The Journal of Engineering

The 3rd Asian Conference on Artificial Intelligence Technology (ACAIT 2019)

Recurrent spatial transformer network for high-accuracy image registration in moving PCB defect detection

eISSN 2051-3305
Received on 14th October 2019
Accepted on 19th November 2019
doi: 10.1049/joe.2019.1182
www.ietdl.org

Weibo Huang[1], Guoliang Hua[1] ✉, Zhaofu Yu[2], Hong Liu[1]

[1]*Key Laboratory of Machine Perception, Shenzhen Graduate School, Peking University, Shenzhen City, Guangdong Province, People's Republic of China*

[2]*Shenzhen Skyworth-RGB Electronic Co., Ltd., Shenzhen City, Guangdong Province, People's Republic of China*

✉ *E-mail: glhua@pku.edu.cn*

Abstract: Defects detection is an extremely important part in the production of printed circuit board (PCB) to guarantee the quality and reliability. A widely used and researched method for this task is referential comparison method. However, this method gets poor performance on the situation that PCB moves on the conveyor, due to the lack of ability to register test images with referential images. Therefore, accurate image registration which estimates the affine transformation between the captured test images and the referential images becomes an urgent problem to be solved. In this study, a spatial transformer network is proposed. In particular, a recurrent image registration strategy is introduced to step-by-step register the images in a recurrent progress. Furthermore, a referential part is designed to help training the network. Besides, in order to simulate the real moving PCB images and train the proposed model, a factitious dataset is generated by applying random affine transformations to real PCB images. Experiments show that recurrent spatial transformer network can achieve pixel-level accurate image registration. The defects detection precision of referential comparison method has a great improvement by using the registration algorithm.

1 Introduction

Nowadays, electronic products have been widely used in our daily life, such as, mobile phone, computer, television and so on. As the core part of electronic products, high-quality printed circuit board (PCB) is required to satisfy the frequent use. Therefore, it is of great significance to detect the defects of PCB during the production progress. Up to now, the existing literature on vision-based PCB defects detection can be divided into three types, reference comparison method, non-reference method and hybrid method. Reference comparison method needs to store a template image in advance, and compare test images with the template image in pixel-to-pixel or feature-to-feature level. If any pixels or features do not match, then defects may exist [1–3]. Therefore, the idea of reference comparison method is rather straightforward thus easy to be implemented. However, due to the pixel-to-pixel or feature-to-feature comparison strategy, the accuracy of image registration has a significant effect on the precision of defect detection. Non-reference methods perform the inspection according to the design specification standards. If any region does not satisfy with the standards, then this region is regarded as a defect [4]. This method works well and fast with less computer storage when detecting some kinds of defects, such as, the widths and spacing violations. However, it may lose some large defects and distortion characteristics. What is more, this method heavily relies on the standardisation of the conductor trace types. Hybrid inspection method combines the reference method and the non-reference method to exploit the strength and overcome the weakness of each method [5–7]. However, it takes a large amount of computational complexity and is hard to implement in the realistic scene.

With the fast development of computer science technology, in recent years, some researchers attempt to improve the robustness, accuracy and effectiveness of the methods mentioned above. One alternative method is regarding the defect detection problem as object detection task which can be solved by deep learning methods. Zhisheng Lu *et al.* [8] proposed a method by extracting the histogram of oriented gradients and the local binary pattern features to train a PCB defects detector. Can Zhang *et al.* [9] used a sliding window to detect the flaw of PCB based on the convolutional neural networks (CNNs). Another alternative method tries to enhance the precision of image registration to further improve the accuracy of reference comparison method. Linhui Dai *et al.* [10] proposed an improved SIFT-PSO algorithm to enhance the image registration precision. In our previous work [11], an image registration method for PCB defects detection was developed by combining the feature-based registration method and the cross-correlation method. In this paper, we still focus on developing the high-precision image registration for PCB defects detection.

An illustration of the production line in the factory is shown in Fig. 1. A camera is fixed in one place and PCBs are continually delivered on the production line. Therefore, there exist transformations between the images of one PCB captured by the camera at different times. In order to deal with this problem, it is common to make the PCB stop at a fixed place directly under the camera, and wait to capture an image. Although this idea is

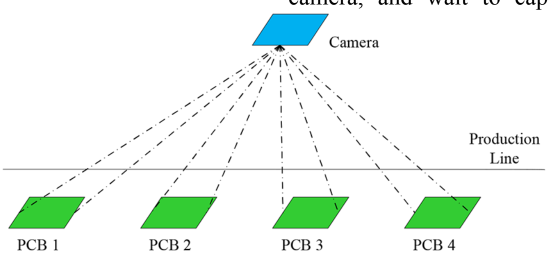


Fig. 1 *Illustration of the production line. PCBs moves on the production line and camera is fixed at a place*

straightforward and easy to be implemented, it will significantly reduce the production efficiency. In contrast, one ideal choice is to capture image and detect defects during the transfer of PCB. Although there are some image registration methods, it is hard to register the moving PCB image accurately, due to the large transformation. Thus in this paper, a high-precision and robust algorithm based on CNNs is proposed to achieve pixel-level accurate image registration and detect defects of moving PCBs by using the referential comparison method. Our main contributions are as follows:

