Pixel-Wise Symbol Spotting via Progressive Points Location for Parsing CAD Images

Junbiao Pang, Zailin Dong, Jiaxin Deng, Mengyuan Zhu and Yunwei Zhang

Abstract—Parsing Computer-Aided Design (CAD) drawings is a fundamental step for CAD revision, semantic-based management, and the generation of 3D prototypes in both the architecture and engineering industries. Labeling symbols from a CAD drawing is a challenging yet notorious task from a practical point of view. In this work, we propose to label and spot symbols from CAD images that are converted from CAD drawings. The advantage of spotting symbols from CAD images lies in the low requirement of labelers and the low-cost annotation. However, pixel-wise spotting symbols from CAD images is challenging work. We propose a pixel-wise point location via Progressive Gaussian Kernels (PGK) to balance between training efficiency and location accuracy. Besides, we introduce a local offset to the heatmap-based point location method. Based on the keypoints detection, we propose a symbol grouping method to redraw the rectangle symbols in CAD images. We have released a dataset containing CAD images of equipment rooms from telecommunication industrial CAD drawings. Extensive experiments on this real-world dataset show that the proposed method has good generalization ability.

Index Terms—CAD images parsing, symbol spotting, pixel-wise localization, Progressive Gaussian Kernels
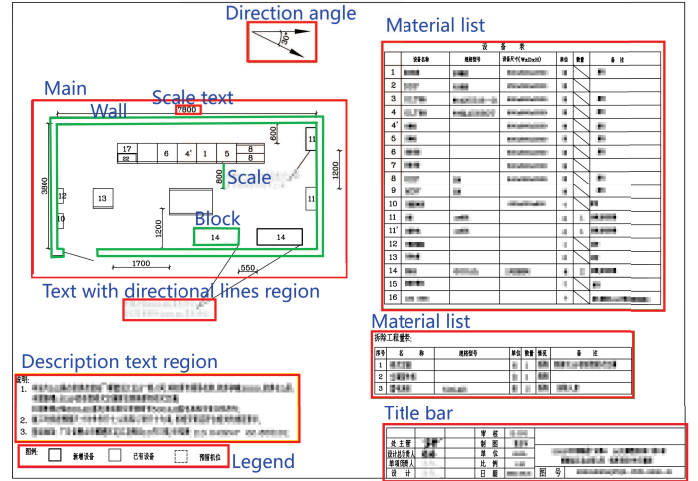
Fig. 1: 12 kinds of semantic objects are defined within CAD images. A red box represents the region symbol, while the green wireframes represent the rectangle symbols.

I. INTRODUCTION

COMPUTER-AIDED DESIGN(CAD) drawings are widely employed for efficient yet precise illustration of products, aiding in the creation, modification, analysis, management, and optimization processes throughout their entire life cycle [1] [2] [3]. Therefore, CAD drawings are extensively utilized in the modern architecture, engineering, and construction (AEC) industries. Currently, many CAD drawings are still stored in the paper format. Semantic understanding of these technical documents in digital libraries becomes necessary due to the following practical requirements:

- *Standardization*: It is necessary to ensure that symbols representing the same objects across different design units or individual engineers adhere to the same drawing standards for the effective communication and reusability.
- *Management*: For an asset owner, indexing, summarizing, and quantifying these assets via CAD drawings would efficiently manage their properties. It is challenging to automatically extract information, such as the number of

J. Pang, Z. Dong and J. Deng are with the Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China (e-mail: junbiao_pang@bjut.edu.cn).

M. Zhu and Y. Zhang are with the Software Development Center, China Information Technology Designing and Consulting Institute Co., Ltd, Beijing 100190, China.

spare frameworks. In fact, a substantial number of drawings in paper or image form result in several drawbacks during actual data storage, retrieval, and editing [4].

- *Re-usability*: Designers or engineers usually reuse CAD drawings from the previous projects for their new designs. In this scenario, an efficient solution is to query semantic symbols from CAD drawings [1] [5].

Symbol spotting and parsing [6] refer to a particular application of computer vision, in which symbols with the domain-specific semantics are localized and recognized to the predefined types [7] [8]. Currently, symbol spotting includes both the domain-specific locations of object instances (*i.e.*, windows, doors, walls and body frames) and the identification of their attributes (*i.e.*, the open direction of a door, the width of a wall and the size of a window). Spotting symbols from the CAD drawings is crucial to the above real-world industrial applications. Traditional symbol spotting [9] [10]for CAD drawings detects instance by classifying the elements (*e.g.*, points, lines, arcs) into different symbols. However, the annotation of CAD drawings requires that annotators should skillfully use a CAD software to group graphical primitives (*e.g.*, arrows, lines, circles) into semantic symbols, as shown in Fig.2(a). The efficient yet low-cost annotation of CAD drawings is a pressing and practical problem that hinders symbol spotting for CAD drawings.

