Applied Intelligence
https://doi.org/10.1007/s10489-022-04068-0

Closed-loop feedback registration for consecutive images of moving flexible targets

Rui Ma[1,2] · Xian Du[1,3]

Accepted: 5 August 2022

Abstract
Advancement of imaging techniques enables consecutive image sequences to be acquired for quality monitoring of manufacturing production lines. Registration for these image sequences is essential for in-line pattern inspection and metrology, e.g., in the printing process of flexible electronics. However, conventional image registration algorithms cannot produce accurate results when the images contain duplicate and deformable patterns in the manufacturing process. Such a failure originates from the fact that the conventional algorithms only use spatial and pixel intensity information for registration. Considering the nature of temporal continuity of the product images, in this paper, we propose a closed-loop feedback registration algorithm. The algorithm leverages the temporal and spatial relationships of the consecutive images for fast, accurate, and robust point matching. The experimental results show that our algorithm finds about 100% more matching point pairs with a lower root mean squared error and reduces up to 86.5% of the running time compared to other state-of-the-art outlier removal algorithms.

Keywords feedback registration · image sequence registration · point pattern matching (PPM) · scale-invariant feature transform (SIFT)

1 Introduction

Image registration is a process of transforming two or more images into a single coordinate system. It is a fundamental step in many image processing and computer vision tasks, such as biomedical image diagnosis [1], super-resolution [2], 3-D reconstruction [3], motion tracking [4], image retrieval [5], and action recognition [6]. Designing and developing an accurate, effective, and robust image registration algorithm is critical for these tasks because the registration results have a significant impact on the quality of the follow-up data processing steps.

Over the past decades, many image registration algorithms were proposed; these algorithms can be categorized into area-based methods [7, 8] and feature-based methods [9–12].

Area-based methods, sometimes called correlation-like methods, take the pre-defined, fixed-size image patch or even the entire image as the reference and compare it with the target image through a specific similarity measurement. After locating the most similar position on the target image, the corresponding position pair between the reference and the target is created. However, there are several limitations of the area-based registration methods. First, the image patch, commonly defined as the rectangular window, suits the registration of images that are locally deformed by a translation transformation. If the images are deformed by more complex transformations, the image patch cannot capture these complex transformations. Second, if the target images contain too many similar and deformable patterns, the pre-defined reference patch cannot locate the most similar position because many positions of the similar patterns will have indistinguishable similarities with the reference patch. Third, if the entire image is the reference, the area-based method will suffer from high computational complexity.

Compared to the area-based methods, feature-based methods are more robust to handle complex image distortions. They also require less computational work than the full image correlation-like methods. Conventional feature-based registration algorithms mainly include five steps. First, interest points are detected at distinctive locations in both the reference and

✉ Xian Du
xiandu@umass.edu

[1] Center for Personalized Health Monitoring, Institute for Applied Life Sciences, University of Massachusetts, Amherst, MA 01003, USA

[2] Electrical and Computer Engineering Department, University of Massachusetts, Amherst, MA 01003, USA

[3] Mechanical and Industrial Engineering Department, University of Massachusetts, Amherst, MA 01003, USA

Published online: 22 August 2022

the target images. The interest points can be corners [13], blobs [9], saddles [12], etc. Second, a certain region around every interest point is represented by a feature descriptor – typically an $n \times 1$ vector (e.g., SIFT [9], SURF [10], BRIEF [11]). Third, the feature vectors are matched between the reference and the target images based on the distance of the feature vectors (e.g., Mahalanobis distance, spectral angular distance (SAD), and Euclidean distance). Fourth, a transformation model can be estimated based on the point-to-point matching result. Finally, the transformation model is implemented on the target image. Moreover, the non-integer coordinates of the transformed target image can be interpolated appropriately. However, the feature-based methods still suffer from the lack of accuracy if the reference and the target images have too many similar and deformable patterns.

In [7], a correlation-based matching algorithm was proposed. The authors demonstrated that the inclusion of periodicity information in registration and matching can significantly improve the accuracy and robustness of the image registration. Inspired by the periodicity registration algorithm, we believe the registration accuracy and speed can be improved by taking advantage of the continuous spatiotemporal information between consecutive images and the speed stability of the moving target. However, few of the feature-based methods used the periodicity information.

Therefore, we propose a feature-based closed-loop feedback registration algorithm that will include the continuity of consecutive images and smoothness of the target motion. The first novelty of the proposed algorithm takes advantage of the preliminary spatiotemporal information of the consecutive image sequence. We generate a coarse translation model to guide the feature matching process instead of using the statistical model-based outlier removal methods. This approach can expedite the feature matching by reducing the computation complexity from $O(n^2)$ to $O(n)$.

The second novelty of the proposed algorithm improves the accuracy of the coarse translation model with the closed-loop feedback process. Consequently, this coarse-to-fine closed-loop feedback registration method expedites the matching computation and improves the matching accuracy and robustness in the whole registration process.

Our proposed algorithm consists of three steps. First, we implement a feature detector and descriptor (both conventional and deep-learning-based feature detectors and descriptors are included in the experiments) to locate and represent the interest points in all images. Second, we implement the closed-loop feedback algorithm to match the interest points between each pair of the reference and the target images in a consecutive image sequence. Third, we stitch the series of the consecutive images using both direct linear transformation (DLT) and As-Projective-As-Possible (APAP) [14] transformation.

2 Related work

The related work of feature-based methods is reviewed according to the five steps of the image registration process. Researchers have optimized interest point detection methods for various situations. In [13], a gradient-based method was adopted to find corners. The corners, named Harris corners, are detected by the second-moment matrix or second-order auto-correlation matrix. However, Harris corners are very sensitive to the variation of image scales. In addition to corners, blob feature detectors are also adopted in many applications. In [9], Lowe introduced the SIFT detector to find blob features by the difference of Gaussian (DoG). Similar to SIFT, in [10], the SURF detector was introduced to find both corner and blob features.

Researchers designed various descriptors to keep the feature vectors distinctive and robust to many irregularities including, noise, detection errors, and geometric and photometric deformations. SIFT [9] is one of the most widely used gradient-based feature descriptors. SIFT descriptor constructs a gradient histogram in a local region to account for each feature. A 128-dimension vector is then created by assigning a Gaussian weighting function to each detected point. SURF [10] is another popular gradient-based local feature descriptor that uses the Haar Wavelet response for assigning an orientation histogram to represent features. SURF also improves the speed of the feature detection and description in SIFT. Moreover, the faster hamming distance, rather than Euclidean distance, was used for feature matching. BRIEF [11] uses binary strings as the descriptors with the hamming distance metric. It is a binary string descriptor that obtains individual bits by comparing the intensities of paired points in the feature patch. ORB [15] descriptor is an extension of BRIEF that created the oriented BRIEF features and expedited their computation. Recently, researchers have deployed a deep-learning method for end-to-end feature detection and description. SuperPoint [16] is a self-supervised interest point detector and descriptor. The authors have presented a fully-convolutional neural network architecture for interest point detection and description trained using a self-supervised domain adaptation framework called Homographic Adaptation. They claimed SuperPoint [16] performs similarly to SIFT detector and descriptor.

Researchers have optimized other distance measurement metrics. The m_p-dissimilarity measure had been recently proposed in [17] which is suitable to measure the data dependency similarity. Unlike l_p-norm distance (Euclidean being l_2-norm), m_p-dissimilarity considers the relative positions of the two vectors with respect to the remaining data.

Unfortunately, these feature matching methods produce many incorrect matches when the pair of images contain a large number of similar features, such as the repeated regular printed patterns (grid-based images) in manufacturing [7]. To

eliminate the incorrect matching points in the third step of the registration task, statistical model-based methods using geometric constraints are commonly used. The most popular method is random sample consensus (RANSAC) [18], which has several extensions, such as MAGSAC++ [19]. These methods adopt a hypothesize-and-verify approach that attempts to find the best parametric model under their defined regulations. Nevertheless, these statistical model-based methods increase running time and perform unstably due to random sampling. In [20], an Iterative Scale-Invariant Feature Transform (ISIFT)-based registration algorithm was proposed. The authors continuously updated the matched point pairs between the sensed image and the reference image until an optimized transformation model was found. However, this method is time-consuming because of the large iterative optimization involved. Alternatively, end-to-end matching algorithms using deep-learning were proposed by some researchers. SuperGlue [21] deployed a neural network that matches two sets of local features by jointly finding correspondences and rejecting non-matchable points. Compared to traditional, hand-designed heuristics, the technique learns priors over geometric transformations and regularities of the 3D world through end-to-end training from image pairs. LoFTR [22] is another end-to-end deep-learning-based image feature matching algorithm. The authors used self and cross attention layers in Transformer [23] to obtain feature descriptors that are conditioned on both images.

Furthermore, some researchers attempted to take advantage of the prior information in their specific situations for accurate registration. In [7], a constellation matching algorithm was proposed using the Normalized Cross-correlation (NCC). The author used the preliminary spatial and pattern information of the images to achieve an accurate matching result. However, the templates for detecting interest points are needed and the initial reference point should be accurately paired at the beginning of the matching algorithm.

Finally, given the optimized matching point pairs, an image stitching algorithm is commonly adopted. The most straightforward method is to align images by estimating the 2-D projective warps. Parameterized by the 3 × 3 homography, the 2-D projective warps are justified in [24] if the scene is planar or if the views differ only by rotation. In practice, these viewing conditions may not be fully satisfied as the projective model can wrongly characterize the warp and thus cause misalignments or ghosting effects. To solve this problem, the APAP stitching method was proposed in [14] by constructing better alignment functions. The strategy of this method is to utilize the local projective models based on the locations of the images rather than using the global projective transformation model. This method can significantly reduce alignment errors while maintaining the overall geometric plausibility.

3 Proposed algorithm

Given the initial speed v_0 of the moving target (e.g., the moving web in Fig. 1) with insignificant image deformation, for any image frame at time t, $t > 0$, the moving distance between two consecutive frames can provide an approximate translation model of the image sequence as a prior registration at time t. Furthermore, the moving distance between the first frame and the second frame ($t = 0$ and $t = \Delta t$) can be obtained by $d_1 = v_0 \Delta t$, where Δt represents the time interval of two consecutive frames, and the moving distances d_i, $i > 1$, can be estimated by the previous moving distances, d_{i-1}. Then, the feature vectors could be matched by the guidance through the prior registration, which differs from the global search for all vectors spanning the entirety of the image. Next, based on the matching result at time t, a revised translation model can be fed into the next matching task at time $t + \Delta t$. Figure 2 shows the workflow of the system. Taking a series of frames as input, a constellation prediction model is created first; given the prior registration result, the point pattern matching (PPM) module can optimize the local registration and feed the registration result to the next matching task. In the following sections, we will detail each step of our registration algorithm.

3.1 2-D translation estimation

Assume there is a global 2-D transformation between every two consecutive images [24]. Given any two consecutive images $image_{k-1}$ and $image_k$, in a sequence of n images, for $1 < k \leq n$, we can transform all the pixels in $image_k$ from its coordinate system into the coordinate system of $image_{k-1}$. The image transformation can be represented as:

$$I_k^{k-1} = T_{proj}(k) \cdot I_k, \tag{1}$$

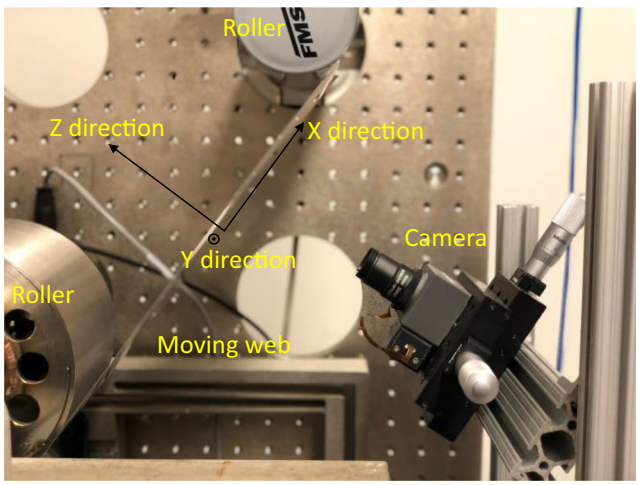


Fig. 1 Experimental setup

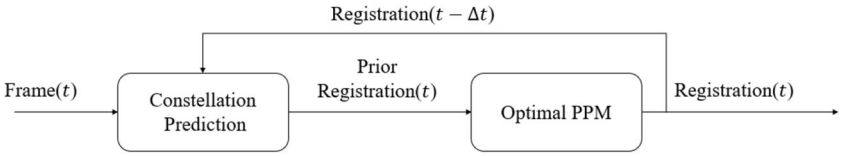


Fig. 2 Workflow of Closed-loop Feedback Registration System

where I_k^{k-1} and I_k represent $image_k$, respectively in the coordinates of $image_{k-1}$ and $image_k$. $T_{proj}(k)$ represents the 2-D projective transformation matrix between $image_{k-1}$ and $image_k$. Therefore, given I_k in the coordinate system of $image_1$, I_k^1 can be represented by:

