IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 October 2023, accepted 3 December 2023, date of publication 7 December 2023,
date of current version 14 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3340310

RESEARCH ARTICLE

Online Assembly Inspection Integrating Lightweight Hybrid Neural Network With Positioning Box Matching

SHIWEN ZHAO[1], JUNFENG WANG[1], (Member, IEEE), WANG LI[2], AND LONGFEI LU[1]

[1]Department of Industrial and Manufacturing System Engineering, School of Mechanical Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

[2]Department of Information Technology Equipment, FiberHome Telecommunication Technologies Company Ltd., Wuhan 430074, China

Corresponding author: Junfeng Wang (wangjf@mail.hust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 72271100, and in part by the Industrial Technology Development Program of China under Grant JCKY2021203B072.

ABSTRACT Assembly inspection methods have been widely used in the process of mechanical product assembly for quality issues. However, some challenges remain to be solved, such as low detection efficiency, poor accuracy and sensitive to camera view. This paper proposes an online assembly inspection scheme based on hybrid neural network and positioning box matching. A hybrid multi-task learning neural network with transformer attention mechanism is designed to simultaneously detect key points and assembly parts with high accuracy and strong robustness. Utilizing the key points detection results, the transformation relationships between the in-site assembly images and the standard templates are solved. According to the results of assembly parts detection, the detected 2D positioning bounding boxes are matched with those in the standard assembly templates, so as to evaluate whether the current step has quality problems. In addition, the proposed method is tested on an assembly dataset constructed in this paper. For key points detection, the average error is less than 1 pixel. For parts detection, the mean average precision is 97.66%. The missing and wrong assembly inspection results show that the average F1 score reaches 93.96%. This inspection method can be employed to detect the missing and wrong assembly faults of each assembly step online, improving the assembly quality of products.

INDEX TERMS Assembly inspection, multi-task learning network, positioning box matching, vision transformer.

I. INTRODUCTION

Assembly process is a key stage of mechanical products development. In order to ensure the product quality, the assembly inspection process usually is needed to make sure whether operators or industrial robots complete the assigned task correctly or not. Moreover, in modern industrial production, the demand for small batch customized products is increasing rapidly. Due to economic and technical reasons, these small batches of complex products tend to be assembled in a manual manner. In this mode, the assembly quality and efficiency depend on the skills of operators largely [1]. Diverse and complex assembly processes significantly increase the cognitive load of employees, rising the probability of assembly faults [2].

The commonly method of assembly faults inspection is to manually compare the assembled products with 2D process manual or standard 3D assembly model. The main purpose is to check whether there are missing components and whether the type, position or angle of assembly parts meet the process requirements. However, the manual inspection highly depends on the status and experience level of the operator, and it is usually a laborious and time-consuming work. In addition, the inspection process is usually carried out after all the assembly steps are completed, lacking of monitoring and quality control for each assembly step. When

The associate editor coordinating the review of this manuscript and approving it for publication was Frederico Guimarães.

the missing or wrong assembly errors occur, parts need to be removed and reinstalled again, which seriously reduce assembly efficiency. Therefore, online quality inspection systems have been gradually applied in the assembly field to detect and avoid faults in real time, ensuring consistent product quality and improving assembly efficiency [3].

Currently, vision-based inspection method has been studied by more and more researchers [4]. It is simple and efficient, without additional use of complex sensors, and can adapt to the inspection of most mechanical products [5]. Previous machine vision inspection works are mainly based on the hand-designed feature descriptors and the template matching operation is usually performed to inspect assembly errors. Although it performs well when the features of target region are rich and the image background is relatively fixed, it is not suitable for dynamic assembly scenes with complex background and uncontrollable lighting conditions [6]. And the shortcoming of the traditional template matching-based inspection method is that when the relative position between the physical camera and the target changes, there will be a difference in perspective between the in-site image and the standard template. This difference in perspective will bring errors to the template matching results.

Recently, as deep learning methods have made remarkable achievements in image processing tasks, more and more researchers begin to explore its application in defect inspection [7]. For instance, in structural damage detection, Cha et al [8]. pioneered the deep learning-based cracks inspection. The designed convolutional neural network (CNN) classifier can automatically learn defect features, and effectively locate concrete cracks under various image conditions by a sliding window method. Although many works employ the deep learning for defect inspection, the vast majority of these works focus on surface texture defects, and less attention is paid on functional defects inspection. Surface defects are usually characterized by local texture anomalies, while functional defects, such as assembly position or angle faults of assembled parts usually have no obvious texture abnormalities. To inspect assembly faults, it is not enough just to detect the assembled part, but also to match the detected part category and position to the standard assembly template. Hence, the commonly used defect inspection method using only object detection or segmentation technology cannot effectively detect assembly faults.

This paper proposes an assembly inspection method to further address these issues. The standard assembly templates for each step are pre-constructed in a computer aided design (CAD) software, and then the key points and parts in in-site image are detected respectively using the designed multi-task learning network. The detected key points are used to align the standard template and the in-site image, and the positioning box matching algorithm is employed to judge the missing and wrong assembly faults. The main contributions lie in two aspects:

1) An encoder-decoder multi-task learning neural network integrating the transformer attention mechanism, termed MTL-CenterNet, is designed to integrate key points and parts detection with high accuracy, robustness and timelines.
2) A homography matrix estimation method based on key points detection is proposed to solve the perspective difference between the in-site image and the standard assembly template. Meanwhile, the detected positioning boxes containing rotation information are matched with the template to determine whether the current assembly step has quality problems.

II. LITERATURE REVIEW

A. HAND-DESIGNED FEATURE-BASED ASSEMBLY INSPECTION

In previous works, the hand-designed feature-based methods are commonly used in assembly inspection. Image features, such as contour, edge, color, or corner features [9], [10], [11], are firstly extracted according to the artificially designed descriptors, and are then matched with the standard assembly template. For instance, Liu et al. [12] studied a template based inspection technology in semi-closed narrow space. The Canny contour detection algorithm [13] and subpixel algorithm [14] were used to detect whether there were missing small parts and whether the assembly clearance met the requirements. However, this detection method does not perform well when the target area lacks rich texture. Kim et al. [15] researched the status detection in ship assembly. The GrabCut algorithm [16] was used to segment the ship section objects, and the boundary contour features are extracted and compared with the template database. Due to the large template search space, the detection speed is limited. Hence, Yang et al. [17] proposed an edge feature-based hatch cover category detection method. A fast cover edge feature and descriptor are designed to recognize different types of cover. Cojocaru et al. [18] employed the color space features of the image to segment the assembled part areas. This method is not ideal when there exists a perspective difference between the in-site image and the standard template. To mitigate this, Tsai et al. [19] designed the expectation-maximization algorithm to conduct the image alignment task.

B. DEEP LEARNING BASED METHODS FOR DEFECT INSPECTION

1) DEEP LEARNING BASED METHODS FOR GENERAL OBJECT DETECTION

Compared with the hand-designed feature-based methods, deep neural networks can automatically learn features in images. and are more robust to background, angle of view, and lighting conditions [20]. The deep learning-based object detection network structure are mainly classified into CNN and transformer.