- A new image registration neural network which is based on the spatial transformer network (STN) is proposed to achieve pixel-level accurate image registration. This network is trained without the labelled transform matrix.
- An end-to-end network is established which can not only predict the transform matrix but also output registered images. Thus, there is no need to do the transformation according to the predicted affine transform matrix in another step. All these works are achieved in one stage.
- The RSTN is further combined with the referential comparison method to detect the defects of moving PCB and validate its performance of pixel-level registration.

2 Related work

In this section, some related works are discussed. According to different kinds of information used in the image registration process, image registration methods can be divided into three types: methods based on greyscale information, methods based on transform-domain information and methods based on feature information. Using similarity measurement function, the greyscale-information-based method gets the registration parameters when the max similarity value is found in the searching process [12, 13]. This method can achieve high-accurate registration but greatly influenced by the similarity measure function with a lot of computation. The method based on transform-domain information transforms the image into corresponding transform domain through Fourier transform or other transform methods [14]. According to the specific performance in the transformation domain of rotation, shift and other types, the transformation parameters can be gained to register images. This method can achieve image registration in some cases, but has great limitations, such as high computation cost and low registration accuracy. The feature-based method extracts and matches the features between reference image and transformed image firstly. Then, transformation parameters are estimated from these matched feature points' coordinates. This method has outstanding performance on image registration in an robust and fast way. Some commonly used features include Harris corners [15], SIFT [16], SURF [17] and so on.

With the deep learning algorithm developing so fast, there are several works searching for modelling transformations with neural networks. Hinton *et al.* [18] proposed transforming auto-encoders to use the neural network to learn the brightness, orientation and scale invariance features. For the complicated full 2D affine transformation, the capsule is set to output nine values as the affine transformation matrix, learning to generate the transformed images of objects. Tieleman [19] further designed a network to predict the transformation where the objects were explicitly affine-transformed. In addition, there are some works that focus on studying transformation-invariance representations and attention mechanism for feature selecting. Lenc and Vedaldi [20] studied the representations of the original image and transformed image by measuring their equivalence and invariance. A feedback connections method which can actively select the important features to do the classification task is proposed in [21]. Google DeepMind developed an extremely outstanding module, named spatial transformer networks, to explicitly do the spatial transformation of data within the network [22]. The result shows that STN can learn invariance to translation, rotation, scale and more generic warping and has amazing capacity to actively spatially transform feature maps when inserting to existing CNNs.

3 Method for defects detection of moving PCBs

In this section, image registration method and defects detection method are introduced. In order to achieve accurate image registration, a recurrent spatial transformer network (RSTN) is introduced, the architecture of which is shown in Fig. 2. RSTN contains two parts: feature extractor and STN cell, which is the same as concatenated spatial transformer network (STN_c) of [22]. RSTN is different with STN_c in the architecture, as is shown in Fig. 3. It shows that STN_c concatenates several STNs in Fig. 3*b*. Images go through these STN cells to achieve registration step by step. However, our proposed RSTN makes STN as a cell in a recurrent architecture, as is shown in Fig. 3*a*. The registered image will be the input of next step and the same STN cell will generate a deeper registered image for another followed step. The details of STN and RSTN are given in Sections 3.1 and 3.2. For defects detection, referential comparison method is used and the implementation details are introduced in Section 3.3.

3.1 Spatial transformer network

Since the STN was proposed in 2016, it gets a lot of attention and is used in many aspects. It can explicitly do the spatial transformation on the input within the network as well as learn invariance to translation, rotation, scale and more generic warping. Meanwhile, STN has amazing capacity to transform feature maps actively when inserted to existing CNNs. In this paper, we focus on the image registration. Therefore, images are taken into consideration specifically instead of feature map. As shown in Fig. 2, STN contains three parts: localisation network, grid generator and sampler. In localisation network part, the input is the feature vector and the output is the transformation matrix $\boldsymbol{\theta}$. Thus, the target of this part is to predict the transformation matrix for the grid generator to generate the sampling grid. The output's size can vary from the type of the transformation, e.g. the output's size is 2×3 for affine transformation.