$$I_k^1 = \prod_{i=1}^{k} T_{proj}(i) \cdot I_k. \tag{2}$$

For a manufacturing process, like roll-to-roll printing in Fig. 1, the product target is typically controlled to move unidirectionally. We define the moving direction as the x axis. Note that there is insignificant offset or motion variation in the y and z axes due to the vibration or dynamics of the roll-to-roll process. Readers can refer to Appendix A for more details of the vibration analysis of our roll-to-roll printing system. Given the setting speed of the moving conveyor ($v(m/s)$), the frame rate of the camera ($f(1/s)$), and the pixel scale of the images ($d(m/pixel)$), we can approximately calculate the transformation between two consecutive images along the x-axis. Figure 3 shows an example of the image acquisition process. For any frame at time t, $t > 0$, the moving distance (pixels) between two consecutive images, Δx, can be presented as:

$$\Delta x = \frac{v}{f \cdot d}. \tag{3}$$

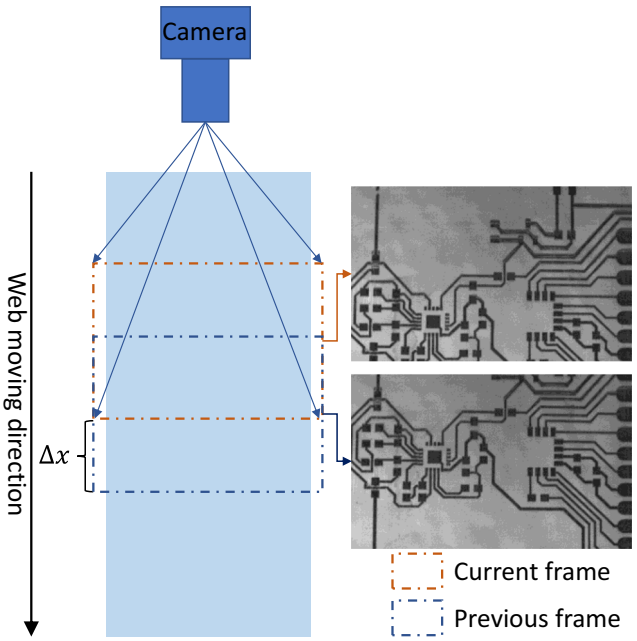


Fig. 3 An example of the image acquisition process

Given a sufficiently fast frame rate, we can estimate the global projective transformation model using a simpler translation model because the consecutive images in the short time interval will only have a significant offset in the x axis. Therefore, T_{proj} in Eq. (2) can be simplified by the translation transformation matrix T_{trans}, which can be expressed as:

$$T_{proj} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \rightarrow T_{trans} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4}$$

However, errors will be introduced in the estimated transformation due to the ignorance of offsets in y and z axes in Eq. (4). Substituting this simplified transformation matrix of Eq. (4) and its offset errors into Eq. (2), I_k^1 can be derived by:

$$I_k^1 = \prod_{i=1}^{k} [T_{trans}(i) + Error(i)] \cdot I_k, \tag{5}$$

where $T_{trans}(i)$ represents the i^{th} estimated translation matrix. $Error(i)$ represents the offset error in the i^{th} estimation. Equation (5) indicates that the errors will be accumulated in each transformation estimation. We will present the error correction process using a feedback loop in the next section.

3.2 Closed-loop feedback matching

The calculation in Section 3.1 involves the estimation of the global 2-D transformation between images. A follow-up algorithm for seeking the 2-D transformation matrices in Eqs. (2) and (5) usually starts with searching for the point correspondences between the reference and the target images.

Conventional feature point matching methods generally compare the similarity of the feature vectors between the reference and the target images. Without loss of generality, $image_{i-1}$ and $image_i$, where $1 < i \leq n$, can be used respectively for the reference and the target images taken from a consecutive image sequence $\{image_1, \cdots, image_n\}$. Assuming that m and n interest points are respectively detected in $image_{i-1}$ and $image_i$, the positions of the interest points are stored in the $m \times 2$ matrix $\boldsymbol{P}_{i-1}$ and $n \times 2$ matrix $\boldsymbol{P}_i$. $\boldsymbol{V}_{i-1}$ represents the matrix of the feature vectors for $image_{i-1}$ where each row of $\boldsymbol{V}_{i-1}$ corresponds to each point position of $\boldsymbol{P}_{i-1}$. Likewise, $\boldsymbol{V}_i$ represents the matrix of feature vectors for $image_i$. The similarity between feature vectors in $\boldsymbol{V}_{i-1}$ and $\boldsymbol{V}_i$ can be computed using a certain distance metric (e.g.,

Euclidean distance), which indicates whether the reference feature vectors match the target feature vectors. Take the first row of $\boldsymbol{V}_{i-1}$, $\boldsymbol{V}_{i-1}(1)$ as an example, to find a match for $\boldsymbol{V}_{i-1}(1)$, the distances between $\boldsymbol{V}_{i-1}(1)$ and all rows of $\boldsymbol{V}_i$ must be calculated. The matching row vector in $\boldsymbol{V}_i$ for $\boldsymbol{V}_{i-1}(1)$ should have the minimum distance and be below a certain threshold. However, this matching method misses the global temporal and spatial continuity between the consecutive images. Moreover, it incurs a high computation complexity of $O(n^2)$, where n represents the number of interest points. To improve the accuracy and speed of the PPM matching method, we can take advantage of the translation model in Section 3.1. Given the translation model $T_{trans}(i)$ in Eq. (4), we can easily derive the corresponding point position $\boldsymbol{P'}_{i-1}$ for point position $\boldsymbol{P}_{i-1}$, (by)

$$\boldsymbol{P'}_{i-1} = T^{-1}{}_{trans}(i)\boldsymbol{P}_{i-1}, \tag{6}$$

where $\boldsymbol{P'}_{i-1}$ represents the estimated locations of all the detected interest points in $image_{i-1}$ shown in the coordinate system of $image_i$, and $T^{-1}{}_{trans}(i)$ is the inverse matrix of $T_{trans}(i)$. Each position in $\boldsymbol{P'}_{i-1}$ is an estimated position of its matching point in $\boldsymbol{P}_i$. To find the matching points between $image_i$ and $image_{i-1}$, only a small range around the positions of $\boldsymbol{P'}_{i-1}$ needs to be considered. In other words, most of the interest points in $image_i$ outside the small range can be excluded. In summary, the matching point $\boldsymbol{P}_i(k_1)$ in $image_i$ for an interest point $\boldsymbol{P}_{i-1}(j_1)$ in $image_{i-1}$ can be found by (note that $\boldsymbol{V}_{i-1}$,$\boldsymbol{V}_i$ respectively represent feature vectors of $\boldsymbol{P}_{i-1}$,$\boldsymbol{P}_i$):

$$\boldsymbol{P}_i(k_1) = \begin{Bmatrix} \boldsymbol{P}_i(k) \in \boldsymbol{P}_i(\boldsymbol{c}) : \\ \boldsymbol{V}_i(k) = \mathbf{argmin}(\|\boldsymbol{V}_i(k) - \boldsymbol{V}_{i-1}(j_1)\|) \\ \cap(\|\boldsymbol{V}_i(k) - \boldsymbol{V}_{i-1}(j_1)\| \le \boldsymbol{THR}) \end{Bmatrix}, \tag{7}$$

$$\boldsymbol{P}_i(\boldsymbol{c}) = \begin{Bmatrix} \boldsymbol{P}_i(c) : \boldsymbol{P}_i(c) \sqsubseteq \boldsymbol{P}_i \cap \\ (\boldsymbol{P}_i(c) \in Range(r)\ of\ \boldsymbol{P'}_{i-1}(j_1)) \end{Bmatrix}, \tag{8}$$

where $\boldsymbol{P}_i(\boldsymbol{c})$ represents all candidate points in the range of r; ***THR*** represents the distance threshold to match two feature vectors; ***argmin***$(\cdot)$ represents the feature vector that has the minimum distance.

Figure 4 illustrates the proposed matching algorithm. $P_{i-1}(1)$ represents an interest point found in $image_{i-1}$; $P'_{i-1}(1)$ represents the corresponding position of $P_{i-1}(1)$ in the coordinate system of $image_i$; r represents the small searching range which we will calculate subsequently. In the example shown in Fig. 4, the $P_{i-1}(1)$ has only two possible matching point candidates, $P_i(c_1)$ or $P_i(c_2)$, which means only 2 iterations of comparison are needed. Therefore, the computation complexity is reduced to $O(n)$.

Given the matching point from the algorithm in Fig. 4, the average Δx and Δy of all the matching pairs can be calculated, creating a more accurate translation model to feed back to the constellation prediction module. Furthermore, in a roll-to-roll manufacturing process, the vibration in the z axis is far more insignificant (< 387 nm) compared to the main x moving direction (mm scale), and the y moving direction (mm scale), which is caused by the misalignment of the roller. Readers can refer to the Appendix A for more details. Therefore, the accumulating errors in Eq. (5) can be resolved by the feedback process. Meanwhile, a global transformation model could be easily derived by the direct linear transformation (DLT) method.

Moreover, when the global transformation model is inadequate, usually caused by the local deformation of the images, the APAP stitching algorithm [14] can be adopted for the creation of panoramas. We will present both DLT and APAP experimental results in Section 4. Figure 5 shows the pseudo-code of the proposed registration algorithm. While the complexity of the proposed matching algorithm has been reduced to $O(n)$, its calculation can be further expedited for real-time applications. Instead of using loops for matching each element of $\boldsymbol{P}_i(\boldsymbol{K})$ with $\boldsymbol{P}_{i-1}(j)$ as shown in Fig. 5, vectorization of the feature matrices can be adopted for the sum of squared differences (SSD) calculation. Vectorization further enhances the SSD calculation from $O(n)$ to $O(1)$. The matching speed will be evaluated in Section 4. Readers can refer to Appendix B for more details of vectorization.

3.3 Setting the small searching range

In Section 3.2, we presented that our matching algorithm only searches for a small neighborhood range r around $\boldsymbol{P'}_{i-1}(\boldsymbol{j})$. In

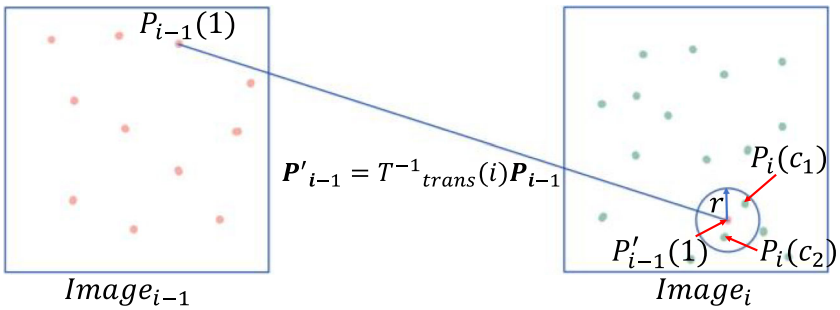


Fig. 4 An example of the proposed matching process

Fig. 5 Closed-loop feedback registration algorithm for consecutive image sequences

Algorithm 1: Closed-loop registration

Input: a sequence of images to be registered, $image_0$ to $image_k$, Δx between the first two images

Output: The sequence of matching points pairs $\{M_1, M_2, \cdots, M_k\}$, the stitched $panorama$

```
Initialization: Δy ← 0, i ← 1
while i ≤ k do
    detect and represent features V_{i-1}, P_{i-1} in image_{i-1} and V_i, P_i in image_i
    j ← 1
    while j ≤ length(V_{i-1}) do
        use equation (7) to find P'_{i-1}(j)
        use equation (8) to find P_i(c)
        Match each element of P_i(c) with P_{i-1}(j)
        if find a best match P_i(k) ∈ P_i(c) then
            add pair [P_{i-1}(j), P_i(k)] into M_i
        end
        j ← j + 1
    end
    save M_i
    update Δx and Δy
    i ← i + 1
end
create panorama based on {M_1, M_2, ..., M_k}
```

this section, we define the small range r in our roll-to-roll printing system, by analyzing the image deformations caused by the distortion of the flexible web, the web vibration, and the uncertainty of the web speed. The speed uncertainty in the moving direction plays the most important role in image deformation because our prior study in [25] demonstrated that the uncertainty of the web moving speed can be $\pm 10\% \cdot v$ which is much larger compared to the vibration and distortion of the web. Therefore, r should be defined mainly by the uncertainties in the moving direction (x direction in Fig. 1).

First, r should be easily calculated if the speed range of the moving web is known. For example, in [25], the linear motion speed is controlled by PID controllers with variations less than 10%, so our range r, according to Eq. (3), should be defined as,

$$r = \left| \frac{v(1 \pm 10\%) - v}{f \cdot d} \right| = \frac{\Delta x}{10}. \tag{9}$$

However, this range is calculated based on extremes. This range could be reduced in non-hypothetical scenarios. Assume the motion speed v presents with truncated normal distribution [26] and the corresponding probability density function (PDF) is shown in:

$$\psi(\mu, \sigma, a, b, x) = \begin{cases} \dfrac{\phi(\mu, \sigma^2)}{\Phi(\mu, \sigma^2; b) - \Phi(\mu, \sigma^2; a)} & if\ a < x < b \\ 0 & otherwise \end{cases}. \tag{10}$$

In Eq. (10), x is a random variable with mean μ, standard deviation σ, and lies within the interval $[a, b]$, with $-\infty \leq a < b \leq \infty$. For the application of our roll-to-roll printing system, the bounds a and b are set at $0.9v$ and $1.1v$ respectively, while μ and σ represent the mean and standard deviation of the motion speed$v$. In most engineering applications, covering 95% of the data should be sufficient, which means a smaller range r could be derived according to certain web speed statistical parameters μ, and σ. Second, if the speed range of the moving target is unknown while the initial movement Δx_1 between first and second images is known, r can be estimated experimentally. Third, if the initial movement Δx_1 is also unknown, the first image pair of the image sequence must be registered using any of the classical registration methods to get the prior registration result of the image sequence. Then, Δx_1 can be calculated from this initial image pair, and r can be estimated experimentally as the second scenario.

