видалення шуму і регулювання контрастності. Потім застосовується алгоритм для виявлення країв або особливостей, які відповідають тріщинам. Вони можуть бути передані в модель машинного навчання для класифікації. Використання моделей глибокого навчання, таких як згорткові нейронні мережі (CNN), показало свою ефективність у подібних задачах.

Ці методи можуть використовуватися в різних областях, таких як медична візуалізація, комп'ютерний зір, стиснення зображень і відео, робототехніка, біометрія та будівництво. Вони можуть поєднуватися з іншими технологіями, щоб зробити результати більш ефективними і точними.

Процес виявлення тріщин на основі обробки зображень включає в себе отримання зображень поверхні, що підлягає перевірці та їх аналіз за допомогою комп'ютерних алгоритмів.

Алгоритми шукають характерні ознаки, такі, як зміни текстури, кольору чи форми, які вказують на наявність тріщин. Результатом аналізу, як правило, є карта тріщин, виявлених на зображенні, яку можна використовувати для подальшої оцінки та ремонту.

Методи обробки зображень можуть використовуватися як для виявлення поверхневих, так і підповерхневих тріщин і можуть виконуватися за допомогою різних програмних засобів.

У обробці зображень для виявлення тріщин використовується кілька методів, зокрема:

– бінеаризація зображення - це процес перетворення зображення у відтінках сірого у бінарне зображення, де кожен піксель є або чорним, або білим [9].

– виявлення країв - цей метод передбачає ідентифікацію країв або меж на зображенні, які відповідають тріщинам. Краї можна виявити за допомогою різних алгоритмів, таких як детектор країв Канні, оператор Собеля або перетворення Хафа [10].

– сегментація зображення - цей метод передбачає поділ зображення на різні регіони та аналіз кожного регіону окремо. Зазвичай сегменти створюються на основі змін кольору або текстури зображення [1].

– зіставлення шаблону - цей метод передбачає порівняння зображення з шаблоном відомих шаблонів тріщин. Потім зображення обробляється для пошуку областей, які відповідають шаблону, які потім позначаються як потенційні місця тріщин

Розроблений алгоритм виконує обробку заданого зображення, використовуючи комбінацію наступних методів:

1. Корекція яскравості.
2. Адаптивне вирівнювання гістограми з обмеженим контрастом (CLAHE).
3. Віднімання фону за допомогою алгоритму кульки, що котиться.
4. Фільтр Габора.
5. Порогова корекція зображення.

Метою цих методів є покращення зображення та виділення тріщин на зображенні. Кінцевий результат обробки зображення отримується шляхом додавання декількох відфільтрованих зображень, створених шляхом застосування фільтра Габора до зображення з різними параметрами. Результат потім порогується для створення бінарного зображення.

Додатково є можливість отримати контур об'єктів на зображенні, для чого до напівтонового зображення застосовується фільтр Габора і віднімається від кінцевого результату. Скрипт також виводить на екран проміжний і кінцевий результати і зберігає їх у вигляді графічних файлів.

Корекція яскравості - це процес регулювання рівня яскравості зображення. Зазвичай це робиться для того, щоб покращити видимість зображення і зробити його візуально привабливішим. Корекцію яскравості зазвичай виконують за допомогою бібліотеки обробки зображень, наприклад, OpenCV, де такі функції, як cv2.addWeighted, можна використовувати для збільшення або зменшення загальної яскравості зображення шляхом коригування ваг оригінального зображення і нульового масиву з постійним скалярним значенням.

Адаптивне вирівнювання гістограми з обмеженим контрастом (CLAHE) - це метод комп'ютерного зору, який використовується для покращення локальної контрастності зображення. Він працює шляхом поділу зображення на невеликі плитки, обчислення гістограми значень інтенсивності в кожній плитці, а потім регулювання яскравості і контрастності в кожній плитці так, щоб гістограма була більш рівномірною. Це допомагає виявити особливості зображень, які важко побачити через варіації освітлення або тіні.

Частина назви "з обмеженням контрастності" означає, що алгоритм обмежує кількість підсилення контрастності, яку можна застосувати до кожної плитки, щоб запобігти надмірному посиленню шуму або інших дрібних деталей на зображенні.

Віднімання фону за допомогою алгоритму кульки, що котиться - це метод віднімання фону при обробці зображень. Він працює шляхом віднімання локально оціненого фону від зображення, де фон оцінюється за допомогою морфологічної операції (наприклад, розширення) зі структуроутворюючим елементом у формі кулі.

Радіус кулі визначає розмір ділянки фону, яка буде врахована для оцінки. Віднімання оціненого фону зазвичай виконується попіксельно і призначене для видалення великомасштабних варіацій інтенсивності із зображення та покращення дрібномасштабних особливостей.

Алгоритм широко використовується в різних додатках, таких як мікроскопія та комп'ютерний зір.