In order to do the registration of the input image, each output pixel should find the corresponding pixel in the input image. The relationship between the pixels' coordinate of input and output image should satisfy the transformation formula. Affine transformation model T can contain the transformation types existed in the real PCB production line, such as: shift, rotation, scale. Therefore, for the coordinate of the regular grid in the output image (x^t, y^t) and the coordinate of the input image (x^s, y^s), the pointwise transformation can be written as

$$\begin{bmatrix} x_i^s \\ y_i^s \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} x_i^t \\ y_i^t \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & t_1 \\ a_3 & a_4 & t_2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i^t \\ y_i^t \\ 1 \end{bmatrix}. \tag{1}$$

For an output image $V = R^{H' \times W' \times C'}$ with height H', width W' and channel C', the grid generator will generate $H' \times W'$ sampling points $\mathcal{T}_\theta(G)$. After getting the sampling grid, a sampler will do the sampling on the input image according to the sampling points $\mathcal{T}_\theta(G)$. The value of each pixel in the output image can be given as

$$\begin{gathered} V_i^c = \sum_n^H \sum_m^W U_{nm}^c \cdot k(x_i^s - m; \Phi_x) k\big(y_i^s - n; \Phi_y\big), \\ \forall i \in [1 \dots H'W'] \; \forall i \in [1 \dots C], \end{gathered} \tag{2}$$

where V_i^c means the intensity value of pixel (x_i, y_i) in the output image's channel C, U_{nm}^c means the intensity value of pixel (x_n, y_m) in the input images channel C, Φ_x, Φ_y are the parameters of a generic sampling kernel $k(\cdot)$, x_i^s, y_i^s are the ith coordinates in $\mathcal{T}_\theta(G)$.

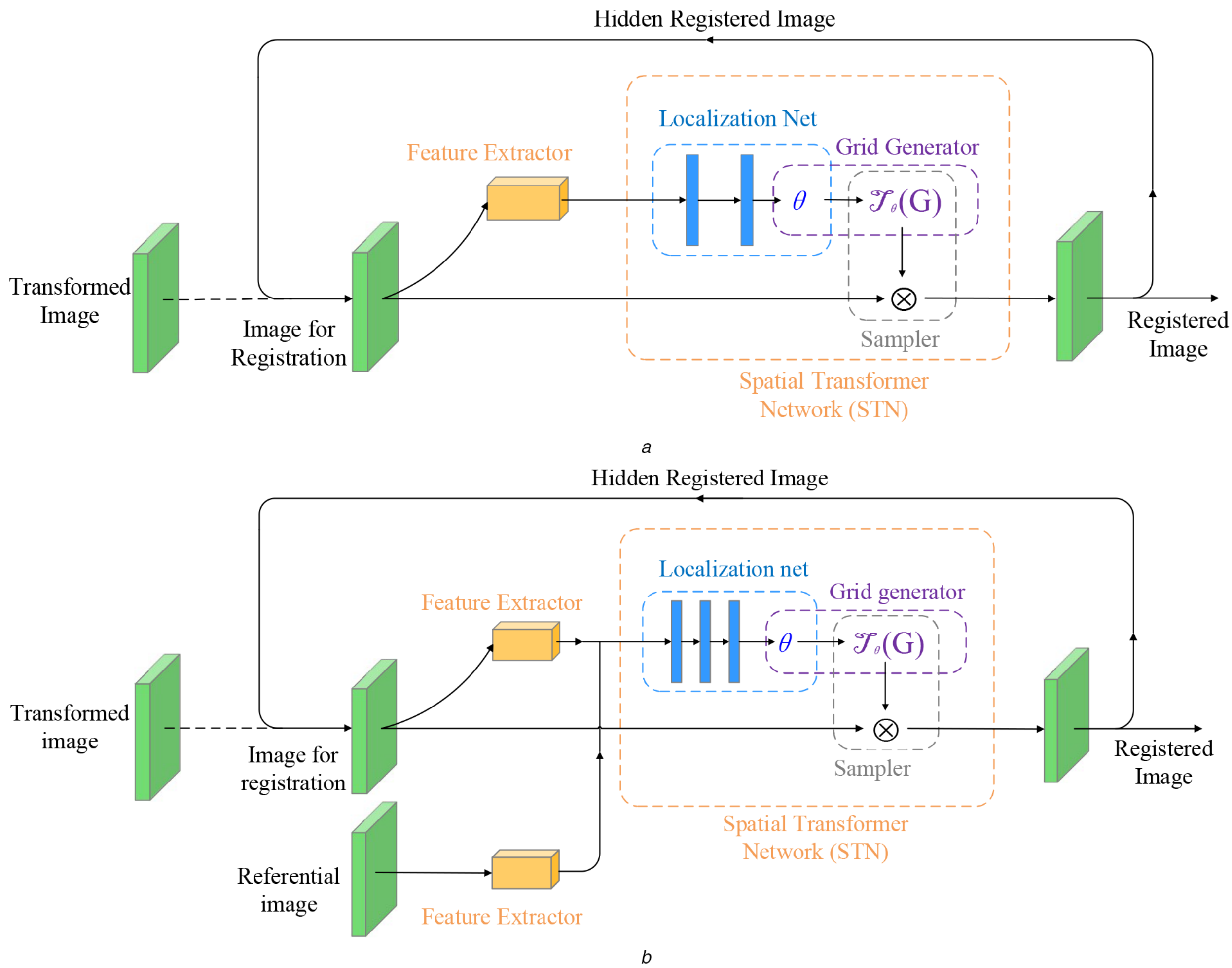


Fig. 2 *Proposed network architecture*
(a) Recurrent spatial transformer network, ***(b)*** Recurrent spatial transformer network with the referential part

