УДК 681.03

В.В. Туманов, А.Ю. Дорошенко

ВИКОРИСТАННЯ КОМП'ЮТЕРНОГО ЗОРУ В СИСТЕМІ ЦИФРОВОЇ НАРІЗКИ МАТЕРІАЛІВ

Запропонований підхід до реалізації системи комп'ютерного зору для розпізнавання та позиціонування об'єктів на різальній поверхні станків для цифрової нарізки матеріалів за допомогою фотознімку їх робочої поверхні разом з промаркованими об'єктами нарізки на ній. Розроблено алгоритми роботи модулів системи, які відповідають за калібрування камери і розпізнавання реєстраційних марок двома принципово різними способами, які доповнюють один одного. Розроблено також алгоритм ідентифікації об'єктів нарізки на основі застосування елементів теорії графів.

Ключові слова: система комп'ютерного зору, цифрова нарізка, OpenCV, калібрування камери, розпізнавання об'єктів.

Вступ

Роль систем комп'ютерного зору в деяких сферах промисловості дуже важко переоцінити. Робота з розпізнавання та ідентифікації об'єктів, що раніше покладалась виключно на людину, нині досить успішно виконується обчислювальною технікою [1]. При цьому, з роками зростає як швидкість виконання цієї роботи технікою, так і якість її результатів. Це пов'язано не тільки зі збільшенням потужності апаратного забезпечення, але й з розробкою нових досконаліших алгоритмів розпізнавання об'єктів на цифровому зображенні.

Деякі галузі промисловості раніше навіть не передбачали, або через недостатній рівень технологій просто не могли використовувати тогочасні системи комп'ютерного зору для своїх цілей. Це стосувалось в основному сфер виробництва з високими вимогами до точності класифікації або позиціонування об'єктів. Але в ході технічного прогресу постійно зростає деталізація та якість зображень, отримуваних за допомогою фото- та відеокамер, розробляються нові, більш досконалі алгоритми розпізнавання, позиціонування та 3D-реконструкції об'єктів за допомогою цифрових знімків. Так стає можливим впровадження систем комп'ютерного зору там, де раніше цієї можливості не було.

Подібна ситуація склалась і в сфері цифрової нарізки. Тут впровадження систем комп'ютерного зору відбувалось крок за кроком, відповідно до ступеню розвитку техніки. Першим і головним кроком стало встановлення *локальної камери* прямо біля ріжучого механізму. Її «погляд» охоплює лише невелику ділянку різальної поверхні безпосередньо під різцем. Дані, отримані від камери після ретельного дослідження всієї різальної поверхні, дозволяють скласти точну карту розміщення об'єкта(-ів) на ній і визначити програму слідування різака по матеріалу нарізки згідно з заздалегідь визначеним шаблоном. Близькість до об'єктів на різальній поверхні забезпечила високу точність розпізнавання і позиціонування об'єктів, і, як наслідок максимальну точність нарізки. Всі сучасні системи цифрової нарізки використовують дану технологію як базову.

Однак такий підхід має суттєвий недолік – швидкодію. Різальному механізму з камерою необхідно пройти вздовж усього контуру об'єкта нарізки щоб скласти повну і точну картину для подальшого нарізання. Частково вирішити цю проблему дозволяє реконструкція позиції всього об'єкта, базуючись на отриманих даних про положення його частини. Втім, економія часу за рахунок цього неминуче призводить до втрати точності, оскільки розрахунок ведеться на ідеально рівний, недеформований відносно шаблону об'єкт, чого інколи досить важко уникнути.

Інший підхід до рішення проблеми швидкодії полягає у нанесенні на нарізуваний матеріал опорних маркерів, так

званих реєстраційних марок. Реєстраційні марки – це чорні круги діаметром від 8 до 16 міліметрів. Набір реєстраційних марок і їх взаємне розташування ідентифікують об'єкт і визначають спосіб його нарізки. Такий спосіб є основним у сфері серійної нарізки, де на неперервному матеріалі (наприклад, на рулоні текстилю, паперу і т. п.) можна розмістити довільну кількість промаркованих об'єктів для вирізання.