Фільтр Габора - це тип лінійного фільтра, який використовується для аналізу текстури в обробці зображень. Він може витягувати як просторову, так і частотну інформацію із зображення і може бути використаний для ідентифікації текстурних візерунків і країв.

Зазвичай фільтр являє собою гауссову функцію, модульовану синусоїдальною хвилею, і може бути налаштований для виявлення особливостей у певних масштабах і орієнтаціях.

Фільтри Габора широко використовуються в комп'ютерному зорі, розпізнаванні зображень і шаблонів, а також для вилучення ознак.

Порогова корекція зображення - це процес перетворення напівтонового або кольорового зображення у двійкове шляхом присвоєння кожному пікселю двійкового значення 0 або 1 на основі порогового значення. Це порогове значення використовується для розділення зображення на дві області, передній і задній план, де пікселі нижче порогового значення отримують значення 0 (чорний колір), а пікселі вище порогового значення отримують значення 1 (білий колір).

Цей метод широко використовується в обробці зображень для розпізнавання об'єктів, виявлення країв та інших застосувань.

Алгоритм покроково - код реалізує конвеєр для обробки зображення та виявлення шаблонів або структур на зображенні. Ось високорівневий опис кроків:

Попередня обробка вхідного зображення:

- перетворення зображення з BGR у відтінки сірого;
- збільшення яскравості зображення за допомогою функції cv2.addWeighted;

- застосування до зображення адаптивного гістограмного вирівнювання з обмеженим контрастом (CLAHE);
- віднімання фону за допомогою функції "subtract_background_rolling_ball" бібліотеки cv2_rolling_ball;
- повторне застосування CLAHE до зображення після віднімання фону.

Застосування банку фільтрів Габора до попередньо обробленого зображення:

- побудова фільтрів Габора із заданими параметрами за допомогою функції cv2.getGaborKernel;
- фільтрація попередньо обробленого зображення кожним з фільтрів Габора за допомогою функції cv2.filter2D;
- збереження максимального відгуку відфільтрованого зображення.

Обробка відфільтрованого зображення для створення бінарного зображення:

- використання cv2.threshold з аргументом "cv2.THRESH_BINARY + cv2.THRESH_OTSU" для перетворення відфільтрованого зображення у бінарне зображення.

Виявлення контурів на бінарному зображенні:

- якщо змінна contour має значення True, то до вихідного зображення застосовується медіанне розмиття і фільтрація Габора, а бінарний результат відкидається;
- потім, беручи побітовий результат кроку 2 і від'ємний результат кроку 4, отримуємо кінцевий результат з контурами.

Програмна реалізація виділення тріщин на зображенні:

1. Імпортуємо необхідні бібліотеки: numpy, cv2, cv2_rolling_ball.
2. Скористаємось функцію build_filters для створення фільтрів за допомогою GaborKernel із заданими параметрами: розмір ядра (ksize), стандартне відхилення (sigma), орієнтація (thet), довжина хвилі (lamb), співвідношення сторін (gam) та фазовий зсув (psi).
3. Застосовуємо функцію change_brightness для регулювання яскравості зображення за допомогою cv2.addWeighted.
4. Скористаємось функцією predv для виконання попередньої обробки зображення, яка включає наступні кроки:

-перетворення зображення у відтінки сірого;
-регулювання яскравість за допомогою функції change_brightness;
-застосовування адаптивного вирівнювання гістограми з обмеженим контрастом (CLAHE) для підвищення контрастності зображення;
-віднімання фону за допомогою функції cv2_rolling_ball.subtract_background_rolling_ball;
-повторне застосування CLAHE для підвищення контрастності зображення.