CNN-based object detection methods have demonstrated great advantages on many large scale public datasets. CNN has strong rotation and translation invariant by using

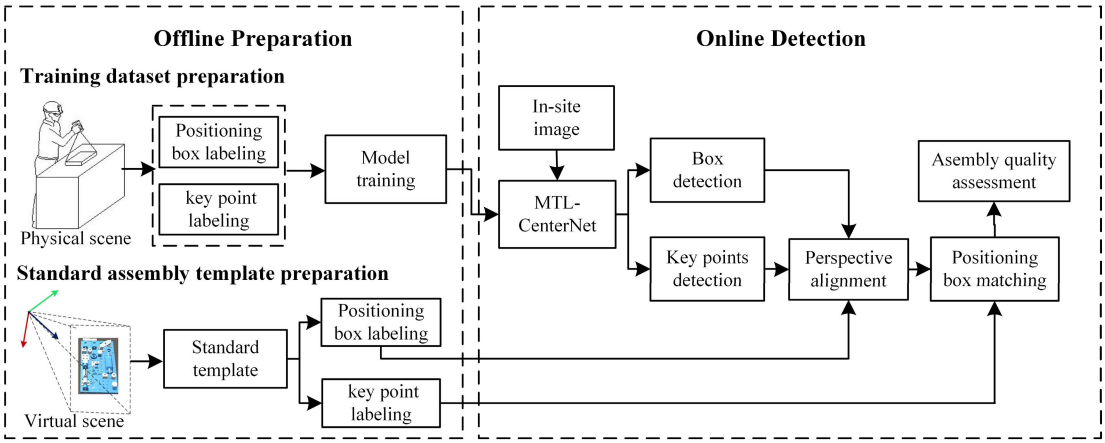


FIGURE 1. Framework of the proposed assembly inspection method.

convolution and pooling operations, which is conducive to the effective identification and classification of objects. Meanwhile, features are extracted by sharing convolution kernel, avoiding redundant calculations. Ren et al. [21] developed a two-stage algorithm called Faster R-CNN. The recommended candidate regions were first generated, and the target detection and localization were then performed. The two-stage detection method is usually time consuming, which is not suitable for the real time target detection. Later, Redmon et al. [22] then proposed a single stage detector, YOLO, to conduct real-time inference. In the process of the above network detection, a large number of candidate boxes would be generated at the same target position. Therefore, Non-Maximum Suppression processing was required [23]. To avoid complex post processing, Zhou et al. [24] proposed Centernet, an object detection model based on fully convolutional network (FCN) [25], where each target was taken as its single central point, so the Non-Maximum Suppression was not required.

In recent years, researchers have introduced transformer architecture from the natural language processing field into vision tasks. Comapared with CNN, transformer has a larger feature receptive field and stronger high-level semantic feature extraction capability, enabling more accurate target positioning [26]. Dosovitskiy et al. [27] proposed a pure transformer-based network, of which images were segmented into small pieces, and a linear embedded sequence of these small pieces was fed to a designed transformer. The full transformer structure has a large number of parameters and high computational cost, which is caused by a significant amount of attention matrix calculations. To mitigate this, some researchers improved the transformer-based visual detection method to reduce the training difficulty and parameters [28]. Further, combined the advantages of CNN in extracting local detail features and transformer in capturing global semantic information, Lewis et al. [29] proposed a hybrid dual network structure named PSNet for Polyp segmentation tasks. Extensive comparative experiments showed that the proposed model achieved the best performance on 5 existing polyp datasets.

2) DEEP LEARNING BASED METHODS FOR SURFACE DEFECT INSPECTION

Currently, inspection methods using deep learning have been applied in many industrial visual defect inspection tasks, such as scratches, chips on glass, cracks and so on [30]. For instance, Cha et al. [31]. proposed a Faster R-CNN-based inspection method to realize the flexible localization and recognition of multiple damages in infrastructures. It is very coarse to utilize positioning boxes to locate damages. Furthermore, to enable pixel-level defect location, Choi et al. [32]. designed a deep semantic segmentation model, SDDNet, to conduct real-time pixel-level defect segmentation, where the deep separable convolution [33] is employed to reduce the computational cost. Later, Kang et al. [34] proposed STRNet to conduct the pixel level crack segmentation task, and the transformer attentionmechanism is employed to ignore the wrong and irrelevant feature information. The proposed network achieved the best performance in crack segmentation and the processing speed achieved 49 FPS on large-size images (1280×720).

Although these schemes have a good performance for the inspection of many surface defects, they cannot be directly applied to the inspection of assembly faults. Assembly faults are mainly concerned with functional defects, such as the missing assembly or the wrong assembly position of objects.

3) DEEP LEARNING BASED METHODS FOR ASSEMBLY FAULT DEFECTS INSPECTION

Some researchers have studied the deep learning-based assembly faults inspection methods. Andrés et al. [35] studied the movement detection of the operator, and SSD algorithm was used to conduct real time monitoring. To solve the assembly quality detection problem of explosion-proof lamp tubes in manual assembly, Riedel et al. [36] designed a system based on YOLOV4, which inspected the assembly omission

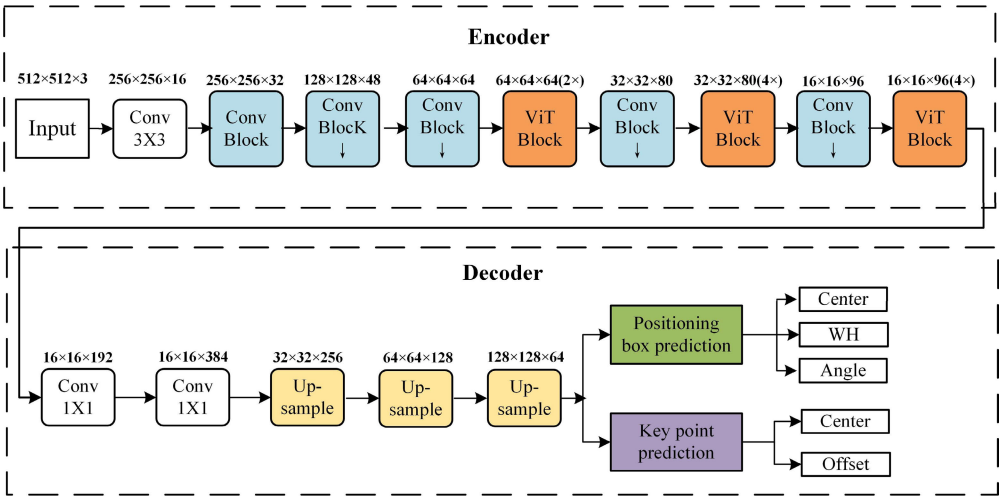


FIGURE 2. Architecture of the MTL-CenterNet neural network.

faults and recorded them, ensuring the traceability of each step. However, the relative pose changes between the camera and the object during assembly process will make inspection difficult. To solve this, Chen et al. [37] proposed an assembly monitoring method, where the depth images are employed to effectively detect the newly assembled parts from different perspectives.

III. HYBRID MULTI-TASK LEARNING MODEL AND POSITIONING BOX MATCHING-BASED ASSEMBLY INSPECTION APPROACH

A. FRAMEWORK OF THE PROPOSED INSPECTION METHOD

In this paper, a deep learning and positioning box matching-based method is developed to realize the online inspection of missing and wrong assembly problems. An encoder-decoder multi-task learning neural network, MTL-CenterNet, is designed to simultaneously detect assembly parts and key points in the in-site image. The framework is shown in Figure 1, including the offline preparation stage and online detection stage.

In the offline preparation stage, a training dataset is generated to train the MTL-CenterNet model. Images in physical assembly scene are captured according to the inspection task. Meanwhile, positioning boxes of parts and four key points on base plate of the assembly object are manual labeled. Using the images and corresponding labels, the MTL-CenterNet model is trained with full supervision.

Another task in offline stage is to establish the standard assembly template. The virtual standard assembly scene is established by CAD software. The standard assembly template of each assembly step is collected using the camera in virtual scene, and the standard positioning boxes and key points information are also marked for the subsequent object matching.