Тим не менш, навіть використання реєстраційних марок не дозволяє повністю вирішити проблему швидкодії, так як камері перед виконанням нарізки так само треба виконати пошук марок на всій різальній поверхні. Ще одним серйозним недоліком такого підходу до пошуку об'єктів нарізки є подвійне зношування механізмів різака, оскільки для виконання однієї операції нарізки йому необхідно двічі пройти вздовж різальної поверхні: перший раз для пошуку і позиціонування об'єктів на ній, а другий – для їх нарізки.

В даній статті розглядатимемо наступний етап технології цифрової нарізки, котрий полягає у застосуванні аналізу повного зображення всієї різальної поверхні і практично позбавлений недоліків локального пошуку об'єктів.

Мета даної статті – це опис загальної концепції технології обробки і аналізу зображення різальної поверхні для ідентифікації і позиціонування об'єктів на ній без суттєвих втрат точності.

1. Постановка задачі та вихідні дані

Загальна задача описуваної системи комп'ютерного зору полягає в ідентифікації об'єктів на різальній поверхні станка для цифрової нарізки за допомогою растрового зображення цієї поверхні разом з об'єктами нарізки на ній. Цільова система призначена виключно для роботи з об'єктами нарізки що мають універсальний спосіб ідентифікації – реєстраційні марки. Крім фотографії різальної поверхні, системі також необхідна база даних шаблонів нарізки у вигляді файлів з координатами реєстраційних марок у межах самого об'єкта, для ідентифікації об'єктів знайдених на фотографії різальної поверхні.

В результаті, описувана система повинна за мінімальний час, проаналізувавши зображення робочої поверхні різального станка, ідентифікувати всі розташовані на ній об'єкти, а також визначити їх положення відносно точки відліку різального механізму. Далі отримані дані передаються керуючій програмі різального станку у вигляді структур даних, що містять ідентифікатор об'єкта, та координати всіх його реєстраційних марок щодо заданої точки відліку. Процес позиціонування буде детально розглянутий далі в даній статті.

Для забезпечення технічної бази описуваної системи комп'ютерного зору необхідне встановлення додаткової *глобальної фотокамери* над різальною поверхнею станка для цифрової нарізки. З метою отримання оптимального для аналізу зображення, камеру слід розташувати над центром різальної поверхні, так щоб оптична вісь була перпендикулярна до неї. Висоту треба обрати достатню для отримання зображення всієї робочої поверхні станка з мінімумом зайвого на фото. Чим вища роздільність та якість знімків камери, тим кращою буде точність позиціонування і ідентифікації об'єктів на них, що свідчить на користь використання якомога більш якісних фотокамер для даної цілі.

Оскільки цифрові різальні станки оснащені комп'ютером за замовчуванням, вимога наявності базової обчислювальної техніки для запуску цільової програми задовольняється заздалегідь. Необхідна лише наявність операційної системи сімейства Windows для запуску та коректної роботи програми.

Реалізація основних алгоритмів обробки та аналізу растрових зображень вже є у відкритій бібліотеці OpenCV (англ. Open Source Computer Vision Library), яку може вільно використовувати в академічних та комерційних цілях. Алгоритми цільової програми повністю базуються на використанні даної бібліотеки.

2. Калібрування камери і позиціонування об'єктів

Перед початком аналізу зображення різальної поверхні слід переконатись, що воно повністю відображає реальність. Отримані за допомогою фото-зйомки зображення часто «страждають» від різних оптичних спотворень (дисторсії) [2], вплив яких варіюється в залежності від якості оптики об'єктива камери. На противагу, існують доволі ефективні методи боротьби з оптичними спотвореннями на фотографіях.

В першу чергу необхідно виконати калібрування камери. Калібруванням камери називають розрахунок її внутрішніх і зовнішніх параметрів за отриманими з її допомогою фото та відео. В результаті отримується необхідна для усунення дисторсії інформація про камеру.

В розрахунках коефіцієнтів дисторсії бібліотека OpenCV враховує як тангенціальні та і радіальні складові [3]. Для радіальної складової використовується наступна формула:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6),$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6).$$

Таким чином, піксельна точка на спотвореному зображенні маючі координати (x, y), отримує координати $(x_{corrected} y_{corrected})$ на виправленому зображенні. Присутність радіальної дисторсії виявляється в ефектах «бочки» та «риб'ячого ока» на фотографіях.