4 Experimental results

In this section, we evaluate the performance of our closed-loop registration algorithm using real-world data. The detailed experiments based on the synthetic data are shown in Appendix C. All these registration algorithms are evaluated in the MATLAB 2020a and Python 3.7 environment with Intel(R) Core (TM) i9-10980HK CPU @ 2.4 GHz, RTX 2080 Super with Max-Q Design GPU laptop. VLFeat [27] library, the source codes from SuperGlue [21], LoFTR [22],

SuperPoint [16], and MAGSAC++ [19] are comparatively evaluated in the experiments.

4.1 Experimental setup for real-world moving flexible targets

To verify our proposed image registration algorithm in a real-world scenario, we set up experiments on a roll-to-roll print system (see Fig. 1). The motion of the web along the x-axis can be controlled. A Pixelink PL-D721 autofocus camera with a 16 mm autofocus liquid lens was set up to inspect the circuits printed on the web. The printed patterns on the moving web are repetitive copper circuits. The motion speed of the web was set to 0.0127 m/s (0.5 in./s). Since there are only minor vibrations in the z-axis compared to the motion in the x- axis (refer to Appendix A), we set a small autofocus range between 38,000–40,000 dB in Pixelink Capture software to speed up the autofocus processing time [28]. Then, the focused images were captured at the frame rate of 2 frames per second (fps) with an image size of 640 × 1024 (pixels)

Two sequences of images are acquired by the Pixelink PL-D721 autofocus camera for the evaluation of the registration algorithms in the experiments. We called the datasets "R2R_Autofocus1" and "R2R_Autofocus2". R2R_Auto-focus1 contains an entire circuit module. Each image of the dataset only contains a part of the module. R2R_Autofocus2 contains many duplicated circuit modules. Each image of the dataset can contain more than one module (see Figs. 6 and 8).

4.2 Algorithm comparisons on real-world moving flexible targets

The performance evaluation experiments include the comparison of SIFT and SuperPoint feature detectors and descriptors; RANSAC, and MAGSAC++ outlier removal methods; the state-of-the-art ISIFT, SuperGlue, and LoFTR registration algorithms, and our proposed algorithm.

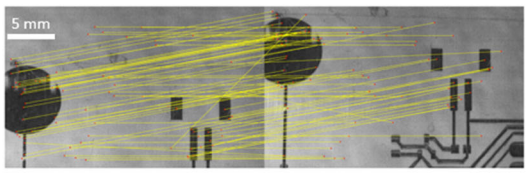
First, given a pair of images, the VLFeat library [27] is used to detect and describe SIFT interest points. The pre-trained SuperPoint model is implemented to detect and describe the SuperPoint interest points. Then, these two kinds of interest points are matched in different ways. For SIFT, we deployed the lowest SSD method, lowest SSD plus outlier removal methods, ISIFT, and the proposed algorithm respectively. For SuperPoint, we deployed the lowest SSD method, lowest SSD plus outlier removal methods, and the proposed algorithm respectively. We also implemented the end-to-end registration algorithms including LoFTR, LoFTR with outlier removal methods, and SuperGlue. We used the pre-trained models provided by their authors for all the deep-learning-based methods.

Figure 6 shows an example of the matching results of an image pair in the R2R_Autofocus1 dataset using different methods. More matching examples for the R2R_Autofocus2 dataset are shown in Appendix D. For the SIFT feature detector and descriptor, Fig. 6a has many false positives because the image pair has many similar patterns spanning the entirety of the whole image. Figure 6b, c and d remove the false positives but also remove some of the true positives. Furthermore, the results of ISIFT are not consistent in each experiment because of the iterative RANSAC process. Figure 6e finds more matching pairs without any false positives. For the SuperPoint feature detector and descriptor, all the matching methods perform better than the SIFT feature detector and descriptor. However, our proposed algorithm finds denser matching pairs which are beneficial to image stitching when there exist more local deformations, such as the roll-to-roll printing process on the flexible substrate. For the end-to-end matching algorithms, LoFTR finds the greatest quantity of matching pairs. However, the number of false positives is greater than the true positives, leading to the failure of both the RANSAC and MAGSAG++ outlier removal methods. SuperGlue finds the similar quantity and accurate matching pairs as our proposed algorithm. Nevertheless, it costs much more computation time when GPU is unavailable. Readers can refer to Section 4.3 and Table 4 for the evaluation of running time. Table 1 shows the numbers of the average matching point pairs using different registration methods in the R2R_Autofocus1 and R2R_Autofocus2 datasets. We use SSD to represent the lowest SSD method in Table 1. Our proposed method finds about twice as many matching point pairs as other outlier removal methods with a lower root mean squared error (RMSE) as shown in Table 3. Therefore, the results in Table 1 are consistent with the results in Fig. 6.

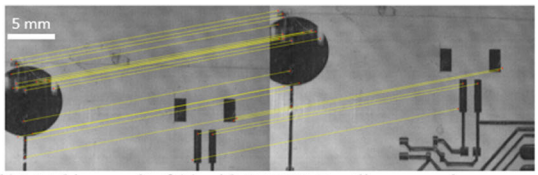
Moreover, in the last two pairs of the consecutive images in R2R_Autofocus1, we found the matching results from SIFT with RANSAC and ISIFT are extremely inaccurate, as shown in (b), (c), (f), and (g) in Fig. 7. The reason is the same as LoFTR - there are too many false positive matches in the SIFT with the lowest SSD registration method in these two image pairs. RANSAC removes the correct matching point pairs as outliers because it tends to choose more quantities of the matching point pairs, which are false-positive matches in the scenarios in Fig. 7. Furthermore, the ISIFT algorithm iteratively refines the RANSAC results based on the maxima of the mutual information between the image pairs, meaning it encounters the same problem as RANSAC. We perform 10 iterations of ISIFT experiments for matching the two image pairs in Fig. 7 to try to eliminate the random nature of ISIFT as much as possible. Nevertheless, 9/10 of the experiments fail to find the correct matching point pairs in Fig. 7. By contrast, as shown in Fig. 7d and h, our proposed method finds the correct matching point pairs in both image pairs.

Table 2 shows the initial, average, and standard deviation of Δx and Δy of the coarse translation models in both datasets. Using Eqs. (3) and (4), we get the initial $\Delta x = -200$, $\Delta y = 0$ pixels

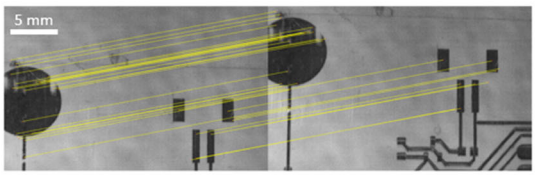
SIFT detector and descriptor



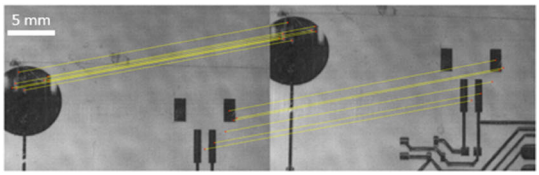
(a) Matching result of SIFT detector and descriptor with lowest SSD method



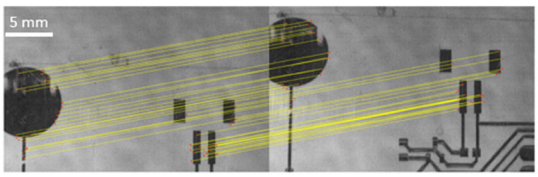
(b) Matching result of (a) with RANSAC outlier removal



(c) Matching result of (a) with MAGSAC++ outlier removal

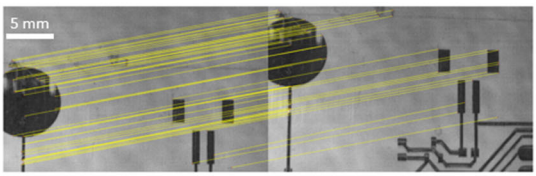


(d) Matching result with ISIFT

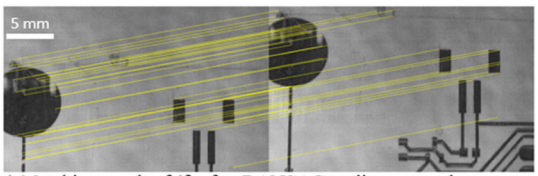


(e) Matching result of our proposed algorithm using the same SIFT detector and descriptor

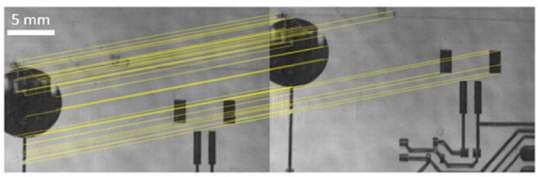
SuperPoint detector and descriptor



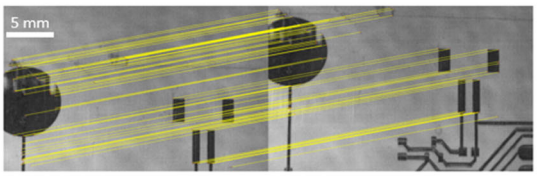
(f) Matching result of feature vectors acquired by SuperPoint detector and descriptor with lowest SSD method



(g) Matching result of (f) after RANSAC outlier removal

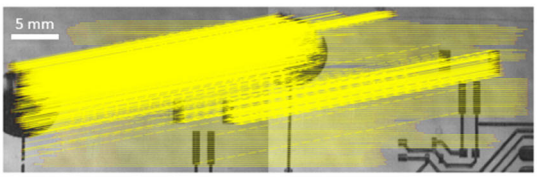


(h) Matching result of (f) after MAGSAC++ outlier removal

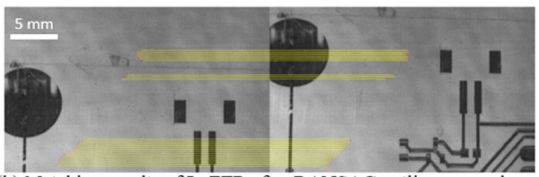


(i) Matching result of our proposed algorithm using the same SuperPoint detector and descriptor

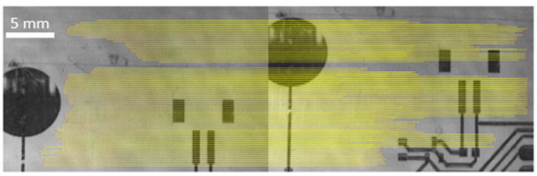
End-to-end matching methods



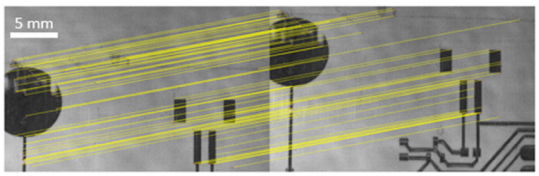
(j) Matching results of LoFTR



(k) Matching results of LoFTR after RANSAC outlier removal



(l) Matching results of LoFTR after MAGSAC++ outlier removal



(m) Matching results of SuperGlue

Fig. 6 Matching examples in different methods. Dataset: R2R_Autofocus1; Methods: SIFT detector and descriptor, SuperPoint detector and descriptor, End-to-end matching methods

Table 1 Number of matching pairs comparisons of different algorithms

datasets	SIFT&SSD	SIFT&SSD& RANSAC	SIFT&SSD& MAGSAC++	ISIFT	SIFT&Ours
R2R_Autofocus1	250.4	126.0	109.3	33.2	268.9
R2R_Autofocus2	127.9	53.0	43.6	14.9	92.2
	SuperPoint&SSD	SuperPoint&SSD& RANSAC	SuperPoint&SSD& MAGSAC++	SuperPoint&Ours	
R2R_Autofocus1	149.2	121.6	72.8	199.0	
R2R_Autofocus2	149.8	107.0	73.9	351.1	
	LoFTR	LoFTR&RANSAC	LoFTR& MAGSAC++	SuperGlue	
R2R_Autofocus1	2390.8	606.1	1970.6	203.7	
R2R_Autofocus2	1341.7	261.6	901.1	361.5	

in the R2R_Autofocus1 dataset and Δx =200, Δy=0 pixels in the R2R_Autofocus2 dataset. Then, the updated Δx and Δy continue to feed back to the next consecutive registration tasks. Using Eqs. (9), (10), and the standard deviation of the web moving speed in [25] (refer to Fig. 1), we get the searching range $r = 18$ pixels as default in the experiments. Then according to Table 2, we find that the speed of the web is more stable than expected. Therefore, we further adjusted the small searching range to 10 pixels to accelerate the algorithm for real-time inspection.