In online detection stage, the assembly quality of each assembly step is assessed. The in-site image of each assembly step is captured via a physical camera, and the positioning boxes of parts and coordinates of key points are detected by the trained MTL-CenterNet. The detected key points are used to align the perspective between the standard template and the in-site image. Then, the positioning box matching algorithm is employed to match the positioning boxes pre-marked in the template with those detected by the MTL-CenterNet. According to the matching results, the assembly quality of the current assembly step can be assessed.

B. INTEGRATED KEY POINT AND PART PREDICTION MODEL

1) ARCHITECTURE OF THE PROPOSED MTL-CENTERNET NEURAL NETWORK

CenterNet [23] is an anchor-free object detection neural network. For each target in the image, CenterNet models it as a single point and generates heatmap to predict the location and size of the object. Inspired by CenterNet, the MTL-CenterNet is proposed in this paper to integrate the task of key point prediction and part positioning box prediction. As shown in Figure 2, MTL-CenterNet includes two parts: encoder and decoder. The input is a RGB image with a size of $512 \times 512 \times 3$, and the output is key point information and positioning box information.

The task of the encoder is to extract semantic features of the assembly objects. Mechanical parts are similar in color, lack of texture and may have changes of rotation and translation during the assembly process, which brings challenges to the feature extraction, location and recognition of parts. To improve the detection accuracy, the transformer-based attention mechanism is integrated into the MobileNetV2 [38] backbone as an improved feature encoder. The CNN layer and transformer layer are placed alternately, and transformer layer is placed in the back half of encoder.

The reason for alternating CNN and transformer layer is to downsample the feature map via convolution and pooling operations in CNN layer, reducing the subsequent computational complexity for transformer layer. Meanwhile, CNN layer is also used to raise the dimension of features, and the global attention mechanism of transformer layer is then leveraged to extract high dimensional semantic features of the input image. Therefore, compared with pure CNN, the improved hybrid encoder has stronger feature extraction and modeling capabilities. Compared with pure transformer, the improved hybrid encoder has lower computational cost and fewer parameters. The output of the encoder is a $16 \times 16 \times 96$ feature map.

Then the decoder part is used to reconstruct the feature map. In the decoder part, the channel dimension of the feature map is firstly adjusted by two 1×1 convolution layers. To realize the prediction from pixel to pixel, deconvolution operation is then employed to up-sample the feature map for three times. A feature map with size $128 \times 128 \times 64$ is finally obtained. Finally, according to our detection task, the key point detection head and part detection head are set as output layers respectively. They share the weight parameters except for the detection head, forming a multi-task learning network, which greatly saves network parameters and processing time.

For the key point prediction head, two parallel convolution layers are used as the output layer to predict the center and offset heatmap of the key points respectively. The output size of the "Center" heatmap branch is $128 \times 128 \times 4$, where four different channels are employed to predict four corresponding key points. To reduce the coordinate error caused by feature downsampling operation, we set another branch to regress the coordinate offset of each key point as compensation. The output size of the "Offset" heatmap branch is $128 \times 128 \times 2$, and 2 is the two offsets of the key point in the width and height direction.

The ground truth of each key point is represented by a gaussian circle, with the key point coordinate as the center of this circle. The gaussian distribution is used to generate the values within the circle, and the values outside the circle are set to 0. The Gaussian probability distribution of the key point is calculated as (1):

$$P_{xyc} = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(x-u_x)^2 + (y-u_y)^2}{2\sigma^2}) \tag{1}$$

where (u_x, u_y) is the coordinate of the key point, (x, y) is the coordinate in the Gaussian circle. σ is the variance, where 1/3 of the radius is taken.

For the prediction head of the positioning box, three parallel convolution layers, "Center", "WH" and "Angle", are used as the output layer to predict the heatmap of the center, size and rotation angle of the positioning box. The size of the "Center" heatmap is $128 \times 128 \times N$, where N is the number of types of parts. Similar to the key point prediction task, the ground truth of the "Center" heatmap is also represented by a Gaussian circle. The "WH" heatmap branch represents the width and height of the detection box, and its size is $128 \times 128 \times 2$.

The commonly used horizontal positioning box contains the position and the approximate size information of the part. During the assembly process, when the parts are tilted or rotated, it is hardly to fit the part well, especially for those parts with large aspect ratio. Considering that, we add the rotation information to make the detected box more fit the part. The rotation information is defined as the angle between the longer side of the part and the y-axis. Figure 3 shows the angle definition of the rotated positioning box. The range of angle is defined as $[0, \pi)$.

Finally, to train the MTL-CenterNet detector, for the "Offset", "WH" and "Angle" branch, the L1 loss is used. To solve the problem of unbalanced part category, the penalty-reduced regression with focal loss is employed to calculate the loss of the "Center" branch. The expression is as follows:

$$Loss = -\frac{1}{M} \sum_{xyc} \begin{cases} (1-\hat{Y_{xyc}})^{\alpha} \log\left(\hat{Y_{xyc}}\right) & if\ Y_{xyc} = 1 \\ (1-Y_{xyc})^{\beta} \log(\hat{Y_{xyc}})^{\alpha} \log\left(1-\hat{Y_{xyc}}\right) & otherwise \end{cases} \tag{2}$$

where M is the total number of labeled key points in one training image, and α, β are hyper parameters, selected as 2 and 4 respectively. Y_{xyc} and $\hat{Y_{xyc}}$ are the actual and the predicted coordinate of key point.

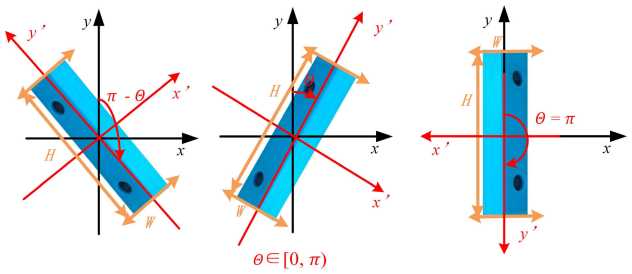


FIGURE 3. Rotated positioning box.

2) HYBRID ENCODER COMBINING CNN AND TRANSFORMER

As shown in Figure2, the encoder is mainly made up of a stack of Conv block and Vit block. The Conv block extract features through CNN layer, and the same residual block structure as MobileNetV2 is adopted. The Vit block is mainly made up of transformer, which is placed in the back half of the encoder to extract higher level semantic features.

The Vit block is shown in Figure 4. For a $H \times W \times C$ feature map, patch embedding is applied firstly. A 1×1 convolution is used to project the dimension of the feature map to a higher dimension of $H \times W \times d$. The feature map is then expanded with the dimension of $(h \times w) \times N \times d$. Then, it is fed into the transformer attention module for

feature extraction, and the dimension of the obtained feature map is still $(h \times w) \times N \times d$. Then, the feature map is folded into the dimension of $H \times W \times d$ and projected back to the original $H \times W \times C$ dimension through a 1×1 convolution.

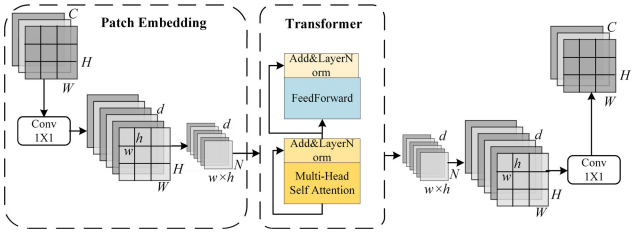


FIGURE 4. Vision transformer (vit) block.