Тангенціальна дисторсія виникає через те, що лінзи об'єктива камери не можуть бути ідеально паралельні площині, яка фотографується. Її можна усунути за наступною формулою:

$$x_{corrected} = x + [2 p_1 xy + p_2 (r^2 + 2x^2)],$$

$$y_{corrected} = y + [p_1 (r^2 + 2y^2) + 2 p_2 xy].$$

В підсумку можна виділити 5 параметрів спотворення, які в OpenCV представлені у вигляді однорядкової матриці з п'ятьма стовпцями.

$$Distortion_{coefficients} = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3).$$

Для перетворення координат точок реального світу в координати пікселей на фотознімку, застосовується *матриця камери:*

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}.$$

Присутність складової w у складі матриці піксельних координат пояснюється необхідністю дотримання відповідності розмірностей матриць координатних систем при множенні, і не носить реального змісту. Невідомими параметрами є f_x і f_y – фокусні відстані камери, та (c_x, c_y), які представляють оптичні центри, виражені в піксельних координатах. Якщо для обох осей використовується спільна фокусна відстань зі співвідношенням a (зазвичай 1), тоді $f_y = f_x * a$ і рівняння вище буде мати єдину фокусну відстань f.

Процес знаходження матриць камери і коефіцієнтів дисторсії називається калібруванням. Розрахунки їх параметрів виконуються за допомогою геометричних рівнянь. Застосовувані рівняння залежать від типу об'єктів, що використовуються для калібрування. В даний момент OpenCV підтримує 3 типи об'єктів для калібрування:

- класична чорно-біла шахова дошка;
- симетричний круговий патерн;
- асиметричний круговий патерн.

Необхідно зробити знімок цих патернів і знайти їх на ньому за допомогою засобів OpenCV. Кожному знайденому патерну відповідає нове рівняння. Для рішення цього рівняння треба зробити деяку кількість знімків і проаналізувати їх, щоби скласти систему з рівнянь. Наприклад, теоретично, шаховий патерн передбачає мінімум 2 знімки. Але цього можна уникнути, зробивши один знімок кількох патернів одразу.

Для наших цілей краще за все підходить шаховий патерн, оскільки він, по суті, потребує лише «сітку» з точок. Справа в тому, що шахова дошка є лише обгорткою для полегшення розпізнавання. Насправді OpenCV цікавлять лише позиції вузлів шахового патерну (точок зіткнення 4 клітинок).

Таким чином, необхідно лише надрукувати і сфотографувати *калібрувальну сітку*, яка складається зі зручних для розпізнавання об'єктів, розташованих у вузлах уявної решітки з комірками у вигляді квадратів з попередньо заданою стороною. Найбільш зручними для розпізнавання об'єктами будуть вище згадані реєстраційні марки, які мають радіальну симетрію з огляду на їх круглу форму і можуть бути легко розпізнані як камерою на різаку станка, так і засобами OpenCV. Сторони квадрата особливого значення для розпізнавання не має, але мусить бути достатньо великою щоб, вузли сітки зливались між собою на фотографії, і водночас достатньо малою щоб забезпечити велику точність позиціонування. Останнє твердження витікає з того, що калібрувальна сітка в нашому випадку служить для двох цілей – надає патерн для калібрування камери і утворює основу системи позиціонування об'єктів на різальній поверхні.

Розроблювана система передбачає нерухоме закріплення камери над різальною поверхнею, з чого слідує, що один і той самий піксель на різних фотографіях камери буде представляти одну й ту саму точку в просторі (на різальній поверхні). Залишається лише задати точку відліку і встановити опорні точки на різальній поверхні з відомими координатами, відносно яких будуть розраховуватись координати всіх інших.

Позиціонування в межах самої калібрувальної сітки є найпростішою задачею, оскільки крок сітки (сторона квадрата комірки) відомий заздалегідь. Зробивши знімок калібрувальної сітки (*калібрувальний знімок*), достатньо тільки розрахувати відношення відстані в міліметрах до відстані в пікселях між вузлами сітки і користуватись його значенням для знаходження відстаней в межах сітки. Наприклад, між вузлами калібрувальної сітки реальна відстань складає 50 мм, а на знімку – 100 пікселей. Співвідношення мм на 1 піксель (mm/px) складає ½. Знаючи, що відстань на фотографії між об'єктами складає, скажімо 482 пікселі, достатньо помножити це значення на розраховане раніше відношення mm/px щоб взнати реальну відстань в міліметрах: 241 мм.