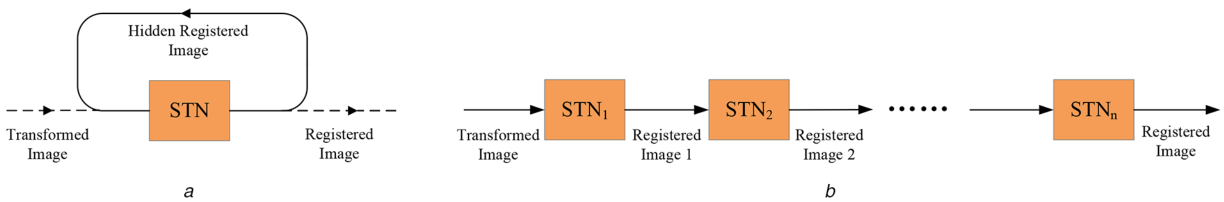


Fig. 3 *Comparison between a standard STN and RSTN*
(a) Recurrent STN structure where the STN cell is put in a recurrent process, ***(b)*** Concatenated STN structure where the STN cell concatenates one by one

3.2 Recurrent spatial transformer network

In this section, the proposed RSTN will be introduced in detail. When the input image has a big transformation compared with the reference image, a single STN may be unable to do the accurate image registration. A solution is to use a sequence of STNs to transform the image step by step [22]. As is shown in Fig. 3*b*, the first STN cell will do the transformation on the input image and the followed STNs will do the further registration on the registered image from the former one. This is a straightforward method to achieve accurate image registration. However, experiments show that this architecture results in early converging. In order to deal with this problem, we proposed a RSTN. In the proposed RSTN, STN is regarded as the transformer cell as is shown in Fig. 3*a*. At the very beginning, the input of the STN cell is the transformed image, then, the first registered image is obtained. Later, the STN cell takes the registered image from former stage as input and output deeper registered image. Therefore, the dotted line in Fig. 3*a* means that the input or output only operate in the first or last stage. The number *r* of recurrent time is defined in advance which means the recurrent registration process will perform *r* rounds.

The weights of STN cell are shared in the recurrent stages, which reduces the amount of parameters greatly compared to the concatenated STN framework. At each stage, the STN cell further registers the input image so that images with great transformation can be registered step by step. This thought is exactly the same as the concatenated architecture's, while the RSTN achieves with less parameters. What is more, because intermediate registered images with different level transformations are put into the STN cell to do further image registration, the STN cell is trained to be sensitive to different level transformations. As for the STN_c, the deeper STN cell is sensitive to the lighter transformed image. At the same time, it is easier to train the RSTN than the STN_c because the amount of the parameters is reduced a lot in RSTN.

In order to take the advantage of having referential image in advance, a RSTN with the referential part (r-RSTN) is proposed. As is shown in Fig. 2*b*, the referential image is put into the feature extractor to get its feature vector. The weights of the feature extractor in the referential part will share with the one in the RSTN part. For one thing, it will not add parameters to the network, for another, it makes that reasonable to compare the feature map of referential image and transformed image for further learning. Then, this feature vector is concatenated with the feature vector of

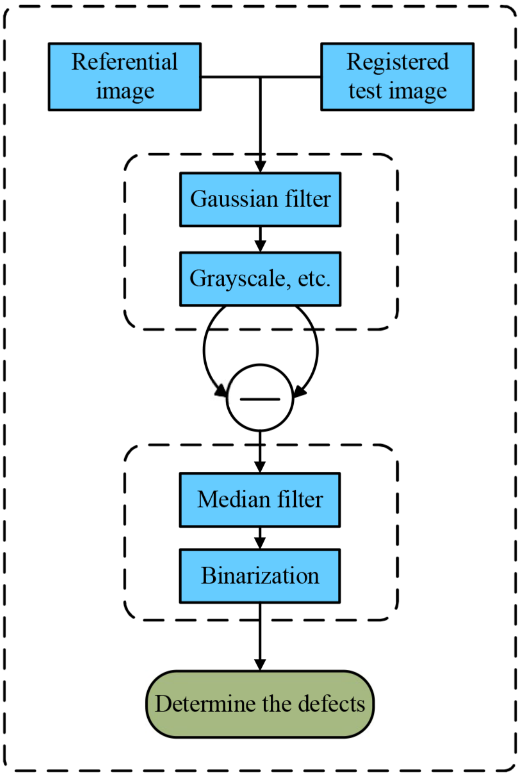


Fig. 4 *Illustration of the process of referential comparison method*

registered image in each stage to predict the transformation matrix. By doing this, the network can learn from the relationship between the transformed image and referential image, which helps to predict the transformation matrix.