The transformer structure includes the multi-head self-attention (MSA) and feed forward modules, with residual connection and layer normalization used after each block. The feed forward module is a multilayer perception (MLP) block that projects the dimension of the feature map to four times the original dimension and then projects back to the original dimension.

The structure of the MSA module is shown in Figure 5 (a). For the input feature map, a series of MLP operations are first performed to project the input to multiple subspaces, obtaining multiple single-head feature maps. Then, each group of single-head feature map is multiplied with three learnable attention matrixes respectively, obtaining three quantities Q_i, K_i and V_i. And the scaled self-attention layer is then employed to purify the single-head feature map. Finally, the obtained multiple groups of single-head feature maps are concatenated together and once again projected by MLP layers.

The detailed computation process of the scaled self-attention module is shown in Figure 5 (b). The calculation formula is as (3):

$$Head_i = softmax(\frac{Q \cdot K^T}{\sqrt{d}_K})V \tag{3}$$

where $Head_i$ is the output single-head feature map, and d_k is the scale factor and its value is the variance of the $Q \cdot K^T$.

C. TRANSFORMATION MATRIX CALCULATION AND POSITIONING BOX MATCHING

During the assembly process, the position or rotation angle of the product on the workbench usually changes with assembly operations. Hence, there is a difference in perspective between the pre-collected standard assembly template and the in-site image, which will cause error for the subsequent positioning box matching to inspect assembly faults.

Based on the perspective transformation theory, to align the in-site image and the standard template, at least four points located in the same plane are needed to be selected to establish the transformation relation. Specially, during the assembly process, the selected four key points should not be occluded. The type and location of the four key points in the

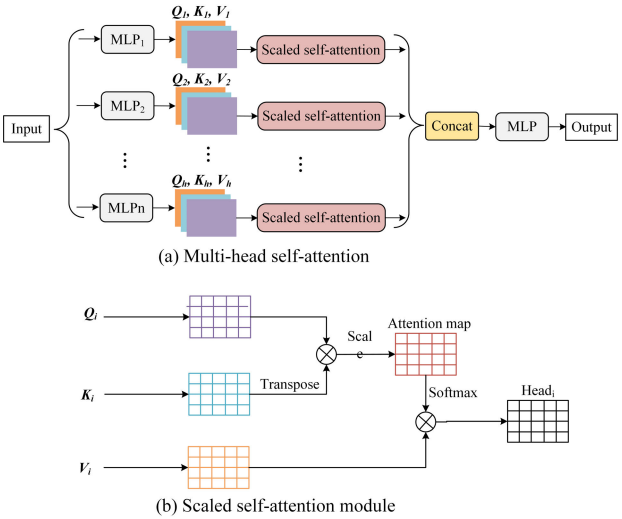


FIGURE 5. Multi-head self–attention (MSA) module.

standard assembly template can be marked and measured in advance. Meanwhile, the corresponding four key points in the in-site image can be predicted by the proposed MTL-CenterNet. The maximum value in each output channel of the "Center" heatmap branch represents the predicted key point, and four channels of the center heat map are responsible for detecting four different key points respectively. According to the detected four key points from the in-site image and the corresponding key points pre-set in the standard template, the transformation matrix can be calculated by (4):

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4}$$

where (x, y) is key point coordinate predicted by the MTL-CenterNet from in-site image, (x', y') is the corresponding coordinate on the standard assembly template.

Based on transformation matrix, the in-site image is aligned with the standard template. Then the missing and wrong assembly faults are inspected by the positioning box matching algorithm. The matching algorithm calculates the intersection over union (IoU) between the detected positioning box and the standard box to determine whether the corresponding relation between the actual assembly position and the standard position is correct. Figure 6 shows the actual positioning box and standard template box of the current assembly part. The IoU matching formula is as (5):

$$IoU_{2D} = \frac{S_r \cap S_v}{S_r \cup S_v} \tag{5}$$

where S_r is the positioning box in the standard assembly template, and S_v is the detected box.

The positioning box matching algorithm contains the following steps.

Step 1: Select a part assembled in this step from the template, named A.

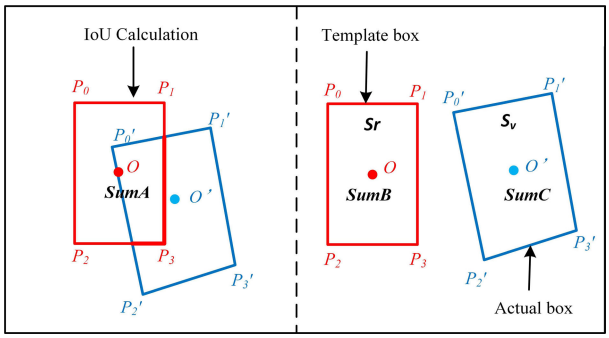


FIGURE 6. Actual positioning box and standard template box of the current assembly part.

Step 2: The standard positioning box of A is expressed as (O, P_0, P_1, P_2, P_3), where O is the central point and P_0, P_1, P_2, P_3 is the four vertices. Search the positioning box of the part intersecting with A at the corresponding position of the in-site image, denoted as Ai. If it is not searched, the current step exists a missing assembly problem.

Step 3: According to (5), calculate the IoU of A and Ai. If the IoU is less than the preset threshold value, the current step exists a wrong assembly problem.

Step 4: Judge the category of A and Ai. If they are of the same type, this step can be considered as qualified assembly. Otherwise, the current step exists an assembly type fault.

Step 5: Repeat the above steps for all parts assembled under the current assembly procedure in the template.

In this paper, the positioning box matching threshold is set to 0.65. When the IoU matching result is less than the preset threshold, the current assembly step is judged to unqualified.

IV. EXPERIMENTS AND ANALYSIS

A. TEMPLATE OBTAINING AND NEURAL NETWORK TRAINING

In this section, the public **mo**del for ma**n**ual **a**ssembly (MONA) [39], is employed for following inspection experiments. The length, width and height of MONA are 250 mm, 150 mm and 50 mm respectively, consisting of a bottom plate, 16 parts and 18 assembly steps. We use 3D printing to make corresponding assembly parts. Eight main parts in MONA are selected for the subsequent experiments, namely TR_M6_Corner, BR_Column, Square, Cylinder, TM_Attachment, TL_M4_Hole, TL_M4_Rest and BL_Block_Top, which consists of 10 assembly steps in total. For template images obtaining, we use the virtual camera in CAD to collect images of each assembly step in the standard assembly state. For each template, the key points and standard positioning box information are manually marked. Partial template images and annotated information are shown in Figure 7.

To train the MTL-CenterNet network, 240 assembly images in the ten selected assembly steps are collected as the training dataset of MONA assembly, and 60 images are collected as the test dataset. These images are obtained from different angles and under different lighting conditions in real assembly backgrounds, as shown in Figure 8. Each image is manually labeled with key points and positioning boxes information. In addition, four common data enhancement methods are used to augment the training data, including image rotation, image flip, adding noise, and color space transformation. The final number of images in training datasets is 1200. The detailed information of the collected dataset is shown in Table 1.

TABLE 1. The prepared dataset for train and test.

	Train	Test	Total
Sizes of images	1024x1024, 1024x768	1024x768, 512x512	1024x1024, 1024x768, 512x512
Number of images	240	60	300

The neural network training and subsequent test experiments are performed on a Windows 10 system, equipped with a NVIDIA Quadro P4000 GPU and an Intel Xeon e5-2630 CPU. The input training images are resized to 512 × 512 x 3. The training batch size, weight decay, momentum is set to 4, 5e-4, 5e-4 respectively. The starting learning rate is 1e-3, dropping to one tenth of the original value every 50 rounds, and the minimum is 1e-5.