Для позиціонування щодо точки відліку різака необхідне разове застосування локальної камери, яка має знайти кожен вузол калібрувальної сітки і надати його координати, знайдені з високою точністю*, системі комп'ютерного зору, яка «прив'яже» реальні координати вузлів у міліметрах щодо точки відліку, до їх координат на фотографіях у пікселях. Це буде єдиний етап роботи розроблюваної системи, коли треба застосовувати камеру різака для пошуку марок. Заново виконувати таку процедуру потрібно буде лише у випадку необхідності зміни калібрувальної сітки (з іншим кроком, патерном тощо) і повторного виконання калібрування вже для нової сітки.

Подальша схема позиціонування нічим не відрізняється від поданої вище. Проте точність буде вищою, оскільки без даних від локальної камери ми можемо знати лише заданий при виготовленні крок між вузлами калібрувальної сітки, який у процесі її виготовлення (при друкуванні) може бути спотворений у силу різних технічних факторів.

Все вище сказане свідчить про важливість міцної фіксації камери щодо різальної поверхні. Найменше її зміщення призведе до невідповідності наступних знімків калібрувальному та необхідності виконувати процес калібрування заново, який, враховуючи час на пошук вузлів камерою різака і в залежності від розмірів стола та калібрувальної сітки може зайняти досить багато часу.

В контексті описуваної системи комп'ютерного зору, процес калібрування включає не тільки знаходження матриць властивостей камери. Це також процедура

* зазвичай локальні камери забезпечують точність +-0.5 мм

повної підготовки системи до подальшої коректної роботи з ідентифікації та позиціонуванню об'єктів на різальній поверхні на основі калібрувальної фотографії та даних від камери різака.

Після отримання даних від обох камер, інформації про розміри та крок калібрувальної сітки та команди запуску калібрування, програма починає пошук реєстраційних марок на калібрувальній фотографії. Процедуру пошуку марок буде детально описано в наступному розділі даної статті. Результати пошуку у вигляді списку координат пікселей центрів марок на фотографії проходять процедуру пошуку закономірностей розташування, на основі яких будуються об'єкти калібрувальних сіток. Основу цих об'єктів складають три двомірні масиви точок, розміри яких збігаються з розмірами сіток, а комірки призначені для збереження координат вузлів сітки, тобто центрів марок. Перший масив зберігає координати вузлів сітки в піксельних координатах фотографії, а другий – отримані від камери різака відповідні координати реального світу в міліметрах. Третій призначений для зберігання неспотворених координат вузлів на «вирівняній» фотографії.

Після побудови всіх об'єктів сіток відповідно до кількості реальних сіток на фотографії, та заповнення їх перших масивів у порядку реального розташування вузлів, відбувається пошук відповідності розташування комірок першого масиву та точок, отриманих від камери різака. Таким чином встановлюється однозначна відповідність реальних координат вузлів калібрувальних сіток у міліметрах до їх координат на фотографії в пікселях та заповнюються другі масиви об'єктів сіток. На основі отриманих даних визначається також позиція точки відліку (початку координат) на фотографії.

Далі слідує власне калібрування камери. Для отримання калібрувальних матриць OpenCV пропонує метод calibrateCamera() [4]. Він приймає координати точок патернів (калібрувальних сіток) в просторі, їх координати на фотографії (перші два масиви об'єктів сіток), а також пусті матриці для власних коефіцієнтів та коефіцієнтів спотворення камери, які заповнює розрахованими значеннями. Отримані матриці зберігаються і далі використовуються для розрахунку «правильних» координат вузлів калібрувальної сітки за допомогою функції OpenCV undistortPoints(). Перераховані неспотворені координати записуються в третій масив об'єктів сіток. В подальшому всі точки, знайдені на фотографіях, будуть проходити таку обробку для підвищення точності позиціонування.

На цьому процес калібрування закінчується. В результаті, програма комп'ютерного зору отримує всю необхідну інформацію для точної ідентифікації та позиціонування об'єктів на наступних фотографіях різальної поверхні.

3. Обробка зображення та знаходження реєстраційних марок

Найголовнішою задачею розроблюваної системи комп'ютерного зору є пошук реєстраційних марок на фотографії. Бібліотека OpenCV пропонує широкий набір інструментів для пошуку різноманітних геометричних об'єктів на растрових зображеннях. Трохи спрощує задачу сама форма марки – круг переважно чорного кольору. На бінарному зображенні це виглядатиме як скупчення темних пікселей з округлим контуром. Спеціально з ціллю пошуку таких об'єктів було розроблено клас SimpleBlobDetector [5]. Цей клас реалізує простий алгоритм для знаходження чорних «плям», до яких як раз відносяться реєстраційні марки на зображенні.