Figure 8 shows the panoramas of R2R_Autofocus1. In the experiments, the SIFT plus lowest SSD with RANSAC and ISIFT methods have similar matching results, but the SIFT plus lowest SSD with RANSAC method is faster than the ISIFT method due to the exclusion of the iterative RANSAC process. Therefore, to evaluate the stitching performance, we employ both DLT and APAP stitching algorithms based on the matching results from the SIFT plus lowest SSD with the RANSAC method and the proposed method. Figure 8a shows the panorama stitching results using the DLT stitching algorithm. SIFT plus lowest SSD with RANSAC matching results are shown on the left side of Fig. 8a. The proposed matching results are shown on the right side of Fig. 8a. Figure 8b shows the panorama stitching results using the APAP stitching algorithm. SIFT plus lowest SSD with RANSAC matching results

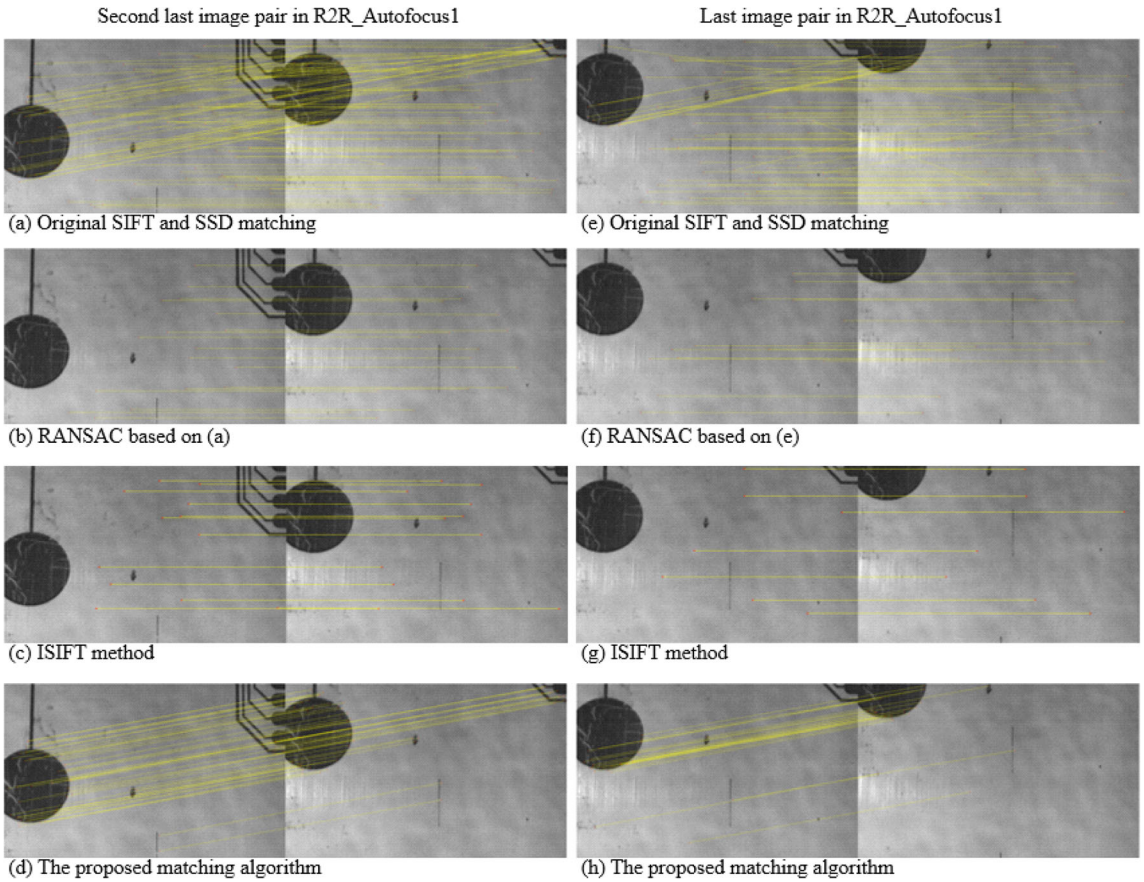


Fig. 7 Illustration of the matching results of the last two image pairs in R2R_Autofocus1

Table 2 Δx and Δy in closed-loop feedback matching experiments

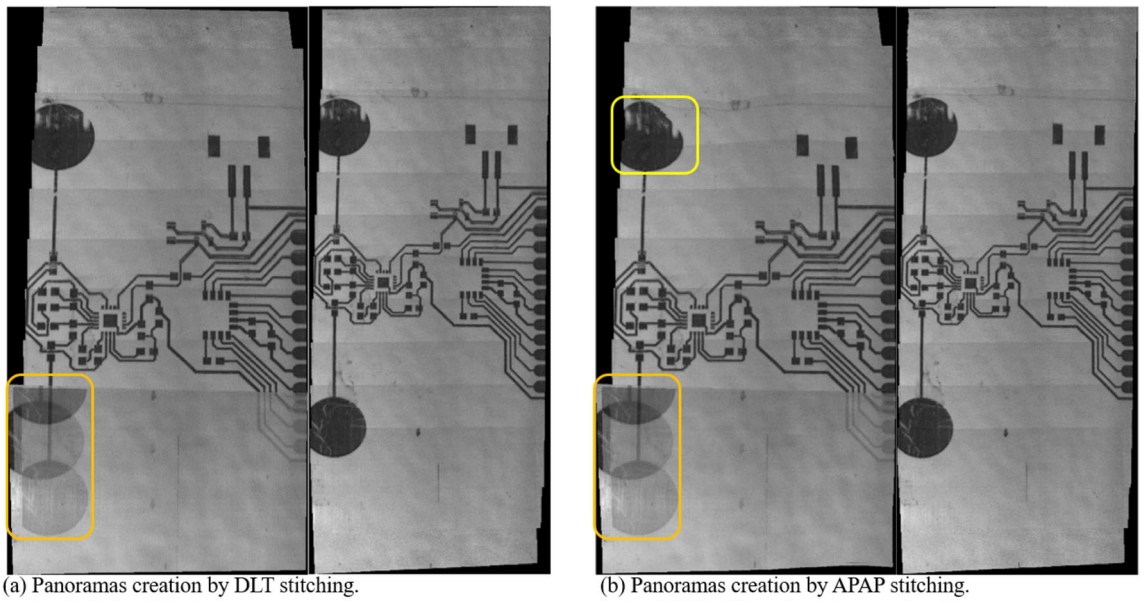
datasets	Initial Δx Δy (pixels)	Average Δx Δy (pixels)	STD Δx Δy (pixels)
R2R_Autofocus1	$\Delta x=-200$ $\Delta y=0$	$Mean(\Delta x)=-191.07$ $Mean(\Delta y)=4.00$	$STD(\Delta x)=3.42$ $STD(\Delta y)=1.66$
R2R_Autofocus2	$\Delta x=200$ $\Delta y=0$	$Mean(\Delta x)=196.49$ $Mean(\Delta y)=-3.81$	$STD(\Delta x)=2.09$ $STD(\Delta y)=0.73$

are shown on the left side of Fig. 8b. The proposed matching results are shown on the right side of Fig. 8b. Bundle adjustment [29] is applied during stitching. In each of the matching results on the left side of both Fig. 8a and b, the matching correspondences on the bottom section (orange regions in Fig. 8) have more misalignments and ghosting effects because of the mismatches in the registration processes as shown in Fig. 7. Incorrect global projective transformation matrices are created during DLT stitching, resulting in the ghosting effects seen on the left side of Fig. 8a and b. Additionally, despite its attempt to stitch the image pairs both locally and globally, the APAP stitching still cannot resolve the problem of producing wildly inaccurate global projective transformation matrices.

Furthermore, in Fig. 8b, the top-left region highlighted in yellow has an obvious deformation when the APAP stitching method is implemented. The deformation comes from the local transformation matrix that the APAP algorithm finds to satisfy the matching points around this top-left region. The reason is that the APAP stitching algorithm tries to find many local transformation matrices according to the local matching correspondence results in certain regions. If an arbitrary region has too many false-positive matching point pairs, the local transformation matrix of this region will be inaccurate, causing deformations during stitching.

However, on the right side of both Fig. 8a and b, the stitching results are much better when compared to the left. The panoramas on the right side lack any ghosting effects or obvious deformations. The results prove that the proposed registration algorithm finds more accurate matching correspondences than the SIFT plus lowest SSD with the RANSAC method.

To further quantify the matching accuracy, we compute the RMSEs of different registration methods. To make a fair comparison, we only choose the image pairs that have more than 30 matched points and are not completely inaccurate matches. In DLT stitching, the global projective homographic transformation $T_{proj}(k)$ is implemented on all the matching points in $image_k$. Similarly, for the APAP stitching method, the local projective homographic transformation h_* [14] is implemented. Table 3 shows the RMSE results for the chosen image pairs in the R2R_Autofocus1 dataset. On average, the proposed matching results have 28% less RMSE than the SIFT plus lowest SSD with the RANSAC method, 9% less RMSE than SuperPoint with RANSAC method, and 5% less RMSE



(a) Panoramas creation by DLT stitching.
Left: SIFT&RANSAC; right: our algorithm.

(b) Panoramas creation by APAP stitching.
Left: SIFT&RANSAC; right: our algorithm.

Fig. 8 The panoramas of R2R_Autofocus1 created by SIFT with RANSAC and the proposed algorithm

Table 3 RMSE of the matching results

R2R_Autofocus1	SIFT + Lowest SSD+ RANSAC + DLT	SIFT + Ours + DLT	SIFT + Lowest SSD+ RANSAC + APAP	SIFT + Ours + APAP	SuperPoint + Lowest SSD+RANSAC + DLT	SuperPoint + Ours + DLT	SuperGlue + DLT
3–4	3.8358	3.1808	3.8640	3.1045	2.9049	2.8615	3.0416
4–5	3.7027	2.5773	3.6501	2.5767	3.9486	3.8304	3.7857
5–6	2.4984	2.4012	2.5543	2.5368	4.4701	4.2202	4.2298
6–7	2.7962	2.0037	2.8162	2.0343	4.1590	3.8247	3.9592
7–8	4.4075	2.2133	4.4666	2.1878	3.8258	2.8206	3.4080

than the SuperGlue method, which proves our closed-loop registration algorithm performs better in these experiments.

4.3 Running time evaluation on real-world moving flexible targets

In addition to accuracy, the fast speed of the registration method is essential for the real-time inspection and monitoring of manufacturing processes. In Section 3.2 and Appendix B, we propose a vectorization method to expedite the proposed matching algorithm. To evaluate the efficiency of the proposed algorithm, different methods are implemented for the registration of both R2R_Autofocus1 and R2R_Autofocus2 datasets for comparison. Table 4 shows the average processing time per frame in different matching algorithms with the conventional local feature detectors and descriptors, as well as the deep-learning-based SuperPoint feature detector and descriptor. To make fair comparisons, we extracted and saved SuperPoint results as "mat" files of all images from the Python source code, and matched the image pairs in the same MATLAB environment as SIFT.

For the conventional local feature detectors and descriptors, the average processing time of the non-vectorization and the vectorization implementations of each method is demonstrated on the left side and right side of the top part of Table 4, respectively. The non-vectorization results on the left side indicate the total floating-point operations (FLOPs) of each method. Our closed-loop feedback matching algorithm has the least FLOPs in all three detectors and descriptors. Particularly in the ORB detector and descriptor, the speed of our algorithm is five times faster than the lowest SSD matching method, and six times faster than the lowest SSD plus the RANSAC outlier removal method. This is because the ORB detector finds far more interest points than SIFT and SURF while our proposed algorithm removes many of these points as outliers efficiently. The vectorization results show that all the registration methods are expedited from vectorization. In the SIFT and SURF detectors and

Table 4 Average running time comparisons of different matching algorithms

Running time of conventional detectors and descriptors (seconds)				
	Without vectorization		With vectorization	
	R2R_Autofocus1	R2R_Autofocus2	R2R_Autofocus1	R2R_Autofocus2
SIFT + Lowest SSD	1.58	1.32	0.26	0.22
SIFT + Lowest SSD+RANSAC	2.21	1.51	0.58	0.51
SIFT + Ours	0.77	0.50	0.25	0.22
SURF + Lowest SSD	0.41	0.38	0.07	0.07
SURF + Lowest SSD+RANSAC	1.16	0.81	0.52	0.45
SURF + Ours	0.26	0.22	0.07	0.07
ORB + Lowest SSD	25.82	29.97	0.65	1.01
ORB + Lowest SSD+RANSAC	27.01	34.64	1.57	2.46
ORB + Ours	5.87	6.82	0.62	0.88
ISIFT	143.69	66.85	143.83	63.37
Running time of deep-learning-based detectors and descriptors (seconds)				
	R2R_Autofocus1		R 2R_Autofocus2	
	CPU	GPU	CPU	GPU
SuperPoint Extraction	3.05	0.10	3.04	0.10
SuperPoint + Lowest SSD	3.05	0.11	3.04	0.11
SuperPoint + SSD+RANSAC	3.52	0.57	3.53	0.59
SuperPoint + Ours	3.05	0.11	3.04	0.11
LoFTR	12.71	0.37	11.38	0.33
SuperGlue	3.52	0.13	3.85	0.13

descriptors, our algorithm is as fast as the lowest SSD matching method, yet much more accurate. In the ORB detector and descriptor, our algorithm is even faster than the lowest SSD matching method due to a large number of interest points. Furthermore, the time cost of the RANSAC outlier removal process dominates the total running time in the vectorization implementation because RANSAC cannot be expedited with vectorization. The ISIFT results show that the time cost from ISIFT is lengthy both with vectorization and without vectorization. The reason for this is that the iterative RANSAC process dominates the total running time in ISIFT.