B. KEY POINTS DETECTION AND ERROR ANALYSIS

1) PERFORMANCE OF KEY POINTS DETECTION

To verify the effectiveness of the MTL-CenterNet model for key points detection, four points, P_0, P_1, P_2 and P_3 on the base plate of MONA assembly are selected, as shown in Figure 9(a). During the assembly operation, the selected four points should not be occluded, thus ensuring that the algorithm can correctly solve the transformation relation between the assembly template and in-site image. The detected heatmap and the corresponding coordinates are shown in Figure 9(b) and Figure 9(c), respectively. As shown in Figure 9(b), the channel dimension of the output heatmap is 4, corresponding to 4 key points.

Table 2 shows the predicted coordinates of key points. Here, x and y represent ground truth, and x_0, y_0 represent the predicted coordinates. Δx and Δy are errors in X direction and Y direction, respectively. According to Table 1, the average pixel error in X direction is 0.975, and in Y direction is 1.

2) ERROR EVALUATION OF KEY POINT DETECTION

During the process of assembly, the rotation angle will change with the movement of the assembly product on workbench. Meanwhile, due to the change of the relative distance between the camera and the assembly product, contents on the in-site image also have different scales. Hence, different rotation angles and grade of scale variations of the

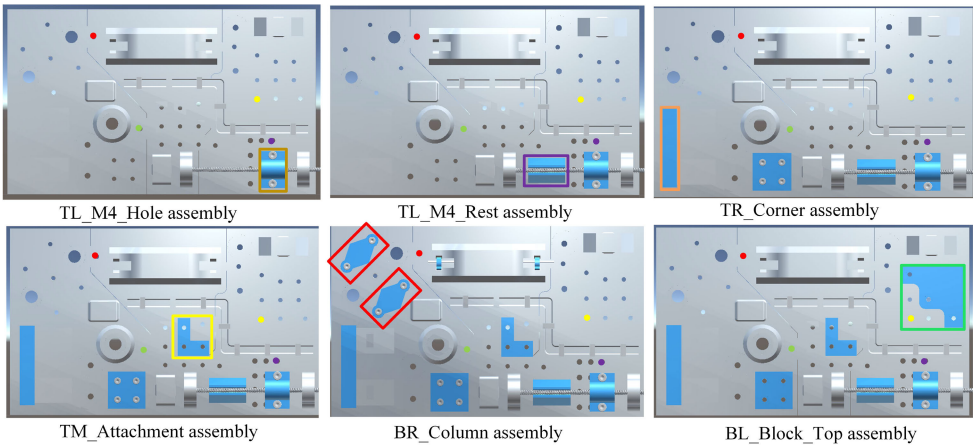


FIGURE 7. Template obtaining for partial assembly steps.

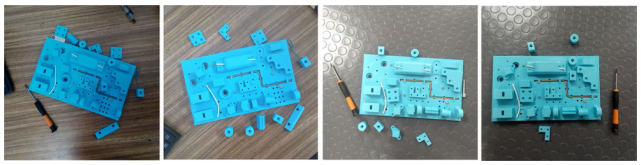


FIGURE 8. Example of image data for training

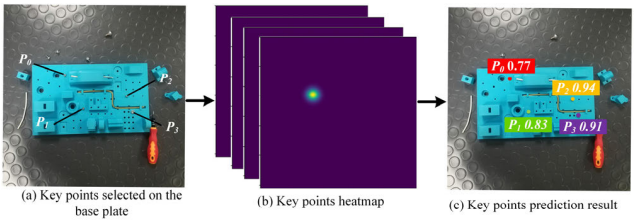


FIGURE 9. Key point detection.

TABLE 2. The pixel coordinates of key points.

Points	x	y	x_0	y_0	Δx	Δy
P_0	172.5	198.3	173	197	0.5	1.3
P_1	224.2	292.1	225	293	0.8	0.9
P_2	344.6	257.8	347	257	2.4	0.8
P_3	365.2	302.0	365	301	0.2	1.0

assembly object will affect the detection accuracy. In this section, the proposed MTL-CenterNet is also compared with SIFT, SURF, ORB and SuperPoint [40] methods, which are commonly used for key points detection. The SIFT, SURF and ORB are traditional feature points detection methods based on descriptors, while SuperPoint is based on deep learning. The Root mean square error (RMSE) is adopted to calculate the pixel error.

$$RMSE = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} (P_i - \bar{P}_i)} \tag{6}$$

where N is the number of key points, P_i is the coordinate of the detected key point and $\bar{P}_i$ is the ground truth.

For the influence of different degrees of rotation, as shown in Figure 10, eight images are collected and tested, and the center of MONA rotates 0°, 20°, 40°, 60°, 80°, 100°, 120°, and 160° in the counterclockwise direction respectively. To eliminate interference, the image content collected only contains the baseplate of MONA, and no other interferences are set.

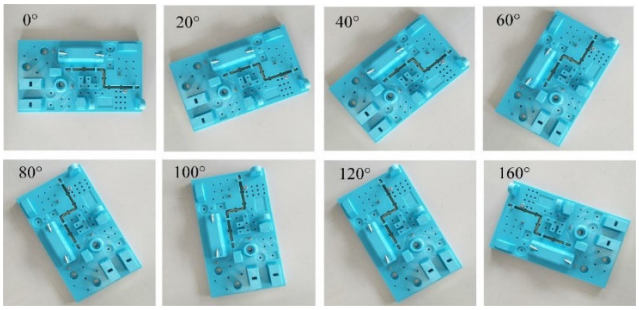


FIGURE 10. Different rotation angles.

The obtained average pixel error of each detection method is shown in Figure 11. For different degrees of rotation, the key points detection errors of our method are relatively stable. The mean pixel error is about 1.5 pixels. SuperPoint shows high accuracy when the rotation angle is less than 60 degrees, and its error is about 0.5 pixels. But when the rotation angle is greater than 60 degrees, its detection error increases rapidly, about 6 pixels. Similarly, the ORB detector shows a poor accuracy and stability for large rotation, with a maximum error of about 7 pixels when the rotation is greater than 80 degrees. SIFT and SURF have relatively stable accuracy, and their results are about 1 pixel against different rotation angles. In conclusion, our method shows a high detection accuracy under different rotations, which is superior to ORB based on descriptors and SuperPoint based on FCN.

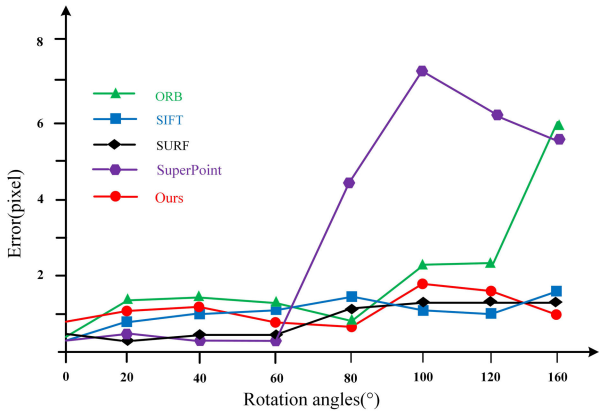


FIGURE 11. Pixel error at different rotation angles.