Спочатку виконується конвертація кольорового зображення в кілька чорно-білих з застосуванням різних порогів від мінімального до максимального заданих (або прийнятих за замовчуванням) з деяким кроком. Суть застосування порогів полягає у порівнянні значення пікселя зі значенням порогу. Попередньо зображення з кольорового переводиться у відтінки сірого. Далі, якщо значення пікселя більше за значення порогу, йому присвоюється один колір (наприклад, чорний), а якщо менше – інший (наприклад, білий). Далі

алгоритм проводить пошук контурів на отриманих чорно-білих зображеннях та знаходить їх центри. Потім, знайдені на різних зображеннях контури групуються по координатах. Близько розташовані центри формують єдину групу, яка відповідає окремій «плямі». Нарешті, аналізуючи кожну групу, алгоритм знаходить остаточні центри та радіуси «плям», та повертає їх у формі колекції структур, що представляють координати та властивості знайдених «плям».

Також клас перед поверненням результатів піддає фільтрації за заданими параметрами знайдені «плями». До цих параметрів відносяться.

1. Яскравість. Фільтр порівнює інтенсивність кольору в центрі «плями» з заданим параметром blobColor. Якщо вони сильно різняться, «пляма» відкидається.

2. Площа. Допускаються лише «плями», що мають площу від заданої параметром minArea (включно) до maxArea (не включно). Тобто встановлення minArea рівним 100, відкине всі контури з площею менше 100 пікселей.

3. Округлість. Тут оцінюється на скільки сильно «пляма» схожа на коло. Наприклад, шестикутник буде більш округлим аніж, скажімо, квадрат. Треба лише встановити відповідне значення параметрам minCircularity та maxCircularity (від 0 до 1). Округлість розраховується за формулою:

$$\frac{4\pi Area}{(perimeter)^2}.$$

З якої слідує, що ідеальне коло має округлість рівну 1, квадрат – 0,785 і так далі.

4. Опуклість. Ступінь опуклості визначається як відношення площі контуру до площі її опуклої оболонки. Опуклою оболонкою контуру є найменший опуклий контур, який повністю включає у себе заданий. Для включення цього фільтра необхідно встановити значення minConvexity та maxConvexity (обидва від 0 до 1).

5. Коефіцієнт інерції. Вимірюється ступінь видовженості контуру. Коло має цей коефіцієнт рівний 1, у еліпса від 0 до 1, а в лінії рівно 0. Для використання фільтрації за коефіцієнтом інерції необхідно встановити параметри minInertiaRatio та maxInertiaRatio відповідно.

Однак у класу SimpleBlobDetector є один суттєвий недолік – нечутливість до кольору. Тому він підходить лише для пошуку чорних реєстраційних марок. Що стосується марок інших кольорів, то тут варто звернути увагу на геометричні властивості марки. Як не повертай коло однорідного кольору навколо його центру – його вигляд завжди залишатиметься однаковим. Така його властивість дозволяє застосувати до нього метод *співставлення зразка*.

Співставлення зразка – це метод пошуку місця зразка на зображенні більшого розміру. Іншими словами, маючи шматочок зображення, можна знайти його положення на цілому зображенні. В OpenCV цей метод реалізує функція matchTemplate(). Вона просто переміщує зразок по цільовому зображенню та порівнює зразок з поточним шматочком під ним. Існує декілька реалізацій методу співставлення зразка в OpenCV. Вибрати конкретний можна встановленням значення одного з параметрів matchTemplate(). Повертає функція сіре зображення, де кожен піксель показує наскільки сусідня з ним область збігається із зразком [6].

Якщо основне зображення має розміри (WxH), а зразок (wxh), вихідне зображення матиме розміри (W–w+1, H–h+1). Як тільки результуюче зображення отримано, можна використовувати функцію minMaxLoc() для отримання координат мінімального (максимального – в залежності від використаного методу співставлення зразка), за шкалою сірого пікселя. Це буде верхній лівий кут області, що відповідає зразку.