For SuperPoint feature detectors and descriptors, the average processing time using CPU and GPU of each method is demonstrated in the bottom part of Table 4, respectively. The SuperPoint extraction time in the first line represents the time cost of finding all the interest points in each image pair using the pre-trained SuperPoint model. In the bottom part of Table 4, all the methods except LoFTR are based on the interest points found by SuperPoint. Our proposed method still performs the fastest, as well as the lowest SSD method. Furthermore, the computation time using CPU is much slower than using GPU. Therefore, for the deep-learning-based feature detector and descriptor, parallel computing hardware is needed for real-time applications. Overall, the experimental running time results show that the proposed algorithm is more efficient than the other registration algorithms.

5 Conclusion

In this study, a closed-loop feedback image registration algorithm is proposed. The algorithm involves two modules, namely the constellation prediction module and the PPM module. In the first module, a coarse translation transformation model is created using the spatiotemporal information of the consecutive image sequence. In the second module, the coarse translation transformation model is optimized to obtain an accurate point-to-point matching result. The accurate matching results keep feeding back into the first module to fine-tune the next coarse translation transformation model. By taking advantage of the spatiotemporal information of the consecutive image sequence, the proposed image registration algorithm demonstrates higher speed and accuracy when the conventional local feature detectors and descriptors such as SIFT, SURF, and ORB are applied. It finds about 100% more matching point pairs with a lower RMSE, and reduces up to 86.5% of the running time compared to other state-of-the-art outlier removal algorithms. The proposed image registration algorithm achieves higher speed when the deep-learning-based feature detector and descriptor SuperPoint are applied. It finds similar number of matching point pairs with a lower RMSE compared to the end-to-end registration algorithm SuperGlue. The end-to-end registration algorithm LoFTR finds more matching point pairs than the proposed algorithm, however, most of them are false positives. The experiments on both simulation and real-world data show promising results for the registration of moving flexible targets. Our algorithm can accurately match more keypoint correspondences and achieve a better panorama than other registration algorithms. Additionally, the running complexity of the keypoint matching process is reduced from $O(n^2)$ to $O(n)$ and further accelerated up to 0.07 seconds per image pair using vectorization in MATLAB 2020a environment. Overall, our proposed closed-loop feedback registration algorithm achieves better accuracy and efficiency compared to other state-of-the-art algorithms.

The proposed algorithm has the following limitations. First, the algorithm requires accurate prior knowledge, e.g., the coarse movement between each image pair. Any wrong prior knowledge will badly affect the performance of the algorithm. Second, we only deploy the translation coarse model for the prior registration. There are different application scenarios of the coarse models. We leave it to our future work.

6 Future work

We will extend our prior registration model from the unidirectional transformation model to translation in both x and y directions, scaling, rotation, and mixed transformation models. Additionally, the proposed visual inspection system can be applied to the control of the roll-to-roll flexible electronics printing system. The speed and position of the moving web can be adjusted by the feedback data from the registration results to guarantee the quality of the products on the manufacturing line.

The Pixelink PL-D721 autofocus camera with a 16 mm autofocus liquid lens can only get as fast as 2 frames per second which is the speed bottleneck of the registration process. Advanced auto-focus algorithms [30] will be included in the future for real-time inline inspection.

Appendix 1: The vibration measurement in the z-direction of the roll-to-roll printing system

Figure 9 shows the setup of the roll-to-roll printing system in our lab. The displacement sensor is installed for inspecting the distance between the sensor and the web. Figure 10 shows the 80-second data acquired by the sensor. The maximum displacement in the z-direction is only 387 nm which can be neglected compared to the mm-scale displacements in the x and y directions.

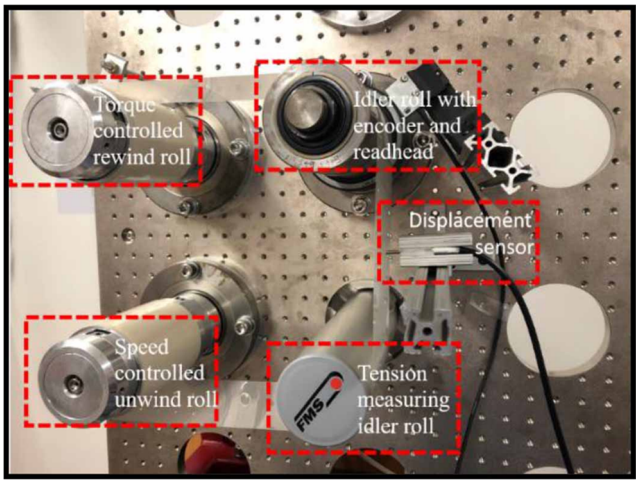


Fig. 9 Roll-to-roll printing system setup

Appendix 2: Expedition of matching algorithm for real-time applications

While the complexity of the proposed matching algorithm has been improved up to $O(n)$ in Section 3.2, its calculation can be further expedited for real-time applications. Instead of using loops for matching each element of $\boldsymbol{P}_i(\boldsymbol{K})$ with $\boldsymbol{P}_{i-1}(j)$ as shown in Fig. 11a and b, vectorized feature matrices multiplication can be adopted for the SSD calculation.

The SSD of two vectors, e.g., $\boldsymbol{V}_{i-1}(j)$ and $\boldsymbol{V}_i(k)$ can be calculated by:

$$\boldsymbol{SSD}_{jk} = \sum(\boldsymbol{V}_{i-1}(j) - \boldsymbol{V}_i(k))^{\boldsymbol{2}}, \tag{11}$$

where $\sum$is the summation operator. However, calculating $\boldsymbol{SSD}_{jk}$ for all combinational vector pairs in series using loops will be time-consuming. In the following, we attempt to parallelize the calculation by vectorization. Mathematically, Eq. (11) can be expressed as:

$$\mathbf{SSD}_{jk} = \sum \mathbf{V}_{i-1}(j)^2 + \sum \mathbf{V}_i(k)^2 - 2 \cdot \mathbf{V}_{i-1}(j) \times \mathbf{V}_i(k)^T, \tag{12}$$

where $\boldsymbol{V}_i(k)^T$ denotes the transpose of $\boldsymbol{V}_i(k)$. Expanding Eq. (12) from one-to-one SSD into $m \times n$ SSDs (meaning there are m features in $\boldsymbol{V}_{i-1}$ and n features in $\boldsymbol{V}_i$),we can calculate all the SSDs between $\boldsymbol{V}_{i-1}$ and $\boldsymbol{V}_i$ (by)

$$\boldsymbol{SSD} = \sum_{rows} \boldsymbol{V}_{i-1}{}^2 \times [1\ 1 \cdots 1]_{1\times n} + [1\ 1 \cdots 1]_{1\times m}{}^T \times \sum_{columns} \left(\boldsymbol{V}_i{}^{\boldsymbol{T}}\right)^{\boldsymbol{2}} - 2 \cdot \boldsymbol{V}_{i-1} \times \boldsymbol{V}_i{}^T, \tag{13}$$

where the element at the j^{th} row and k^{th} column of $\boldsymbol{SSD}$ can be denoted by $\boldsymbol{SSD}_{jk}$. The total FLOPs of Eqs. (11) and (13) are similar. However, the SSD calculation time will be significantly reduced after applying the vectorization in Eq. (13), because by taking advantage of the parallel computing capability of multi-core CPUs or GPUs, matrix multiplication using Eq. (13) is inherently faster than the computation in sequential loops using Eq. (11). Such a vectorization calculation can improve the speed of $\boldsymbol{SSD}$ calculation from $O(n)$ to $O(1)$. The running speed was evaluated in Section 4. Figure 11 shows the comparison of the two methods. Figure 11a shows the loop implementation of the feature matching algorithm. Figure 11b shows the $\boldsymbol{SSD}$ calculation process of (a). Figure 11c shows the vectorization implementation of the feature matching algorithm. Figure 11d shows the $\boldsymbol{SSD}$ calculation process of (c). As an example, the dimensions of all the feature vectors are set to 1×128.

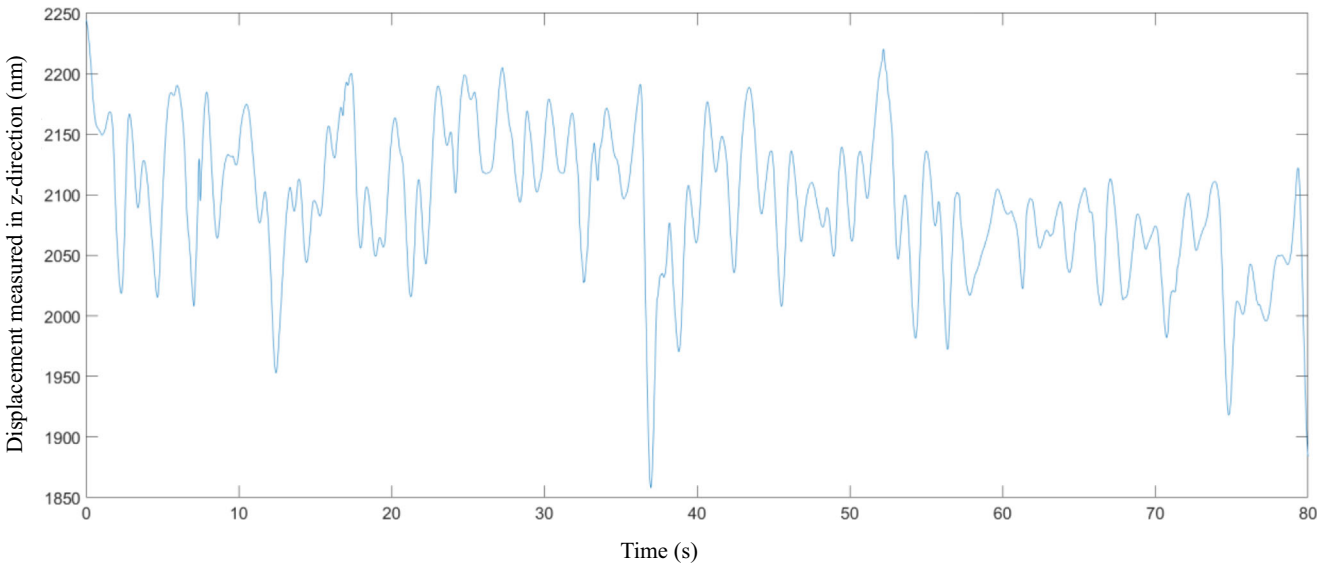


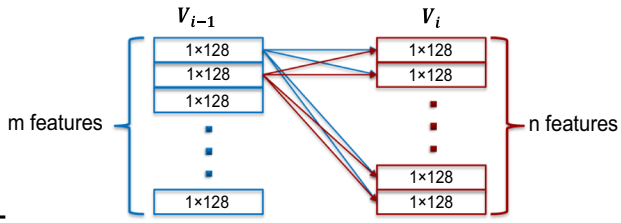
Fig. 10 The measurement of displacement in z-direction in the roll-to-roll printing system

Algorithm 2: Loop Feature Matching Algorithm

Input: V_{i-1}, P_{i-1} from $image_{i-1}$ and V_i, P_i from $image_i$
Output: all the matching pairs M_i

Initialization: $l \leftarrow length(V_{i-1})$, $j \leftarrow 1$ $M_i \leftarrow empty$
while $j \leq l$ **do**
  // $V_i(c)$ comes from $P_i(c)$
  calculate D_j, the sum of absolute differences (SSD) between $V_{i-1}(j)$ and $V_i(c)$
  find $D_j(k)$, the minimum SSD of D_j
  if $D_j(k) \leq threshold$ **then**
    add pair $[P_{i-1}(j), P_i(k)]$ into M_i
  end
end

(a) Loop feature matching method



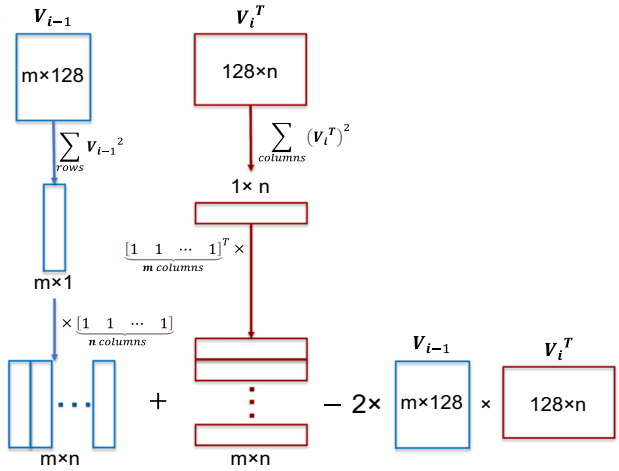
(b) SSD calculation process of loop matching