3.3 Referential comparison method

As illustrated in Section 1, it is efficient to use the referential comparison method to detect the defects after registering PCB images. In our method, we first use RSTN to pixel-level image registration, then the referential comparison method is applied to detect PCB defects. The process of referential comparison method is shown in Fig. 4. The input is a pair of registered image and its referential image. Since there may exist serration on the edges of the registered image, a Gaussian filter is firstly applied to smooth the image. Then, both referential image and registered test image are transferred into greyscale style. To do the comparison in pixel level, we directly do subtraction between the greyscale referential image and registered test image and get the subtraction image. Ideally, pixels in subtraction image which do not equal to zero are supposed to be the defects. However, because of a lot of factors, such as imbalance illumination, salt and pepper noise, camera shaking and so on, it is also essential to do image processing on the subtraction image. Firstly, a threshold is set to select the parts where big difference exists. Pixels of the subtraction image that are less than threshold will be set to 0 and others will be set to 255. Furthermore, a median filter is used to filter some salt and pepper noise which may be greater than the threshold. Finally, existed parts are likely to be the position of defects and the corresponding parts in the test image will be regarded as defects.

4 Experiments and discussions

In this section, experiments will be introduced to show that the proposed algorithm has outstanding performance on image registration and defects detection comparing with the traditional referential comparison method. The moving PCB dataset used in our experiments is introduced in Section 4.1. In Section 4.2, image registration experiment is performed to show the advantage of the proposed method against STN_c. In Section 4.3, the process of defects detection is shown and experiment shows that the proposed method achieves pixel-level accurate image registration so that traditional referential comparison method can be successfully used to detect defects for moving PCB. The defects detection result is compared with the traditional method using SIFT or SURF to do the image registration.

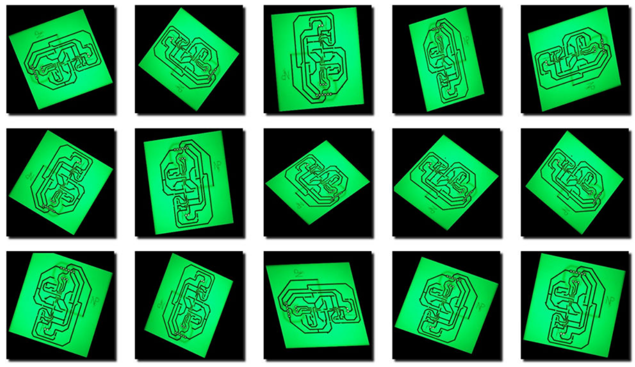


Fig. 5 *Some examples of the PCB dataset. These images are generated from real PCB images by performing different affine transformation*

4.1 Moving PCB dataset

As illustrated in Section 1, PCB images captured on the moving production line may exhibit transformation relationship with the referential image. Commonly, these transformations include scale, rotation and shear, belonging to affine transformation, which can be described by formula (1). However, to the best of our knowledge, there is no dataset of moving PCB images. Therefore, we generate a factitious dataset to simulate the real moving PCB images, in which images are transformed by random affine transformation matrix. Here, method to generate the dataset is illustrated in detail. First of all, we captured 11 real PCB images with no transformation, among of which, one is the referential image and others are test images with different defects. Then, we set the rotation angle a random value between 0° and 360°, scale varies from 0.6 to 1.1 and the extent of shear is −0.2 to 0.2. All the transformation parameters are set randomly in order to make sure that the algorithm is robust enough. For each test image, we generate 300 images with random affine transformation matrix to build the PCB dataset. Finally, generated PCB dataset is split into two parts: training dataset and test dataset. The training dataset contains 2100 images generated from 7 test images and the test dataset contains 900 images from other 3 test images. Some examples are shown in Fig. 5.