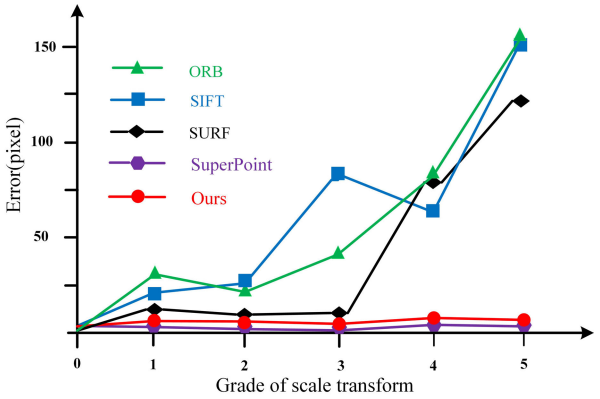


FIGURE 13. Pixel error at different grade of scale variations.

To compare the scale invariance of different algorithms, Gaussian blur is applied to the acquired images to simulate the scale variation of the images caused by the camera shooting distance. As shown in Figure 12, the grade of scale variation is controlled by the size of Gaussian convolution kernel. The larger the convolution kernel is, the more blurred the image is, which means that the camera is farther away from the assembly object. The convolution kernel varies from 1 to 11, with a step size of 2, and is divided into 6 levels in total.

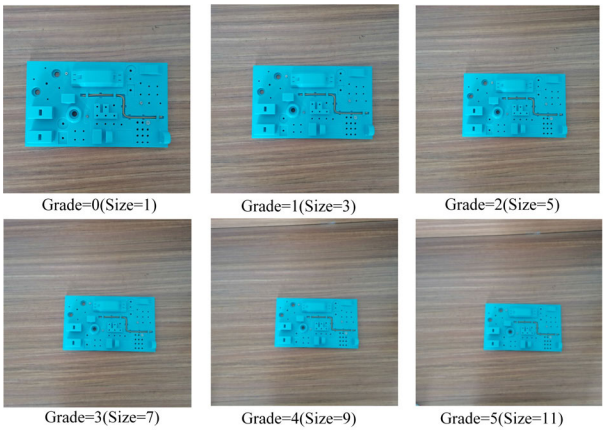


FIGURE 12. Different grade of scale transform.

The average pixel error at different scales is shown in Figure 13. When the roughness of the image increases, the proposed 8 algorithm and SurperPoint show good stability. For different grades of roughness, the pixel error of the proposed method is stable at 1.5 pixels, and the SuperPoint is stable at about 0.9 pixels. For the descriptor-based detection method, the error increases with the increase of image roughness level. The pixel error of the SIFT and ORB detectors is up to 150 pixels, and SURF is also over 120 pixels.

The detection speed has an important influence on whether it can provide real time feedback of assembly inspection for operators. The average time and frame rate of these algorithms are also tested. For fair comparison, the size of the input image of different methods is unified to 512 × 512 x 3, and the result is shown in Table 3. The methods based on descriptors are more time-consuming, while the deep learning methods are fast. The frames per second (FPS) of the algorithm proposed in this paper can reach 35.7 for the 512 × 512 size of the input image, which can detect key points in real time.

TABLE 3. Key points detection speed comparison.

Algorithms	ORB	SIFT	SURF	SuperPoint	**Ours**
Time consumption (ms)	112.74	271.54	185.10	34.50	**28**
Frame rate (FPS)	8.87	3.68	5.40	28.98	**35.7**

C. ASSEMBLY PARTS DETECTION

1) COMPARISON WITH OTHER DETECTION METHODS

The test dataset is selected to verify the assembly parts detection performance of the MTL-CenterNet model. These assembly parts included in the test dataset have similar colors, different sizes and lack of texture. Partial detection results are shown in Figure 14, where different color boxes represent different parts types. The MTL-CenterNet is also compared with two types of mainstream algorithms in parts detection, including the Faster-RCNN, Yolo-V5s and SSD based on anchor and CenterNet without anchor. For fair comparison, the input size of different methods is also unified to 512 × 512 x 3. The results of various algorithms are shown in Table 4, which includes the detection accuracy, the number of model parameters, the detection time and frame rate. The detection accuracy is evaluated by the mean intersection over union (mIoU) and the mean average precision (MAP). The computational complexity is also measured by the index of floating point operations (FLOPs).

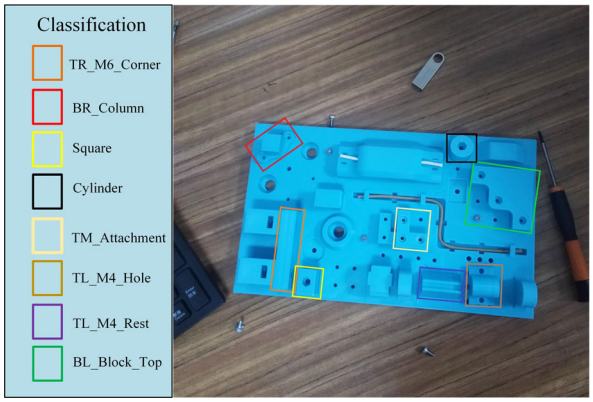


FIGURE 14. Sample test results of partial parts.

In terms of detection accuracy, the proposed MTLCenterNet achieves the highest mIoU and MAP of 93.13and 97.66%, respectively. And the algorithm has the lowest number of parameters. This means that our model is more lightweight, which is ideal for deployment on edge devices with limited computing resources. In terms of detection speed, Yolo-v5s achieves the fastest detection speed, but our algorithm is only 4.22ms slower and the MAP value exceeded it by 5.29%. Meanwhile, the computational complexity of our model is also relatively low in compared ones.

TABLE 4. Performance of parts detection.

Method	mIoU (%)	$MAP_{0.5}$ (%)	Params (M)	Time (ms)	FPS	FLOPs (G)
Faster-RCNN	91.37	95.56	521.97	169	5.92	184.94
Yolo-V5s	89.17	92.37	27.01	**21.10**	**47.39**	**10.58**
SSD	88.81	90.51	93.64	22.90	43.6	24.54
CenterNet	90.62	94.73	124.61	24.50	40.82	35.10
Ours	**93.13**	**97.66**	**19.54**	**25.32**	**39.49**	**13.12**

2) K-FOLD RANDOM VALIDATIONS

To determine whether the trained model has the problem of overfitting, 10-fold cross validation experiment is carried out. In each round of experiment, 10% of the data is randomly selected from the train dataset, test dataset and total dataset respectively. The final trained model is tested on these data, and the mIoU is used as the evaluation index. Results are shown in Table 5. Each column represents the mIoU obtained from each round of experiment. After 10 round experiment, the average mIoU of the trained model in the training dataset, test dataset and total dataset are 94.07%, 93.12% and 93.57%, respectively. The final performance we get in Table 4 is 93.13%. The cross validation results are very close to the claimed performance, so it is found that no overfitting occurs in the trained MTL-CenterNet model.

In order to fully test the performance generalization ability of different detection methods on the test dataset, we also carried out the 10-fold cross-validation. In each round of validation, 10% of the data is randomly selected from our test dataset, which is then used to test our detection method and the other four comparative modes respectively. The validation results are shown in Table 6. It can be seen that our method achieves the highest mIoU in 7 out of 10 rounds of validation. The average mIoU is 93.22%, which is very close to 93.13% we obtained using the total test dataset. The above two 10-fold cross validation experimental results demonstrate that our test dataset are not specifically prepared to achieve a higher performance, and our designed model has good generalization.

TABLE 5. 10-fold random validation for the trained model (%).