Algorithm 3: Vectorization Feature Matching Algorithm

Input: V_{i-1}, P_{i-1} in $image_{i-1}$ and V_i, P_i in $image_i$
Output: all the matching pairs M_i

Initialization: $l \leftarrow length(V_{i-1})$, $j \leftarrow 1$ $M_i \leftarrow empty$
while $j \leq l$ **do**
  find $P_i(c)$ for each $P_{i-1}(j)$
  save it into $[j, c]$
end
implement equation (11) to find SSD
// for $(V_i^T)^2$ and $V_{i-1} \times V_i^T$ parts, only calculate related values based on each c calculated above, set other values into NaN
find SSD_j^{min}, the minimum values in each row of SSD
while $j \leq l$ **do**
  if $SSD_j^{min} \leq threshold$ **then**
    $k = index(SSD_j^{min})$
    add pair $[P_{i-1}(j), P_i(k)]$ into M_i
  end
end

(c) Vectorization feature matching method



(d) SSD calculation process of vectorization matching

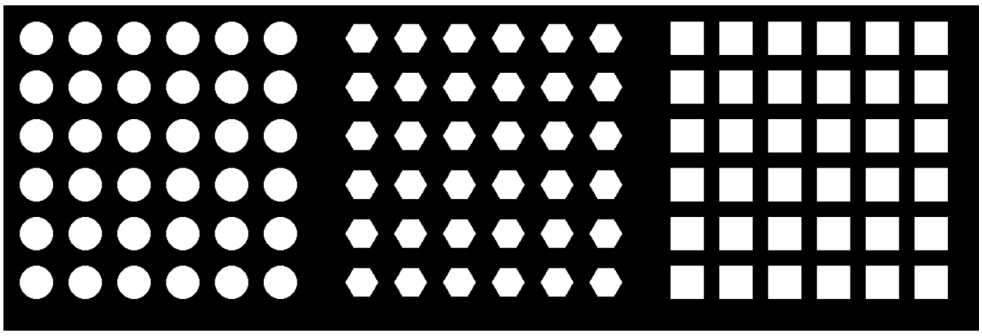
Fig. 11 Comparisons of the loop and the vectorization matching implementations, (**a**) Loop feature matching method, (**b**) SSD calculation process of loop matching, (**c**) Vectorization feature matching method, (**d**) SSD calculation process of vectorization matching

Appendix 3: More experimental results on simulation data

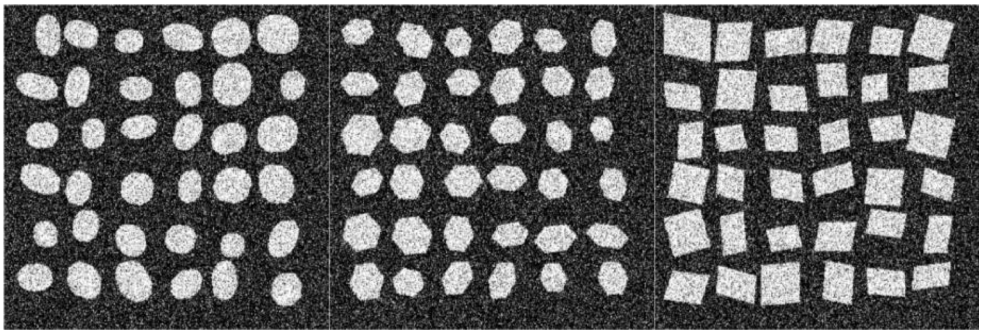
Simulation data experiments

In this sub-section, we evaluate the performance of our closed-loop feedback registration algorithm with synthetic images. Consecutive repetitive and regular printed patterns are commonly encountered in real-world flexible electronics manufacturing due to the nature of printing, e.g., roll-to-roll printing. To simulate the images in the real-world production line of such repetitive and regular grid-based patterns, three synthetic image datasets are created. Each of the three datasets contains twenty images with a size of 400 × 400 pixels with their own unique basic grid patterns. The first dataset consists of squares with a side length of 41 pixels, the second dataset consists of circles with a radius of 20 pixels, and the last dataset consists of hexagons with a radius of 20 pixels. All these patterns are deformed randomly by affine transformation locally. Meanwhile, each image is corrupted by Gaussian-blurring-filtering with a mask size of 7 × 7, salt and pepper noise of density of 0.05, and white Gaussian noise of variance of 0.02. We use a distortion factor to define the level of local affine deformation, i.e., the greater the distortion factor is, the broader the random translation, rotation, scaling, and shear ranges are (readers can check the source code in [31] for more details). Figure 12 shows an example of the three datasets. For each image in the three synthetic datasets, the center positions of all the deformed patterns are saved as the ground truth when the image is created. To simulate the production lines in the real-world scenario, we set the movement between every two consecutive images (see the red arrows in Fig. 14). The moving speed of each dataset is $dx = dy = 60$ *pixels/frame*, i.e., the distance between every two undistorted pattern's centers.

To evaluate the performance of the proposed algorithm, three feature-based algorithms are implemented. Each of the SIFT, SURF, and ORB detectors and descriptors are used on a sequence of grid-based images to detect and describe the



(a) The original grid-based images



(b) The deformed grid-based images with distortion fact 3.0

Fig. 12 An example of the three synthetic datasets. (**a**) without deformation. (**b**) with deformation, distortion factor = 3.0

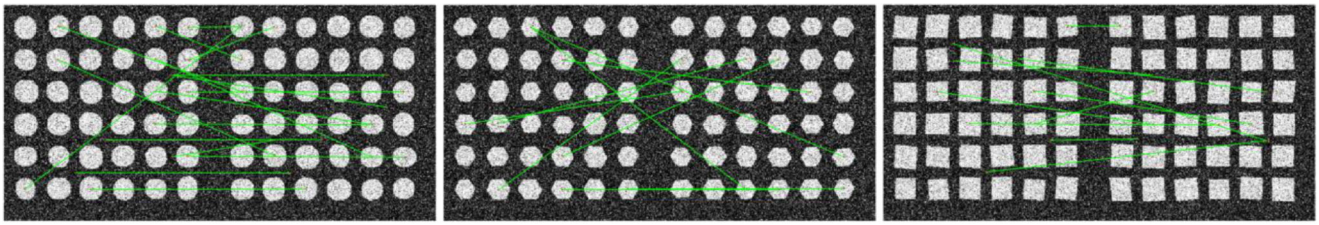
interest points. Then, these interest points are matched through the four methods described below. In the first matching method, we implement the lowest SSD method directly. In the second method, the RANSAC outlier removal algorithm is applied to optimize the matching results in the first method. In the third method (if SIFT detector and descriptor are adopted), a new variant of SIFT, ISIFT [20] is implemented. ISIFT iteratively optimizes the results of RANSAC based on mutual information and shows higher registration accuracy than SIFT in remote sensing images. In the fourth method, the proposed closed-loop feedback matching method is implemented. An example of the matching results is shown in Fig. 13. In Fig. 13, all of the results are based on SIFT detector and descriptor. The results of SURF and ORB detectors and descriptors are shown in next sub-section. From Fig. 13a-c, we can see both the traditional and the state-of-the-art feature-based matching algorithms failed to find the correct correspondences between two grid-based images. The reason is that these algorithms ignore the smooth motion of the moving target and continuous spatiotemporal information in the consecutive image sequence, and thus cannot discriminate similar grid patterns. However, as shown in Fig. 13d, taking advantage of the prior registration results, our proposed closed-loop feedback matching algorithm has better accuracy and abundancy in the matching correspondences.

To further quantify the matching accuracy of the proposed algorithm, we first use the average Euclidean distances between the detected centers (we define the detected centers as the correspondences located inside the range of 5 pixels of the ground-truth centers) and the ground-truth centers to measure the detected center errors. Then, we also define the detected center ratio, which is the ratio of the number of the detected centers to the number of the ground-truth correspondences. Based on the moving speed ($dx = dy = 60$ *pixels/frame* in our experiment), the ground-truth centers of two consecutive images can be easily matched which can be defined as the ground-truth correspondences. Finally, to quantify the accuracy of all the detected matching correspondences (centers and non-centers), we compute the respective RMSE using:

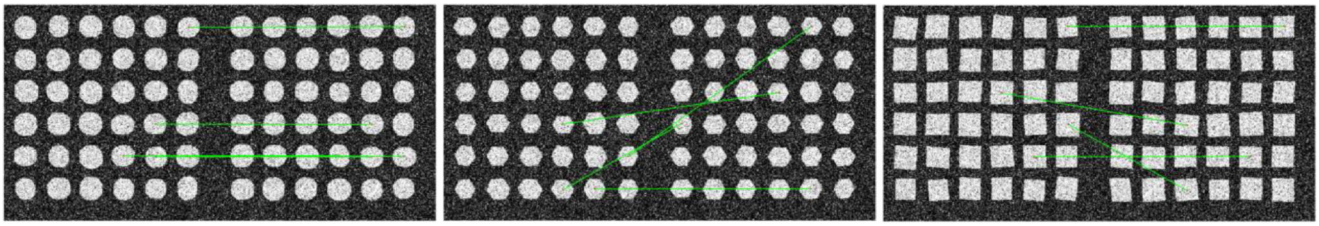
$$RMSE(k) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\| T_{proj}(k)\cdot P_k(i) - P_{k-1}(i)\right\|^2}, \qquad (14)$$

where $P_{k-1}(i)$ and $P_k(i)$ represent the coordinates of the i^{th} matching point pair in $image_{k-1}$ and $image_k$. $T_{proj}(k)$ can be calculated using the ground-truth correspondences and the DLT method. Using Eq. (14), we can transform the matching points in $image_k$ into the coordinate system of $image_{k-1}$ and then calculate the average L^2 distance of all the matching pairs in the coordinates of $image_{k-1}$. Figure 14a shows an example of the ground-truth correspondences. Figure 14b shows the matching result of the same image pair using the SURF detector and descriptor with our closed-loop feedback matching algorithm. The small black circles are the detected pattern centers of this image pair.

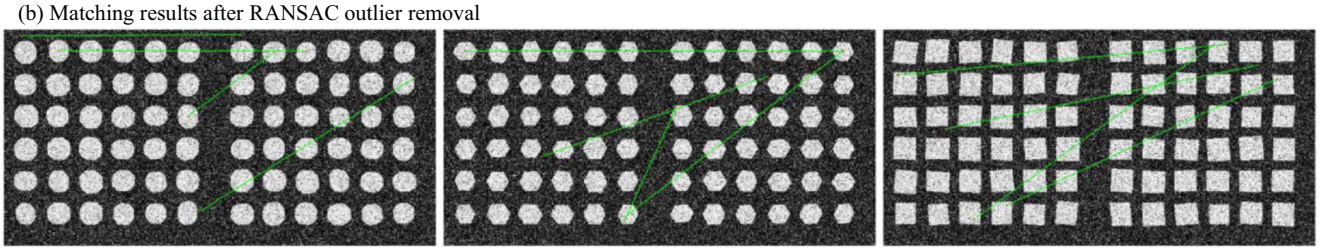
Figure 15 shows the quantitative errors from all three synthetic datasets. The horizontal axes are the distortion factors, the vertical axes on the left (blue color) are the RMSE errors,



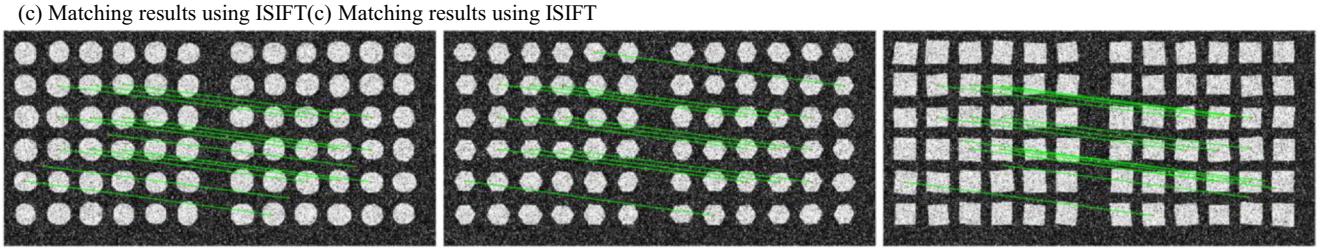
(a) Matching results of feature vectors acquired by SIFT detector and descriptor



(b) Matching results after RANSAC outlier removal



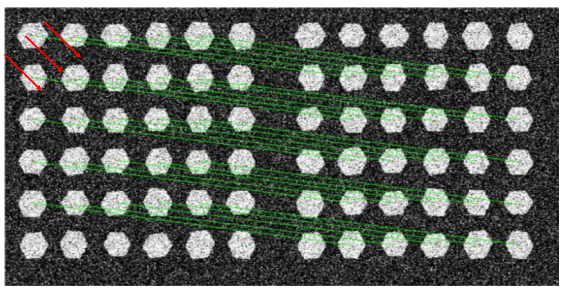
(c) Matching results using ISIFT(c) Matching results using ISIFT