Однак, такий метод підходить лише для випадку колу розшукується єдине збігання заданого зразка на цільовій фотографії. Якщо таких збігів очікується більше

одного, необхідно використовувати поріг відтінків сірого [7]. Якщо метод передбачує мінімальне значення пікселя (світлий) в області збігання, то необхідно встановити поріг від 0 до 1 ближче до 0 і циклічно викликати функцію minMaxLoc(), порівнювати найменше значення яке вона повертає з заданим порогом. Якщо воно менше, то можна зберігати його координати та зафарбовувати вже знайдену область в колір зворотній шуканому (для мінімуму – чорний) для запобігання повторного повернення тієї ж самої координати. Якщо ж minMaxLoc() повернула мінімальне значення більше порогового, значить можна переривати цикл – досить чіткі збігання закінчились.

Для методів, які повертають збігання з максимальним значенням відтінку сірого (темним) пікселя, загальний принцип зберігається, але поріг береться ближче до 1, а приймаються точки з більшим ніж порогове значенням відтінку.

Використання співставлення зразка в описуваній системі потребує лише створення кількох зразків марок (простим вирізанням області марки з фотографії) різного кольору, діаметру, освітленості та інших параметрів, що впливають на її вигляд. Далі можна застосовувати пошук відповідного зразка в залежності від того які марки представлені на фотографії.

Загальним недоліком як класу SimpleBlobDetector, так і методу співставлення зразка є знаходження на фотографії зайвих чорних точок окрім реєстраційних марок. Одним з можливих рішень може бути жорстке обмеження значень параметрів для SimpleBlobDetector і зменшення порога для співставлення зразка. Однак, так скоріш за все будуть відкинуті й потрібні, але з тих чи інших причин (освітлення, тіні й т. п.) невдало сфотографовані марки.

Найкращим варіантом використання цих методів разом за даних умов, буде їх комбінування з застосуванням великих допусків з наступним прийняттям тільки тих результатів, які знайдені обома алгоритмами. Коли стосується пошуку кольорових марок, то тут варто використовувати лише співставлення зразка.

4. Ідентифікація об’єктів нарізки

Ідентифікація об’єктів нарізки починається із створення знімку різальної поверхні глобальною камерою. Разом із знімком системі комп’ютерного зору також на вхід подається шаблон(и) об’єктів розпізнавання які присутні на знімку, та зразки марок для розпізнавання, якщо планується використовувати метод співставлення зразка. Після отримання всіх даних та відповідної команди система починає власне процес ідентифікації.

Спочатку відбувається пошук реєстраційних марок описаними попередньо способами. Результат пошуку у вигляді списку координат центрів знайдених реєстраційних марок подається на вхід процедури пошуку відповідностей шаблонам об’єктів нарізки. Запускається цикл, який за чергою перевіряє наявність конфігурації марок поточного шаблону серед знайдених на фотографії різальної поверхні.

Алгоритм пошуку відповідностей шаблонам нарізки полягає у використанні деяких елементів з теорії графів. Марки шаблону мають координати та утворюють вершини зваженого графа з ребрами, вага яких дорівнює відстані між марками шаблону. Залишається лише знайти такий самий граф серед усіх знайдених на знімку реєстраційних марок. Зрозуміло, що пошук марок допускає, що деякі марки на фото можуть бути не знайдені через недосконаліть алгоритмів пошуку, якість зображення, освітлення тощо. Це означає, що відсутність деяких вершин цільового графа серед знайдених на знімку реєстраційних марок є допустимою, якщо з урахуванням всіх інших наявних марок, можна встановити однозначну їх відповідність шуканому шаблону об’єкта нарізки.

Оскільки координати марок шаблону дані в міліметрах відносно невідомого нам початку координат матеріалу, на який ці марки нанесені, оперувати можна лише абсолютними значеннями відстаней між ними. Результатом процедури калібрування, яка має бути обов’язково виконана перед ідентифікацією об’єктів нарізки, є система позиціонування, яка дозволяє досить точно визначати відстані між

точками на фотографії. Так для кожної знайденої марки ми можемо знайти всі відстані від неї до інших (ребра графа) та порівняти їх з набором ребер між марками з шаблону, виявивши збігання. В цьому і полягає основна ідея алгоритму пошуку відповідностей набору знайдених реєстраційних марок заданим шаблонам об'єктів нарізки.