Round	Train data	Test data	Total data
R1	93.56	92.54	93.13
R2	93.29	93.13	93.01
R3	94.33	93.15	93.83
R4	94.64	92.66	93.78
R5	93.78	94.20	94.15
R6	94.68	93.13	93.95
R7	93.59	94.20	93.72
R8	94.02	92.66	93.15
R9	94.75	92.54	93.52
R10	94.02	92.97	93.43
Average	94.07	93.12	93.57

TABLE 6. 10-fold cross validation for different methods on test dataset (%).

Round	Faster-RCNN	Yolo-V5s	SSD	CenterNet	**Ours**
R1	90.28	90.74	86.63	89.80	**93.28**
R2	92.66	88.21	90.67	89.02	**94.98**
R3	92.66	90.21	86.47	92.80	**93.14**
R4	**91.38**	88.45	89.46	89.16	91.02
R5	90.37	89.01	89.74	90.63	**92.49**
R6	90.86	87.19	91.72	89.16	**94.98**
R7	90.98	90.24	89.43	**93.43**	92.56
R8	91.46	88.79	90.40	89.67	**92.38**
R9	89.64	90.35	88.35	**93.67**	93.27
R10	91.40	88.49	86.63	90.63	**95.03**
Average	91.17	89.17	88.95	90.80	**93.22**

3) EXPERIMENTS FOR THE IMPROVED HYBRID ENCODER

To verify the performance of the improved hybrid encoder, the proposed hybrid encoder is compared with the pure Transformer or CNN encoders, including the Swin- Transformer [28] and Resnet50 [41], and the original MobileNetV2 [38] is also leveraged as the baseline. The experimental results are shown in Table 7.

In terms of detection accuracy, the Swin-Transformer achieves the best performance, while the number of param-

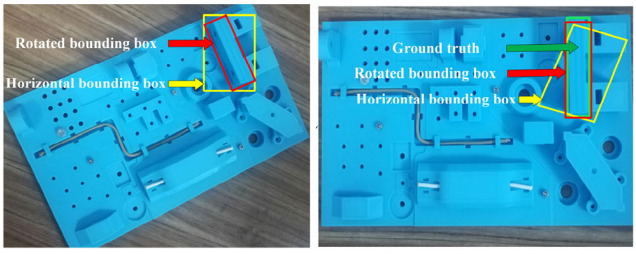
eters and computational complexity are also higher, which are 12 times and 4 times of ours respectively. Meanwhile, with only about 8% of the parameters and 25% of the computational cost of the Swin-Transformer backbone, our mIoU and MAP performance lagged by only 0.48% and 0.73%, respectively. The MobileNetV2 baseline achieves the fastest detection speed and lowest computational complexity, but the mIoU and the MAP is 5.25% and 4.29% lower than ours respectively. As a whole, our improved hybrid encoder achieves a good balance between the detection accuracy and the computational cost.

TABLE 7. Performance of parts detection with different backbone.

Backbone	mIoU (%)	$MAP_{0.5}$ (%)	Params (M)	Time (ms)	FLOPs (G)
Swin-Transformer	**93.61**	**98.39**	27.49	43.46	22.85
Resnet50	90.97	94.12	23.51	25.30	20.37
MobileNetV2	87.88	93.37	**2.22**	**16.08**	**1.70**
Ours	**93.13**	**97.66**	**2.28**	**25.32**	**5.7**

4) COMPARATIVE EXPERIMENT OF DIFFERENT TYPES OF POSITIONING BOX

Experiments are conducted to compare the positioning accuracy of different types of box for the part with large aspect ratio. As shown in Figure 15, the TR _Corner part in MONA, is chosen to demonstrate the detection results of the rotated box and horizontal box. Figure 15(a) is the image taken by the camera, and Figure 15(b) is the image obtained according to the transformation relation between the standard template and in-site image. The experimental results show that the horizontal detection box contains a lot of background information, while the rotated box can effectively fit the parts, improving the accuracy of subsequent object matching.



(a) In-site image (b) Image after transformation

FIGURE 15. Detection results of different types of positioning boxes.

D. MISSING AND WRONG ASSEMBLY FAULTS INSPECTION

In this section, the assembly inspection method is used to carry out the missing and wrong installation faults inspection experiment, and the workbench is shown in Figure 16. For each assembly step of MONA, sample images of qualified assembly and unqualified assembly are collected respectively, and the assembly states in these images are recorded. A total of 150 sample images are collected, including qualified, 8 missing and wrong assembly, and the number of each assembly sample is 50. The types of wrong assembly consist three common cases, part type error, angle error and position error. The precision, recall and F1 score are used in this paper as evaluation indexes for assembly fault inspection. Based on the positioning box matching algorithm, the IoU value between the detected positioning box and the standard box is calculated. When the IoU result is less than the preset threshold, it's judged that the current step has a missing or wrong assembly error.

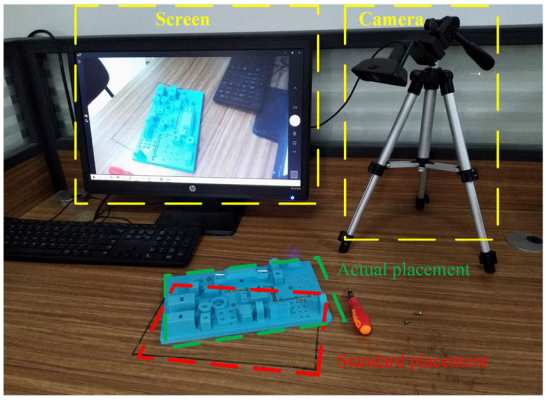
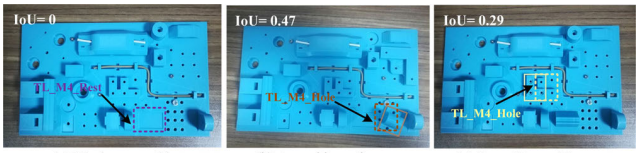


FIGURE 16. Assembly workbench.

After testing all samples, the final precision, recall and F1 score of three types of assembly samples detected by the proposed inspection method are calculated respectively, and the result is shown in Table 7. In Table 8, the qualified assembly samples, missing assembly samples and wrong assembly samples are denoted as "Qualified", "Missing", and "Wrong", respectively. The proposed inspection method has achieved the F1 score of 90.91% for qualified assembly, 97.09for missing assembly and 93.88% for wrong assembly. The inspection result of some samples is shown in Figure 17. The dashed line box is the real position of the part, and the solid line part is the bounding box detected by the proposed part detection model. Experimental results demonstrate that the proposed method can effectively inspect the assembly faults for each assembly step.



(a) Missing assembly (b) Assembly angle error (c) Assembly position error

FIGURE 17. Assembly quality inspection samples.

TABLE 8. Assembly quality inspection result.

Assembly type	Precision (%)	Recall (%)	F1 (%)
Qualified	91.84	90.00	90.91
Missing	94.34	100	97.09
Wrong	95.83	92.00	93.88

V. CONCLUSION

In this paper, an online deep learning-based inspection scheme is proposed to inspect assembly faults. A novel multi-task learning model with transformer attention mechanism, named MTL-CenterNet, is proposed to integrate key points detection and parts detection. The detected key points are used to solve the transformation relation between the in-site image and the template image. Based on this, the detected positioning box is matched with the standard positioning boxes to judge whether there are missing or wrong assembly faults. To verify the proposed model, key points and parts detection experiments are carried out, and the results are superior to other mainstream algorithms. In addition, the assembly inspection experiment is also conducted. The results show that the average F1score for missing and wrong assembly error identification reaches 93.96%.

The proposed deep learning-based assembly inspection method still requires a large amount of manually annotated training data, which will limit its further application in industry. Future works will focus on unsupervised or weakly supervised detection methods to reduce the burden of datasets preparation. Moreover, the structure of the MTL-CenterNet model will be further optimized to reduce the computational complexity.