4.2 Image registration

Here are the details of training strategies. Resnet34 is used to be the feature extractor. In the referential RSTN, the referential image and the transformed image share the same feature extractor. We train our model 300 epochs with the batch size of 32. The learning rate is 0.001 at first and decrease at the rate of 0.8 per 20 epochs. For the optimiser, we choose the Adam algorithm which is proposed by Diederik Kingma and Jimmy Ba in 2015 [23]. The hyper-parameters are set as follows: $\alpha_1 = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$. Instead of using the groundtruth of transformation matrix as supervision, the consistency between registered image and referential image is used to train the network. If images are registered in high accuracy, they should be similar to referential image in pixel level. Therefore, image similarity loss function is built to train the network. The loss function combines the structure similarity index (SSIM) and L1 loss together for better performance. The SSIM loss can speed up the network converging and the L1 loss supervises the model to achieve the pixel-level accurate image registration. Thus, the loss function can be defined as

$$L(X,Y) = \lambda \cdot L_{\mathrm{SSIM}}(X,Y) + (1-\lambda)L_1(X,Y),$$
$$L_{\mathrm{SSIM}}(X,Y) = 1 - \frac{(2u_Xu_Y + C_1)(2\sigma_{XY} + C_2)}{(u_X^2 + u_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)}, \tag{3}$$
$$L_1(X,Y) = \frac{1}{N}\sum_i \sum_j \| X(i,j) - Y(i,j) \|_1,$$

where λ is a hyper-parameter that added to weight the SSIM loss and L1 loss. In our experiment, λ is set to 0.6. X, Y are the registered image and target image, respectively. u_X, u_Y are the average value of X, Y, while σ_X and σ_Y are the standard deviation of X,Y and σ_{XY} is the covariance of X, Y. C_1, C_2 are the constant values to make sure the SSIM loss stable by avoiding the situation that the denominator equals to zero.

The comparison of STN, r-STN, RSTN and r-RSTN in training loss is shown in Fig. 6. It can be seen from Fig. 6*a* that the loss decreases a little when two STNs are connected directly. However, the network is hard to train and converges to be a high-value loss when more STN concatenates. For RSTN, because there is no more parameters added when increase the circulation of the network, it will not create any burden to train it. It shows that RSTN with three times circulation gets the best performance. However, RSTN also converges in a high-value loss, which may be due to the lack of referential information to supervise the network. Figs. 6*b* and*c* show the comparison between r-STN, RSTN and r-RSTN. For r-STN and r-RSTN, the referential part is added to input more information for the localisation net, in order to help it predict the transformation matrix. As is shown in Fig. 6*c*, all the r-RSTN with different rounds can converge in a low level, which indicates that the problem of RSTN that it converges in a high-value loss with the increase of circulation is solved after adding the referential part. What is more, the performance of r-RSTN with double circulation is as good as RSTN with three times circulation. It shows the function of referential part which helps supervise network to converge.

We evaluate these architectures on the test dataset by calculating the mean SSIM value. The SSIM value of different neural network with different times of transformation on the test dataset and their parameter quantity are shown in Table 1. It can be seen that the quantity of the concatenated architecture's parameters are double increased when the circulation increases. But for our recurrent architecture, the parameter quantity does not increase when the circulation increases. At the same time, r-RSTN gets the best performance no matter how many circulations it has. The results show that the proposed architecture outperforms the concatenated architecture with less parameters.

In Fig. 7, we show some examples of the image registration process of the RSTN with three rounds. The first column is the test images, the second column is the output images of the first round, the third column is the output images of the second round and the last column is the images of the third round. From the second column to fourth column, it is obvious that the images are registered little by little, which illustrates the proposed network architecture is meaningful.

4.3 Defects detection

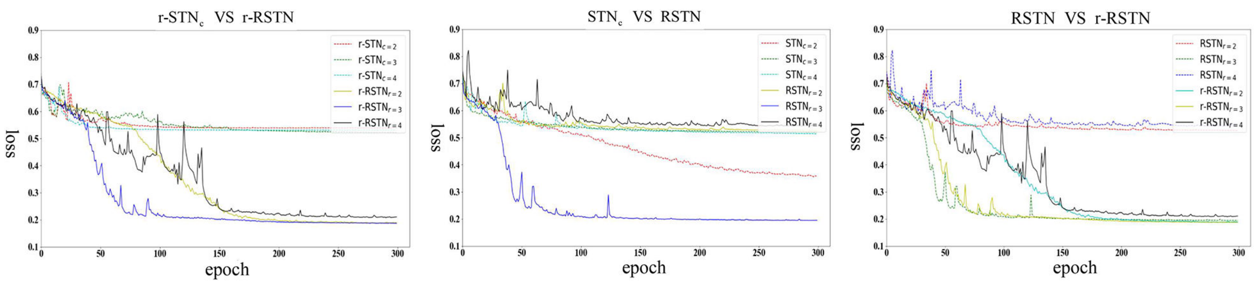
In this section, we show the experiment of defects detection. The registered test images are used to do the defects detection. First of all, we use a Gaussian filter with a window size of 7×7 to smooth the edges of these test images and then convert them into the greyscale images to do the subtraction where the binary threshold is set to 15. Latter, a median filter with a window size of 3 is used to filter some noises. Furthermore, a morphological processing is done to make up the influence of the binary threshold and filters. Finally, the existed parts in the subtraction image are possible to be defects. The performance is compared with the method using SIFT or SURF to do the image registration. Table 2 shows that the proposed method improve the detection precision rate about 26%, and the F1 score about 11%.