Процедура розпізнавання шаблону об'єкта нарізки серед марок виглядає наступним чином. На вхід приймається цільовий шаблон та набір піксельних координат знайдених на знімку марок. Спочатку перевіряється чи достатньо знайдено марок на фото. Якщо всього знайдених марок менше ніж половина від кількості марок у шаблоні об'єкта, то процедура переривається, вважаючи, що об'єктів з таким шаблоном на знімку різальної поверхні немає. Далі циклічно перебираються всі марки шаблону. Для кожної марки шаблону запускається ще один внутрішній цикл, який проходить по всім знайденим на фото маркам та перевіряє їх відповідність поточній марці шаблону за допомогою порівняння їх наборів ребер. Для цього розраховуються всі відстані в міліметрах (ребра) від поточної знайденої марки до інших, а ребра шаблонних марок розраховуються попередньо, ще при зчитуванні з файлу та містяться в спеціальній структурі, яка представляє шаблонну марку разом з її ребрами. Далі проходить процедура пошуку ребер шаблонної марки серед ребер марки з фото.

Якщо марка, знайдена на знімку, має більше 50 % ребер, що відповідають марці з шаблону, вона заноситься до *словника пар відповідностей* вигляду: «марка шаблону – марка на фотографії», та видаляється з набору знайдених марок. Перебір продовжується далі, оскільки на різальній поверхні може бути одночасно кілька однакових об'єктів нарізки, тож одній марці з шаблону може відповідати декілька марок із знімку.

Слід зазначити, що порівняння ребер відбувається з допуском в 5 % від довжини ребра шаблонної марки, адже повної рівності довжин відповідних ребер бути не може через присутність допустимої похибки в системі позиціонування (приблизно +-1 мм).

Після закінчення циклу пошуку марок шаблону серед знайдених на знімку методом порівняння наборів ребер, маємо заповнений словник їх відповідностей. Якщо кількість пар у словнику менша ніж половина від кількості марок в цільовому шаблоні, процедура закінчується, оскільки це означає, що об'єктів з таким шаблоном на різальній поверхні немає. Якщо ж пар в словнику достатньо, то починається пошук вже окремих екземплярів заданого шаблону об'єкта серед відібраних пар марок.

Береться довільна (перша) *опорна пара* зі словника «марка шаблону – марка на фотографії», та видаляється з нього. Далі, програма намагається знайти відносно неї всі інші марки, які разом сформують об'єкт нарізки. Одразу ж створюється новий програмний екземпляр об'єкта, який містить контейнер для його марок та поле ідентифікатора шаблона нарізки. До списку його марок заноситься позиція марки на фото з опорної пари. Також йому присвоюється ідентифікатор поточного шаблону. Після цього починається цикл перебору ребер шаблонної марки з опорної пари, всередині якого знаходиться вкладений цикл, який проходить по всім парам словника. У внутрішньому циклі для шаблонної марки поточної пари розраховується відстань від неї, до шаблонної марки опорної пари та порівнюється з поточним ребром зовнішнього циклу. Якщо вони рівні, то розраховується відстань від марки на фото з поточної пари до марки на фото з опорної пари. Якщо ця відстань приблизно (з урахуванням 5 % допуску) рівна значенню поточного ребра, то поточна пара є ще однією маркою об'єкта нарізки, і значення марки на фото додається до списку марок екземпляра цього об'єкта.

Після закінчення обох циклів, перевіряється кількість марок у списку екземпляра об'єкта. Якщо вона більша ніж половина кількості марок шаблона, то такий об'єкт приймається як розпізнаний та додається в колекцію для повернення процедурою. Якщо в словнику залишилось ще достатньо марок, то це означає

що потенційно є ще один екземпляр такого шаблону об’єкта. Обирається нова опорна пара і для неї так само виконуються описані вище дії.

Після того як в словнику залишиться замало пар для відбудови об’єкта шаблону, або не залишиться зовсім, процедура повертає колекцію розпізнаних об’єктів нарізки, що відповідають заданому шаблону. Ці об’єкти в свою чергу додаються до контейнера, що містить всі розпізнані об’єкти різних шаблонів, адже на різальній поверхні одночасно можуть бути розміщені представники різних шаблонів нарізки.