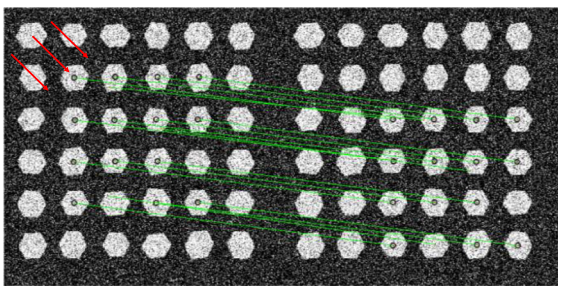
(d) Matching results of our proposed algorithm using the same SIFT detector and descriptor

Fig. 13 Comparisons of matching results based on SIFT detector and descriptor (**a**) Matching with lowest SSD (**b**) optimizing (a) with RANSAC outlier removal algorithm (**c**) Matching with ISIFT algorithm (**d**) Matching with the proposed closed-loop feedback matching algorithm

and the vertical axes on the right (red color) are the detected center ratios. Given different distortion factors, the blue lines are the average detected center errors, the green lines are the average RMSE of all the detected correspondences, and the red lines are the average detected center ratios. Figure 15a shows the errors using the SIFT detector and descriptor and our matching algorithm, Fig. 15b shows the errors using SURF detector and descriptor and our matching algorithm, and Fig. 15c shows the errors using the ORB detector and descriptor and our matching algorithm. The detected center

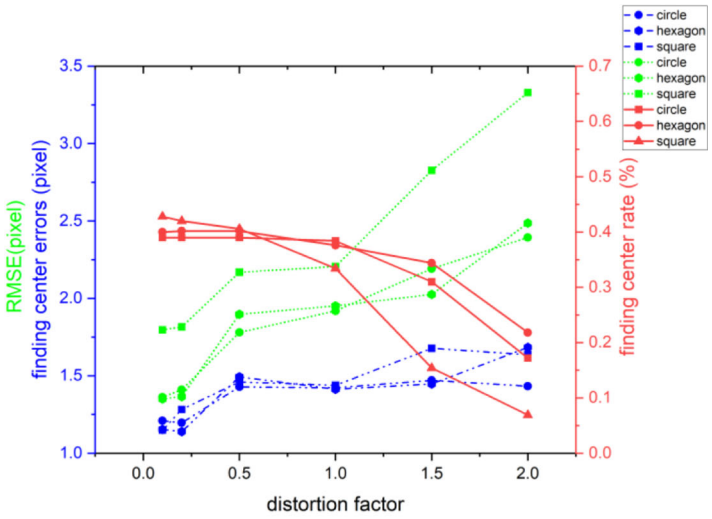


(a) Ground-truth correspondences

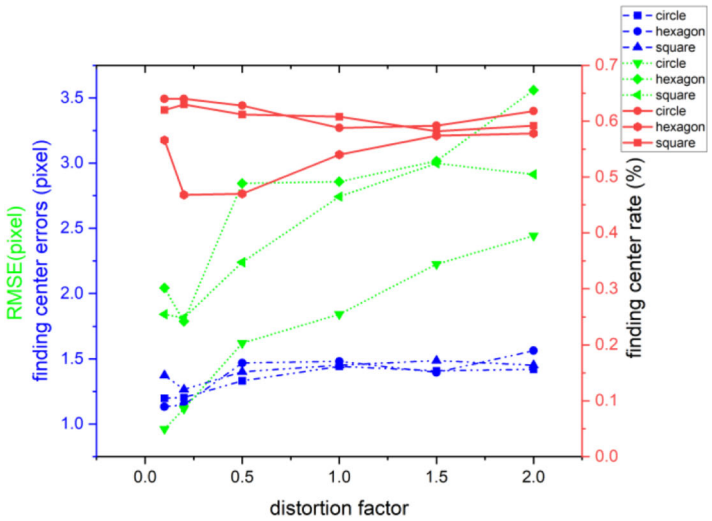


(b) correspondences calculated by SURF and our matching algorithm

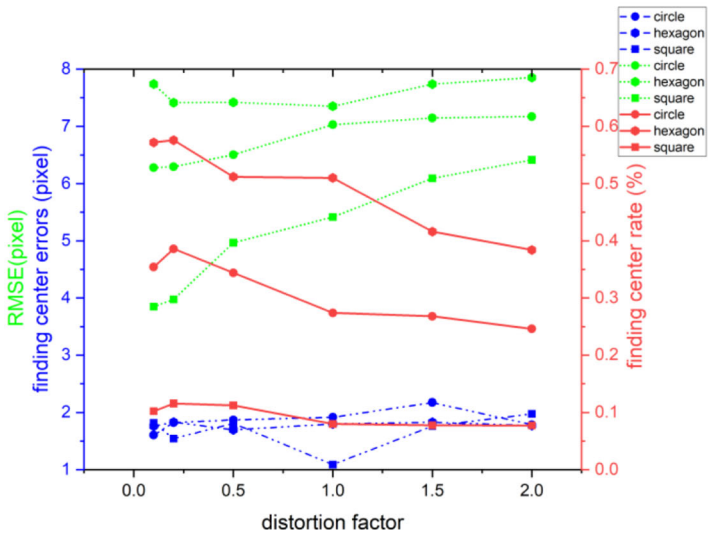
Fig. 14 Matching correspondences comparison, the red arrows represent the moving direction of the simulated production line (**a**) Ground-truth correspondences calculated by the moving speed and the center positions of the patterns (**b**) Matching results calculated by SURF detector and descriptor and our closed-loop feedback matching algorithm



(a) Errors from SIFT

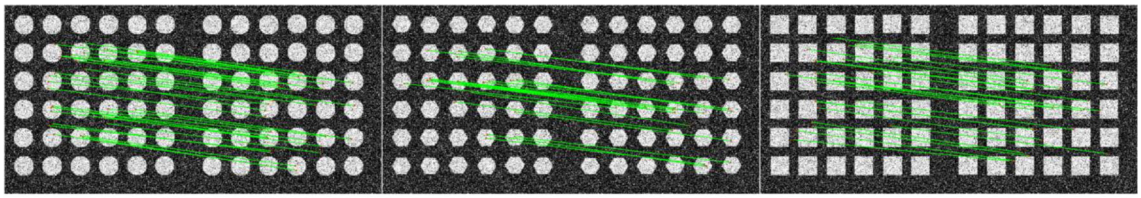


(b) Errors from SURF

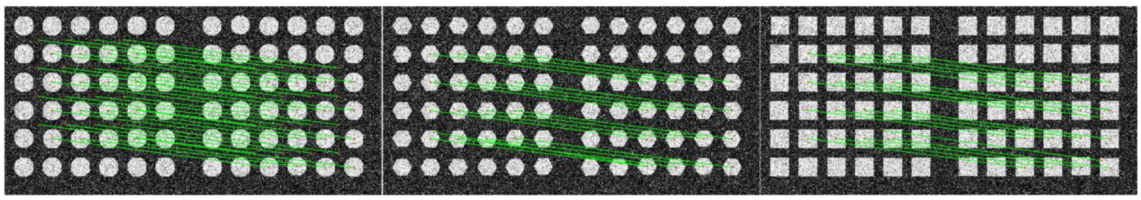


(c) Errors from ORB

Fig. 15 Quantitative errors of our proposed algorithm on deformed grid-based images

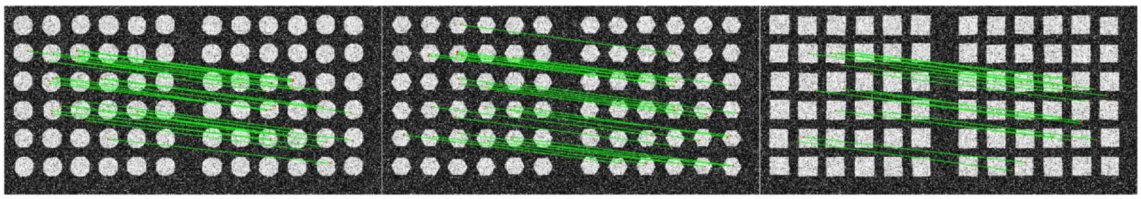


(a) Matching results of feature vectors acquired by ORB detector and descriptor (distortion factor = 0.1)

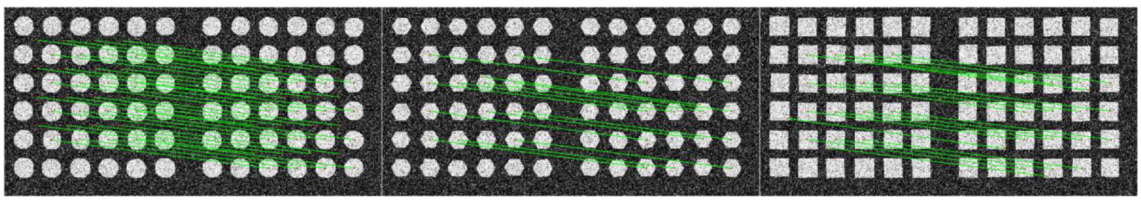


(b) Matching results of feature vectors acquired by SURF detector and descriptor (distortion factor = 0.1)

Fig. 16 The matching results of our algorithm using ORB and SURF detectors and descriptors (distortion factor = 0.1)

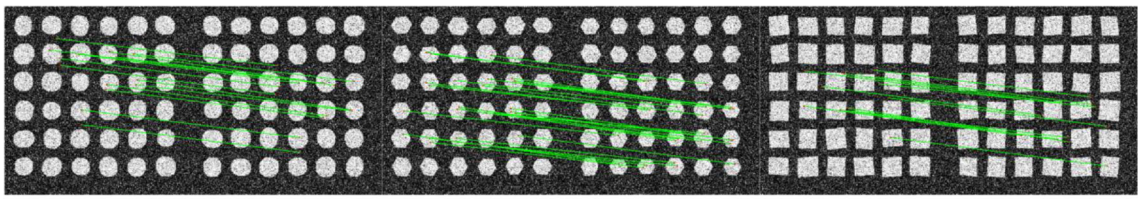


(a) Matching results of feature vectors acquired by ORB detector and descriptor (distortion factor = 0.5)

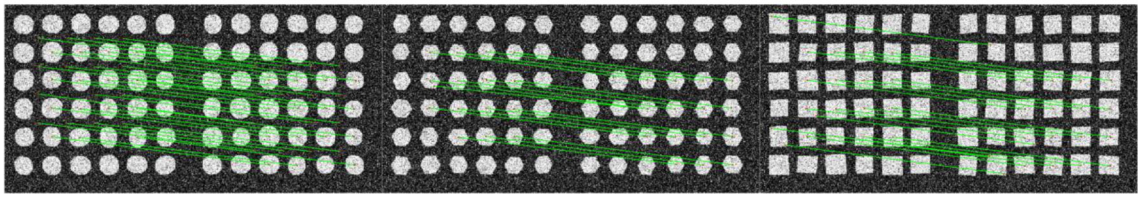


(b) Matching results of feature vectors acquired by SURF detector and descriptor (distortion factor = 0.5)

Fig. 17 The matching results of our algorithm using ORB and SURF detectors and descriptors (distortion factor = 0.5)

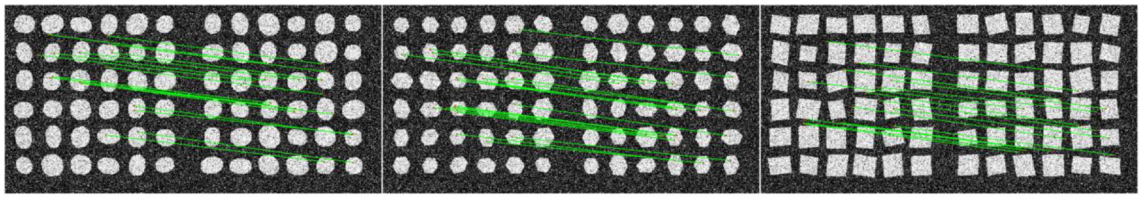


(a) Matching results of feature vectors acquired by ORB detector and descriptor (distortion factor = 1.0)

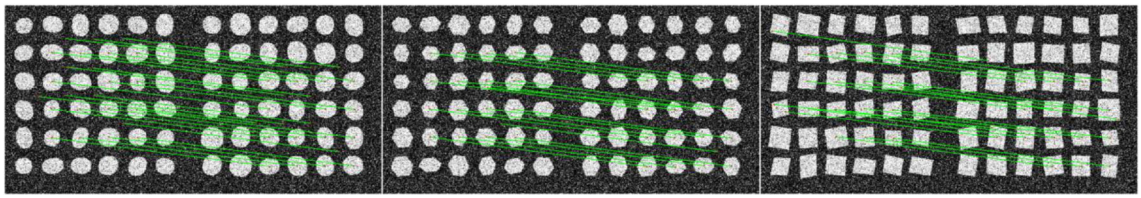


(b) Matching results of feature vectors acquired by SURF detector and descriptor (distortion factor = 1.0)

Fig. 18 The matching results of our algorithm using ORB and SURF detectors and descriptors (distortion factor = 1.0)



(a) Matching results of feature vectors acquired by ORB detector and descriptor (distortion factor = 2.0)