Some detection examples are shown in Fig. 8, in which the first column is the referential image, second column is the registered test image, third column is the subtraction image and the last column is the detection results on the registered image.

It can be seen from the subtraction images that the proposed method achieves pixel-level accurate image registration, since only the defect parts exist after image subtraction.

5 Conclusion and future work

In this paper, an improved STN, named RSTN, is proposed, which can be implemented in the defects detection of moving PCB. In addition, according to the specific scene where the referential image can be obtained in advance, an referential attention part is added, r-RSTN, which gets the best performance in image registration experiment. It is shown that RSTN can achieve pixel-level accurate image registration and the defects detection precision of referential comparison method has a great improvement by using our registration algorithm. In the future, we look forward to add the RSTN network into the object detection neural network to achieve end-to-end defects detection.



a *b* *c*

Fig. 6 *Training loss of STN$_C$, RSTN, r-RSTN, respectively*

(a) Comparison between STN$_C$ and RSTN, ***(b)*** Comparison between r-STN$_C$ and r-RSTN, ***(c)*** Comparison between RSTN and r-RSTN. Here, r-* means adding the referential part; STN$_{c=i}$ means i STNs concatenate with each other; RSTN$_{r=i}$ means that RSTN with i rounds and r-RSTN$_{r=i}$ means that r-RSTN with i rounds

Table 1 SSIM value on the test dataset

Method	SSIM				Parameters/*M*			
	1	2	3	4	1	2	3	4
STN	0.3508	0.4602	0.3653	0.3710	22	43	64	85
r-STN	0.3810	0.4492	0.3645	0.3566	22	44	65	87
RSTN	0.3508	0.3572	0.7258	0.3476	22	22	22	22
r-RSTN	0.3810	0.7519	0.7686	0.7304	22	22	22	22

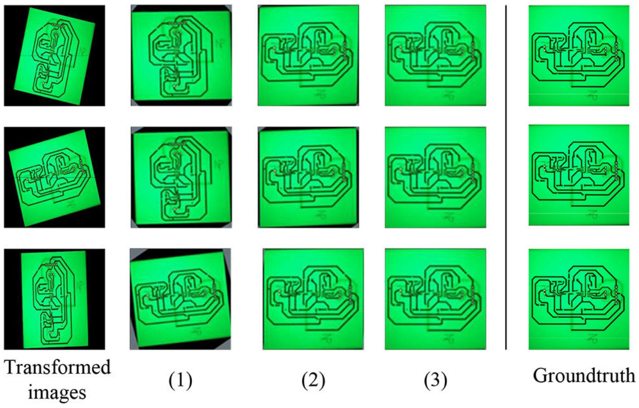


Fig. 7 *Some output examples of RSTN in different stages. (1) to (3) means the stage in recurrent progress. The first and last columns are the images to register and groundtruth, respectively*

Table 2 Defects detection experiment results on the test dataset

Method	*F*1	Precision	Recall
SIFT	0.69569	0.59653	0.83439
SURF	0.51936	0.38904	0.78141
$\text{STN}_{c=2}$	0.42752	0.32209	0.63558
$\text{RSTN}_{r=3}$	0.82032	0.85939	0.78464
$\text{r-RSTN}_{r=3}$	0.79606	0.81871	0.77377

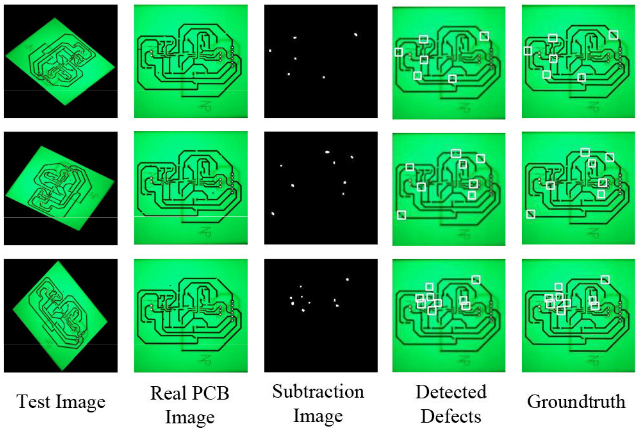


Fig. 8 *Some defects detection examples. Test images are generated from the corresponding real PCB images*

6 Acknowledgments

This work is supported by National key R&D program of China (2018YFB1308600, 2018YFB1308602), Specialized Research Fund for Strategic and Prospective Industrial Development of Shenzhen City (no. ZLZBCXLJZI20160729020003).