Після завершення процедури розпізнавання, програма повертає колекцію, де зібрані всі розпізнані об’єкти нарізки з їх унікальними ідентифікаторами, та координатами марок у міліметрах. Форма даних що повертає система комп’ютерного зору залежить від того, в якому вигляді їх приймає на вхід комп’ютер різального станка (файл, пакети даних тощо).

Отримані дані дозволяють програмі комп’ютера станка сформувати траєкторію руху різака по робочій поверхні та правильно вирізати розпізнані об’єкти на ній.

Висновки

Розроблена та описана концепція системи комп’ютерного зору для точної ідентифікації та позиціонування об’єктів цифрової нарізки з реєстраційними марками за фотознімком робочої поверхні станка для цифрової нарізки. Створено алгоритми роботи модулів системи, які відповідають за калібрування камери та утворення системи позиціонування на основі калібрувальної сітки та даних від локальної камери різака, розпізнавання реєстраційних марок двома принципово різними способами, які доповнюють один одного, а також алгоритм ідентифікації об’єктів нарізки на основі застосування елементів теорії графів.

1. Шаіпро Л., Стокман Дж. Комп'ютерний зір: переклад з англ. М.: БИНОМ. Лаборатория знаний, 2006. 752 с.
2. Іофис Є.А. Фотокінотехніка. М.,: «Советская энциклопедия», 1981. 447 с.
3. Шеліський Р. Комп'ютерний зір: Алгоритми та Застосунки. Нью-Йорк: Springer, 2011. 957 с.
4. Калібрування камери з OpenCV [Електронний ресурс]. Документація OpenCV 2.4.13.3. Режим доступу до ресурсу: http://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html.
5. Маллік С. Знаходження точок за допомогою OpenCV [Електронний ресурс]. Вивчення OpenCV. 2015. Режим доступу до ресурсу: https://www.learnopencv.com/blob-detection-using-opencv-python-c/.
6. Співставлення зразка [Електронний ресурс]. Документація OpenCV 2.4.13.3. Режим доступу до ресурсу: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html.
7. Брадський Г.Р., Келер А. Вивчаючи OpenCV. Себастополь: O’Reilly Media, 2008. 556 с.

References

1. Shapiro, L. & Stockman, G. (2006) Computer vision. Moscow: BINOM. Laboratoriya znaniy. (in Russian)
2. Iofis, E.(1981) Fotokinotehnika. Moscow: “Sovietskaya encyklopediya”. (in Russian)
3. Szeliski, R. (2011) Computer Vision: Algorithms and Applications. New York: Springer.
4. OpenCV 2.4.13.3 documentation. Camera calibration With OpenCV. Available at: http://docs.opencv.org/2.4/doc/tutorials/calib3d/camera_calibration/camera_calibration.html.(accessed 1 October 2017).
5. Satya, M. (2015) Learn OpenCV. Blob Detection Using OpenCV. Available at: https://www.learnopencv.com/blob-detection-

using-opencv-python-c/. (accessed 29 September 2017).
6. OpenCV 2.4.13.3 documentation. Template Matching. Available at: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html (accessed 1 October 2017).
7. Bradski, G. & Kaehler, A. (2008) Learning OpenCV. Sebastopol: O'Reilly Media.

Одержано 04.10.2017

***Про авторів*:**

Туманов Владислав Валерійович,
студент магістерської програми НТУУ «КПІ імені Ігоря Сікорського»
Кількість наукових публікацій в українських виданнях – 2.
http:// orcid.org/0000-0002-1813-5021,

Дорошенко Анатолій Юхимович,
доктор фізико-математичних наук, професор, завідувач відділом теорії комп'ютерних обчислень Інституту програмних систем НАН України
професор кафедри автоматики та управління в технічних системах НТУУ "КПІ імені Ігоря Сікорського".
Кількість наукових публікацій в українських виданнях – більше 150.
Кількість наукових публікацій в закордонних виданнях – більше 40.
Індекс Хірша – 5.
http://orcid.org /0000-0002-8435-1451,

Місце роботи авторів:

НТУУ «КПІ імені Ігоря Сікорського».
03056, Київ, просп. Перемоги, 37,
Тел.: (095) 486 3307.
E-mail: tumanovlad@gmail.com.

Інститут програмних систем
НАН України.
03187, Київ,
проспект Академіка Глушкова, 40.
Тел.: (044) 526 3559.
E-mail: doroshenkoanatoliy2@gmail.com