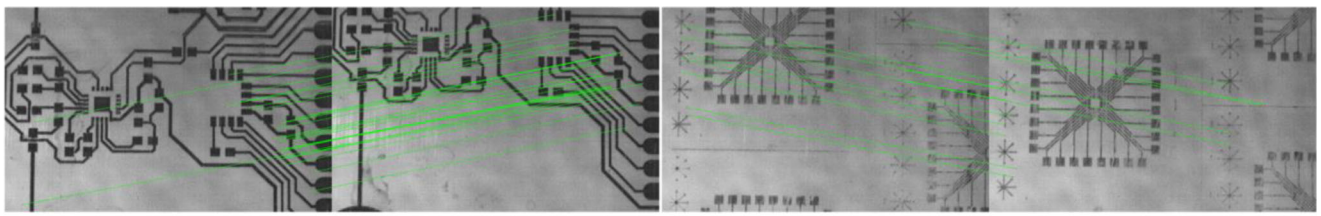
(b) Matching results of feature vectors acquired by SURF detector and descriptor (distortion factor = 2.0)

Fig. 19 The matching results of our algorithm using ORB and SURF detectors and descriptors (distortion factor = 2.0)

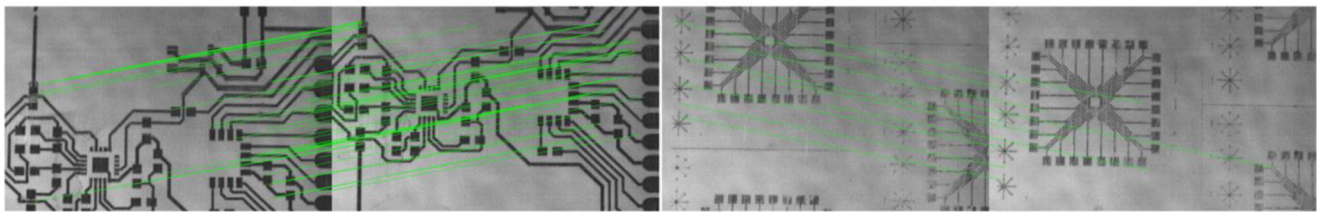
errors of all three detectors and descriptors are less than 2 pixels. That proves our matching algorithm can locate the distorted pattern centers accurately. Meanwhile, the SURF detector finds the most centers and keeps the detected center ratio stable when the distortion factor even increases up to 2.0. The SIFT detector finds fewer centers than SURF and the detected center ratio decreases when the distortion factor increases. The pattern centers on the edges are partially missed which can be seen from Fig. 14b. It is because that the DoG detector in the SURF and SIFT algorithms is not stable on the borders of the images due to the zero-padding for the Gaussian blurring process. The ORB detector finds more centers in the circle pattern dataset, fewer in the hexagon pattern dataset, and the least in the square pattern dataset. It is because that the ORB detector is more sensitive to corners than the centers of the pattern in the hexagon and square pattern datasets. Furthermore, compared to the other methods mentioned before, our closed-loop feedback matching algorithm successfully finds correct correspondences in the highly distorted (distortion factor up to 2.0) image sequences. Using the SIFT and SURF detectors and descriptors, the RMSEs are less than 3.5 pixels even though the distortion factor is up to 2.0. Using the ORB detector and descriptor, the RMSE is a little higher than SIFT and SURF, which is less than 8 pixels. It is because that the SIFT and SURF descriptors are more robust to noise due to the longer feature vectors. However, the local distortions cannot be simply represented by a projective transformation matrix ($T_{proj}(k)$, and calculated by the ground-truth centers using Eq. (14)). Therefore, the RMSEs of the closed-loop feedback matching algorithm are acceptable.

More experiment results

Figure 16, 17, 18 and 19 show the matching results of our algorithm using ORB and SURF detectors and descriptors. They are mostly consistent as SIFT (Fig. 13d).

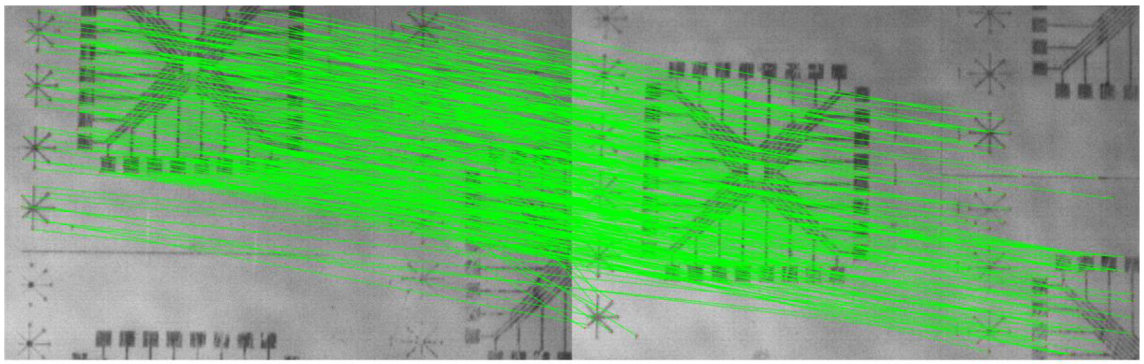


(a) Matching example of our proposed algorithm using the SURF detector and descriptor

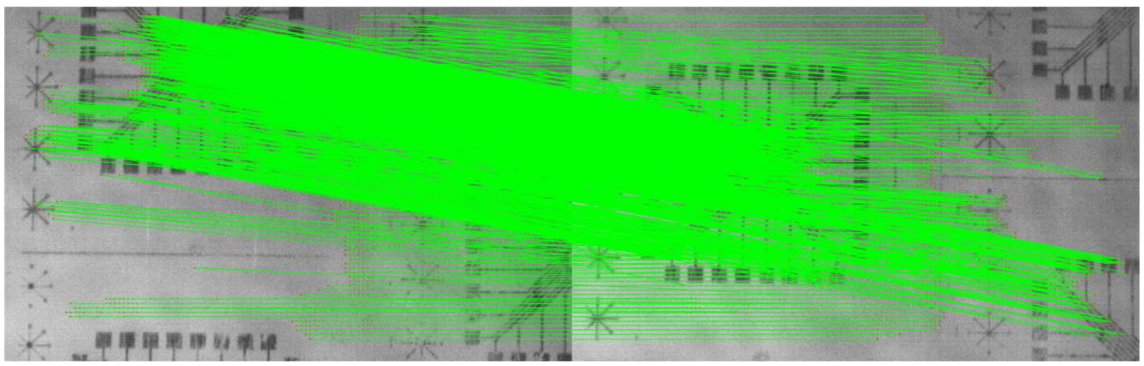


(b) Matching example of our proposed algorithm using the ORB detector and descriptor

Fig. 20 The matching results of our algorithm using ORB and SURF detectors and descriptors



(a) Matching example of the end-to-end registration algorithm SuperGlue



(b) Matching example of the end-to-end registration algorithm LoFTR

Fig. 21 The matching results of end-to-end registration algorithms SuperGlue and LoFTR on the R2R_Autofocus2 dataset

Appendix 4: More experiment results on real-world moving flexible targets

Implementing the proposed algorithm, Fig. 20 shows the matching examples with ORB and SURF detectors and descriptors in R2R_Autofocus1 and R2R_Autofocus2 datasets. Figure 21 shows the matching results of the two deep-learning-based end-to-end registration algorithms SuperGlue and LoFTR on R2R_Autofocus2 dataset.

Acknowledgements This work is supported in part by the National Science Foundation (CMMI #1916866, CMMI #1942185, and CMMI #1907250). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Ma Y, Niu D, Zhang J, Zhao X, Yang B, Zhang C (2022) Unsupervised deformable image registration network for 3D medical images. Appl Intell 52:766–779. https://doi.org/10.1007/s10489-021-02196-7
2. Kawulok M, Benecki P, Piechaczek S, Hrynczenko K, Kostrzewa D, Nalepa J (2019) Deep learning for multiple-image super-resolution. IEEE Geosci Remote Sens Lett 17:1062–1066
3. Devi PRS, Baskaran R (2021) SL2E-AFRE : personalized 3D face reconstruction using autoencoder with simultaneous subspace learning and landmark estimation. Appl Intell 51:2253–2268. https://doi.org/10.1007/s10489-020-02000-y
4. Hosseini MS, Moradi MH (2022) Adaptive fuzzy-SIFT rule-based registration for 3D cardiac motion estimation. Appl Intell 52:1615–1629. https://doi.org/10.1007/s10489-021-02430-2
5. Mehmood Z, Mahmood T, Javid MA (2018) Content-based image retrieval and semantic automatic image annotation based on the weighted average of triangular histograms using support vector machine. Appl Intell 48:166–181. https://doi.org/10.1007/s10489-017-0957-5
6. Chen J, Xu Y, Zhang C, Xu Z, Meng X, Wang J (2019) An improved two-stream 3D convolutional neural network for human action recognition. In: 2019 25th International Conference on Automation and Computing (ICAC), pp 1–6. https://doi.org/10.23919/IConAC.2019.8894962
7. Du X, Anthony BW, Kojimoto NC (2015) Grid-based matching for full-field large-area deformation measurement. Opt Lasers Eng 66:307–319
8. Wang X, Liu X, Zhu H, Ma S (2017) Spatial-temporal subset based digital image correlation considering the temporal continuity of deformation. Opt Lasers Eng 90:247–253
9. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Image Vis Comput 60:91–110
10. Bay H, Tuytelaars T, Van Gool L (2006) Surf: speeded up robust features. In: Leonardis A, Bischof H, Pinz A (eds.) European conference on computer vision. Springer, Berlin, Heidelberg, pp. 404–417. https://doi.org/10.1007/11744023_32
11. Calonder M, Lepetit V, Strecha C, Fua P (2010) Brief: binary robust independent elementary features. In: Daniilidis K, Maragos P, Paragios,N (eds.) European conference on computer vision. Springer, Berlin, Heidelberg, pp.778–792. https://doi.org/10.1007/978-3-642-15561-1_56

12. Aldana-Iuit J, Mishkin D, Chum O, Matas J (2020) Saddle: fast and repeatable features with good coverage. Image Vis Comput 97: 3807
13. Harris CG, Stephens M (1988) A combined corner and edge detector. In: Alvey vision conference. Citeseer, vol 15, pp 10–5244
14. Zaragoza J, Chin T-J, Tran Q-H et al (2014) As-projective-as-possible image stitching with moving DLT. IEEE Trans Pattern Anal Mach Intell 36:1285–1298
15. Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: an efficient alternative to SIFT or SURF. In: 2011 international conference on computer vision. IEEE, pp 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544
16. DeTone D, Malisiewicz T, Rabinovich A (2018) SuperPoint: self-supervised interest point detection and description. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp 224–236
17. Aryal S, Ting KM, Washio T, Haffari G (2017) Data-dependent dissimilarity measure: an effective alternative to geometric distance measures. Knowl Inf Syst 53:479–506
18. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Association for Computing Machinery, New York, pp 381–395. https://doi.org/10.1145/358669.358692
19. Barath D, Noskova J, Ivashechkin M, Matas J (2020) MAGSAC++, a fast, reliable and accurate robust estimator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 1304–1312
20. Chen S, Zhong S, Xue B, Li X, Zhao L, Chang CI (2020) Iterative scale-invariant feature transform for remote sensing image registration. IEEE Trans Geosci Remote Sens 59:3244–3265
21. Sarlin P-E, DeTone D, Malisiewicz T, Rabinovich A (2020) SuperGlue: learning feature matching with graph neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 4938–4947
22. Sun J, Shen Z, Wang Y, Bao H, Zhou X (2021) LoFTR: detector-free local feature matching with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 8922–8931
23. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser (2017) Attention is all you need. In: Guyon I, Von Luxburg U, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds.) Advances in Neural Information Processing Systems. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
24. Szeliski R (2006) Image alignment and stitching. In: Handbook of mathematical models in computer vision. Springer, Boston, pp 273–292. https://doi.org/10.1007/0-387-28831-7_17
25. Yan J, Du X (2020) Real-time web tension prediction using web moving speed and natural vibration frequency. Meas Sci Technol 31:115205
26. Burkardt J (2014) The truncated normal distribution. Department of Scientific Computing Website, Florida State University 1–35
27. Vedaldi A, Fulkerson B (2010) VLFeat: an open and portable library of computer vision algorithms. In: proceedings of the 18th ACM international conference on multimedia. Association for Computing Machinery, New York, pp 1469–1472. https://doi.org/10.1145/1873951.1874249
28. Pixelink Capture Software (n.d.) https://pixelink.com/products/software/pixelink-capture-software/. Accessed 25 Jul 2020
29. Triggs B, McLauchlan PF, Hartley RI, Fitzgibbon AW (1999) Bundle adjustment—a modern synthesis. In: Triggs B, Zisserman A, Szeliski R (eds.) Vision Algorithms: Theory and Practice. Springer, Berlin, Heidelberg, pp 298–372. https://doi.org/10.1007/3-540-44480-7_21
30. DiMeo P, Sun L, Du X (2021) Fast and accurate autofocus control using Gaussian standard deviation and gradient-based binning. Opt Express 29:19862–19878. https://doi.org/10.1364/OE.425118
31. Ma R (2021) Grid-based-patterns creation. https://github.com/cucum13er/Grid-based-patterns-creation. Accessed 3 May 2021