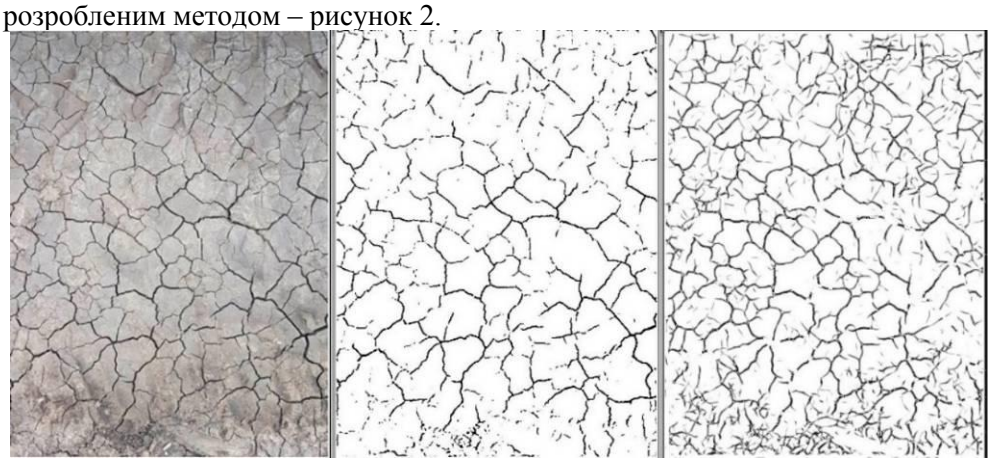
5. Визначаємо процес функції для застосування фільтрів до зображення за допомогою cv2.filter2D і повертаємо результат.
6. Скористаємось функцією result для застосування фільтрів до вхідного зображення за допомогою функцій build_filters та process і виведемо результат на екран, а також збережаємо його у файл.
7. Викликаємо функцію result1 для застосування фільтрів до попередньо обробленого зображення за допомогою функції process, а потім застосуємо порогове значення для отримання бінарного зображення. Підсумовуємо бінарні зображення для різних орієнтацій і застосовуємо побітові операції. Якщо параметр контур має значення True, застосовуємо той самий процес до вихідного зображення і віднімаємо результат від суми бінарних зображень.
8. Нарешті, викликаємо функцію result1 з відповідними параметрами на вхідному зображенні.
9. Виведемо на екран кінцевий результат і зберігаємо його у файл.
10. Повторюємо кроки 7-9 для різних значень thet, lamb, gam і psi, щоб отримати оптимальний результат.

Результати роботи програми виділення тріщин на зображенні наведені на рисунку 1.

а) б)

Рис. 1. **Результати роботи програми виділення тріщин на зображенні: а) оригінальне зображення; б) виділені тріщини.**

Для порівняння результатів приведемо зображення виділення тріщин методом фільтр Габора та розробленим методом – рисунок 2.



а) б) в)

Рис. 2. **Порівняння результатів: а) оригінальне зображення; б) виділення тріщин методом фільтра Габора; в) виділення тріщин розробленим алгоритмом.**

Як видно з рисунка 2, пропонований метод виявлення тріщин дає кращі результати. Виявлено не тільки базові (великі) тріщини, але і сітку малих тріщин.

Для цифрової оцінки ефективності пропонованого алгоритму, зображення, отримані методом Габора та пропонованим методом, перетворені в бінарні і пораховано процент чорних пікселів. Для зображення, отриманого методом Габора, цей процент склав 8.6% від загальної кількості пікселів на зображенні, для пропонованого - 11.7%. Таким чином, пропонований метод виявляє на 36% тріщин більше, ніж метод фільтра Габора.

Висновки. Таким чином, розроблений алгоритм є ефективним для виявлення тріщин на зображеннях завдяки використанню відповідних ядер Габора, кроків попередньої обробки, а також порогових і побітових операцій. Програмна реалізація дозволяє точно налаштувати процес виявлення, повторюючи певні кроки з різними значеннями параметрів. Загалом, цей метод забезпечує комплексний підхід до виявлення тріщин.

Список використаних джерел:

8. Rafael C. Gonzalez. Richard E. Woods. Digital Image. Processing. Pearson Education Limited 2018. 1019 pages.

9. Pitas, Ioannis. Digital Image Processing Algorithms and Applications. (2000). Published by Springer. 611 pages.

10. Milan Sonka, Vaclav Hlavac, Roger Boyle. Image Processing, Analysis, and Machine Vision. 3rd edition. CL Engineering, 2007. 872 pages.

11. Richard Szeliski. Computer Vision: Algorithms and Applications (Texts in Computer Science) 2nd ed. Springer. 2022. 947 pages.

12. Tom Mitchell. Machine Learning. McGraw-Hill Science/Engineering/Math. 1997. 432 pages.

13. Luo, Huafen & Xu, Jing & Binh, Nguyen & Liu, Shuntao & Zhang, Chi & Chen, Ken. (2014). A simple calibration procedure for structured light system. *Optics and Lasers in Engineering*. vol. 57. pp. 6–12. 10.1016/j.optlaseng.2014.01.010.

14. Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning (Adaptive Computation and Machine Learning series) Illustrated Edition. The MIT Press. 2016. 800 pages.

15. Ali Sharif Razavian. Convolutional Network Representation for Visual Recognition. Doctoral Thesis Stockholm, Sweden, 2017. 675 pages.

16. Seelaboyina, R., Vishwakarma, R. (2023). Different Thresholding Techniques in Image Processing : A Review. In: Kumar, A., Senatore, S., Gunjan, V.K. (eds) ICDSMLA 2021. *Lecture Notes in Electrical Engineering*, vol 947. Springer, Singapore.

17. "Algorithm and Technique on Various Edge Detection : A Survey" by Rashmi, Mukesh Kumar, Rohini Saxena. Signal & Image Processing : *An International Journal (SIPIJ)* Vol.4, No.3, June 2013. P. 65-75.