7 References

[1] Lee, D.T.: ‘A computerized automatic inspection system for complex printed thick film patterns’. Technical Symp. East, 1978

[2] Nakagawa, Y., Hara, Y., Hashimoto, M.: ‘Automatic visual inspection using digital image processing’, *Hitachi Rev.*, 1985, **34**, (1), pp. 55–60

[3] Lees, D.E., Henshaw, P.D.: ‘Printed circuit board inspection-a novel approach’. Automated Inspection and Measurement. Int. Society for Optics and Photonics, 1987, vol. 730, pp. 164–174

[4] Ejiri, M., Uno, T., Mese, M., *et al.*: ‘A process for detecting defects in complicated patterns’, *Comput. Graph. Image Process.*, 1973, **2**, (3–4), pp. 326–339

[5] Mandeville, J.R.: ‘Novel method for analysis of printed circuit images’, *IBM J. Res. Dev.*, 1985, **29**, (1), pp. 73–86

[6] Benhabib, B., Charette, C.R., Smith, K.C., *et al.*: ‘Automatic visual inspection of printed circuit boards: an experimental system’, *Int. J. Robot. Autom.*, 1990, **5**, (2), pp. 49–58

[7] Yu, S.-S., Cheng, W.-C., Chiang, C.S.: ‘Printed circuit board inspection system pi/1’. Automated Inspection and High-Speed Vision Architectures II. Int. Society for Optics and Photonics, 1989, vol. 1004, pp. 126–135

[8] Lu, Z., He, Q., Xiang, X., *et al.*: ‘Defect detection of pcb based on Bayes feature fusion’, *J. Eng.*, 2018, **2018**, (16), pp. 1741–1745

[9] Zhang, C., Shi, W., Li, X., *et al.*: ‘Improved bare pcb defect detection approach based on deep feature learning’, *J. Eng.*, 2018, **2018**, (16), pp. 1415–1420

[10] Dai, L., Guan, Q., Liu, H.: ‘Robust image registration of printed circuit boards using improved sift-pso algorithm’, *J. Eng.*, 2018, **2018**, (16), pp. 1793–1797

[11] Hua, G., Huang, W., Liu, H.: ‘Accurate image registration method for pcb defects detection’, *J. Eng.*, 2018, **2018**, (16), pp. 1662–1667

[12] Kanneganti, R.K., Pamula, V.K.: ‘Image registration using hampel and modified hampel m-estimators’. 2018 Int. Conf. on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 2018, pp. 1678–1683

[13] Xu, X., Li, X., Liu, X., *et al.*: ‘Multimodal registration of remotely sensed images based on jeffreyâĂŹs divergence’, *ISPRS J. Photogramm. Remote Sens.*, 2016, **122**, pp. 97–115

[14] Hurtós, N., Ribas, D., Cufí, X., *et al.*: ‘Fourier-based registration for robust forward-looking sonar mosaicing in low-visibility underwater environments’, *J. Field Robot.*, 2015, **32**, (1), pp. 123–151

[15] Yan, M., Chen, S., Lan, Y., *et al.*: ‘Image registration and stitching algorithm of rice low-altitude remote sensing based on harris corner self-adaptive detection’, *Trans. Chin. Soc. Agric. Eng.*, 2015, **31**, (14), pp. 186–193

[16] Xiang, Y., Wang, F., You, H.: ‘Os-sift: A robust sift-like algorithm for high-resolution optical-to-sar image registration in suburban areas’, *IEEE Trans. Geosci. Remote Sens.*, 2018, **56**, (6), pp. 3078–3090

[17] Huang, J., Li, J., Liu, L., *et al.*: ‘Pcb image registration based on a priori threshold surf algorithm’. Int. Conf. on Smart Vehicular Technology, Transportation, Communication and Applications, Emeishan City, Sichuan Province, China, 2018, pp. 440–447

[18] Hinton, G.E., Krizhevsky, A., Wang, S.D.: ‘Transforming auto-encoders’. Int. Conf. on Artificial Neural Networks, Espoo, Finland, 2011, pp. 44–51

[19] Tieleman, T.: ‘*Optimizing neural networks that generate images*’ (University of Toronto, Canada, 2014)

[20] Lenc, K., Vedaldi, A.: ‘Understanding image representations by measuring their equivariance and equivalence’. Proc. of the IEEE Conf. on computer vision and pattern recognition, Boston, America, 2015, pp. 991–999

[21] Stollenga, M.F., Masci, J., Gomez, F., *et al.*: ‘Deep networks with internal selective attention through feedback connections’, Adv. Neural. Inf. Process. Syst., Montreal, Canada, 2014, pp. 3545–3553

[22] Jaderberg, M., Simonyan, K., Zisserman, A., *et al.*: ‘Spatial transformer networks’, Adv. Neural. Inf. Process. Syst., Montreal, Canada, 2015, pp. 2017–2025

[23] Kingma, D.P., Ba, J.L.: ‘Adam: amethod for stochastic optimization’. Proc. 3rd Int. Conf. Learning Representations, Banff, Canada, 2014