REFERENCES

[1] Y. Torres, S. Nadeau, and K. Landau, "Classification and quantification of human error in manufacturing: A case study in complex manual assembly," *Appl. Sci.*, vol. 11, no. 2, p. 749, Jan. 2021.

[2] C. Chen, T. Wang, D. Li, and J. Hong, "Repetitive assembly action recognition based on object detection and pose estimation," *J. Manuf. Syst.*, vol. 55, pp. 325–333, Apr. 2020.

[3] R. Iqbal, T. Maniak, F. Doctor, and C. Karyotis, "Fault detection and isolation in industrial processes using deep learning approaches," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3077–3084, May 2019.

[4] H. F. Le, L. J. Zhang, and Y. X. Liu, "Surface defect detection of industrial parts based on YOLOv5," *IEEE Access*, vol. 10, pp. 130784–130794, 2022.

[5] Y. Fu, X. Ma, and H. Zhou, "Automatic detection of multi-crossing crack defects in multi-crystalline solar cells based on machine vision," *Mach. Vis. Appl.*, vol. 32, no. 3, p. 60, May 2021.

[6] X. Jiang and S. Wang, "Railway panorama: A fast inspection method for high-speed railway infrastructure monitoring," *IEEE Access*, vol. 9, pp. 150889–150902, 2021.

[7] Y. Wang, M. Perry, D. Whitlock, and J. W. Sutherland, "Detecting anomalies in time series data from a manufacturing system using recurrent neural networks," *J. Manuf. Syst.*, vol. 62, pp. 823–834, Jan. 2022.

[8] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, Mar. 2017.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[10] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.

[12] Y. Liu, S. Li, J. Wang, H. Zeng, and J. Lu, "A computer vision-based assistant system for the assembly of narrow cabin products," *Int. J. Adv. Manuf. Technol.*, vol. 76, nos. 1–4, pp. 281–293, Jan. 2015.

[13] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[14] F. Da and H. Zhang, "Sub-pixel edge detection based on an improved moment," *Image Vis. Comput.*, vol. 28, no. 12, pp. 1645–1658, Dec. 2010.

[15] M. Kim, W. Choi, B.-C. Kim, H. Kim, J. H. Seol, J. Woo, and K. H. Ko, "A vision-based system for monitoring block assembly in shipbuilding," *Comput.-Aided Des.*, vol. 59, pp. 98–108, Feb. 2015.

[16] C. Rother, V. Kolmogorov, and A. Blake, "'GrabCut': Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, 2004.

[17] X. Yang, X. Fan, J. Wang, X. Yin, and S. Qiu, "Edge-based cover recognition and tracking method for an AR-aided aircraft inspection system," *Int. J. Adv. Manuf. Technol.*, vol. 111, nos. 11–12, pp. 3505–3518, Dec. 2020.

[18] J.-I.-R. Cojocaru, D. Popescu, and L. Ichim, "Real-time assembly fault detection using image analysis for industrial assembly line," in *Proc. 43rd Int. Conf. Telecommun. Signal Process. (TSP)*, Jul. 2020, pp. 484–487.

[19] D.-M. Tsai and Y.-C. Hsieh, "Machine vision-based positioning and inspection using expectation–maximization technique," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 11, pp. 2858–2868, Nov. 2017.

[20] P. Tang, Y. Guo, H. Li, Z. Wei, G. Zheng, and J. Pu, "Image dataset creation and networks improvement method based on CAD model and edge operator for object detection in the manufacturing industry," *Mach. Vis. Appl.*, vol. 32, no. 5, Sep. 2021.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[23] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5562–5570.

[24] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, *arXiv:1904.07850*.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[26] N. Carion, F. Massa, and G. Synnaeve, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 2020, pp. 213–229.

[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[28] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.

[29] J. Lewis, Y.-J. Cha, and J. Kim, "Dual encoder–decoder-based deep polyp segmentation network for colonoscopy images," *Sci. Rep.*, vol. 13, no. 1, pp. 1–12, Jan. 2023.

[30] X. Tao, X. Gong, X. Zhang, S. Yan, and C. Adak, "Deep learning for unsupervised anomaly localization in industrial images: A survey," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–21, 2022.

[31] Y. Cha, W. Choi, G. Suh, S. Mahmoudkhani, and O. Büyüköztürk, "Autonomous structural visual inspection using region-based deep learning for detecting multiple damage types," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 33, no. 9, pp. 731–747, Sep. 2018.

[32] W. Choi and Y.-J. Cha, "SDDNet: Real-time crack segmentation," *IEEE Trans. Ind. Electron.*, vol. 67, no. 9, pp. 8016–8025, Sep. 2020.

[33] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.

[34] D. H. Kang and Y.-J. Cha, "Efficient attention-based deep encoder and decoder for automatic crack segmentation," *Struct. Health Monitor.*, vol. 21, no. 5, pp. 2190–2205, Sep. 2022.

[35] M.-A. Zamora-Hernández, J. A. Castro-Vargas, J. Azorin-Lopez, and J. Garcia-Rodriguez, "Deep learning-based visual control assistant for assembly in industry 4.0," *Comput. Ind.*, vol. 131, Oct. 2021, Art. no. 103485.

[36] A. Riedel, J. Gerlach, M. Dietsch, S. Herbst, F. Engelmann, N. Brehm, and T. Pfeifroth, "A deep learning-based worker assistance system for error prevention: Case study in a real-world manual assembly," *Adv. Prod. Eng. Manage.*, vol. 16, no. 4, pp. 393–404, Dec. 2021.

[37] C. Chen, C. Li, D. Li, Z. Zhao, and J. Hong, "Mechanical assembly monitoring method based on depth image multiview change detection," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–13, 2021.
[38] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[39] J. Blattner, J. Wolfartsberger, and R. Lindorfer, "A standardized approach to evaluate assistive systems for manual assembly tasks in industry," in *Proc. Conf. Learn. Factories. (CLF)*, 2021, pp. 1–6.
[40] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 224–236.
[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

SHIWEN ZHAO received the B.E. degree in aerocraft manufacture engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2021. He is currently pursuing the M.S. degree in mechanical engineering with the Huazhong University of Science and Technology, Wuhan, China. His research interests include deep learning and machine vision.

JUNFENG WANG (Member, IEEE) received the B.E. degree in engineering from Beijing Jiaotong University, Beijing, China, in 1991, and the Ph.D. degree in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2004. From 2009 to 2010, he was a Visiting Scholar with the S. M. Wu Manufacturing Research Center, University of Michigan, Ann Arbor. He is currently a Professor and the Dean of the Department of Industrial and Manufacturing System Engineering, School of Mechanical Science and Engineering, Huazhong University of Science and Technology. His research interests include production systems, industrial augmented reality, and human–robot collaborative assembly. He is a member of IISE and CMES.

WANG LI received the B.E. degree in mechanical design manufacture and automation from Weifang University, Weifang, China, in 2013, the M.S. degree in mechanical engineering from Central South University, Changsha, China, in 2016, and the Ph.D. degree in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2022. He is currently a Postdoctoral Researcher with the Department of Information Technology Equipment, FiberHome Telecommunication Technologies Company Ltd., Wuhan. His research interests include augmented reality and artificial intelligence.

LONGFEI LU received the B.E. degree in industrial engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2022, where he is currently pursuing the M.S. degree with the Department of Industrial and Manufacturing System Engineering, School of Mechanical Science and Engineering. His research interests include deep learning and automatic assembly.