In this paper, we propose to spot symbols from CAD images that are converted from CAD drawings. The motivations for

spotting symbols from CAD images are threefold: 1) the requirements for labelers are lower compared to those for CAD images; 2) CAD drawings in papers or images are handled in a uniform approach; 3) CAD images only require labelers to annotate the keypoints or lines of symbols, as shown in Fig.2(b).

Technically, the symbol spotting for CAD images can be further classified into two sub-tasks as follows:

- *Object-Based Spotting (OBS)*: OBS utilizes off-the-shelf methods to coarsely locate symbols, severing as layout analysis [11] [12] . OBS usually unifies symbols that have the same semantics but different visual representations into uniform symbols. Empirically, the object detection method (*e.g.*, YOLOX [13])achieves excellent performance for the object-level sub-task.

- *Pixel-Based Spotting (PBS)*: PBS precisely decomposes a symbol into a set of keypoints (*e.g.*, the start point and the end one of a line), aiming to locate and redraw this symbol in a pixel-wise approach. PBS paves the way for reusing, changing, or adding symbols for management requirements.A significant challenge of PBS is the pixel-wise keypoint location since most of the images converted from CAD drawings are high resolution, such as, 4000×4000. Therefore, PBS is a non-trivial task for parsing CAD images.

The state-of-the-art(SOTA) keypoint detection approaches use the Gaussian heatmap to represent a point [14] [15] [16] [17] for Human Pose Estimation (HPE). However, HPE is totally different to the keypoint location in CAD images due to the following aspects: 1) keypoints in HPE themselves contain labeling uncertainty, while keypoints in parsing CAD images barely have uncertainty; 2) HPE considers the scale of different objects, while parsing CAD images do not have the scale problem;3) Additionally, the appearances of keypoints in CAD images differ significantly from those in HPE [18] [19] or object counting [20]. Therefore, pixel-wise keypoint location is a novel yet challenging task for parsing CAD images.

Naturally, for the PBS for CAD images, we seek an accurate and stable pixel-wise keypoint location method, based on two motivations. First, although an enormous volume of literature has been devoted to the accuracy of the point detection for HPE [21] [22] [23], there is little attention to the pixel-wise point location method, and practitioners lack guidelines on how to achieve it. Second, we want to combine the advantages of the popular heatmap based method [24] [25]and the regression-based method [26] [27] [28]: the heatmap-based method anchors the accuracy, while the regression-based method enhances the prediction performance. In summary, we desire a pixel-wise keypoint prediction ability for the PBS for CAD images.

In this paper, we handle the pixel-wise keypoint location from the viewpoint of role of Gaussian Kernel Size (GKS) and the compensation of the quantization error. Concretely, a small GKS facilitates more accurate point location than a large GKS by finding the maximum response [29]. However, a small GKS provides insufficient gradient information to train a neural network (as will be discussed in Fig.4).

Conversely, a large GKS expedite the training of a neural network but is susceptible to be miss-guided by the Max-Value Drift (MVD) problem (as will be discussed in Fig.3) during the decoding process [16], rendering the use of large GKS unnecessary. However, a naive approach, switching the GKS during training, has been empirically shown to result in an unstable training process and may even lead to non-convergence.

Motivated by the above analysis, we propose Progressive Gaussian Kernels (PGK) to harmonically use both a large GKS and a small one in an annealing approach without suffering from the complex position decoding process [29] for symbol spotting in CAD images. There are many potential benefits of the combination of the large GKS and small one: accelerating neural network training, and improved keypoint location accuracy. Besides, we introduce an offset in the position encoding stage to reduce the quantization error. Based on keypoint localization, we propose a symbol grouping method for the rectangle-like symbols. We have released an image dataset depicting the layout of equipment rooms from telecommunication industrial CAD drawings. We verify the effectiveness of our method on the proposed dataset.

To the best of our knowledge, this paper is the first to combine the large GKS and the small one for keypoint detection, presenting a comprehensive series of experiments to illustrate the benefits of this novel technique. The proposed method is computationally simple yet exceptionally powerful. Simply by annealing the heatmap size, with no further parameter tuning, we find an accurate keypoint location method that meets or exceeds the current state of the art!

II. RELATED WORK

A. Background of Symbol Spotting

In practice, there are two research lines for symbol spotting: 1) the CAD drawings-based approach uses CAD format(*e.g.*, .dwg)[1][2] [30] [31] and 2) the image-based approach uses images (*e.g.*, .png,. jpeg) converted from the CAD format files [5] [32]. The differences between the two lines are summarized as follows:

- *Pixel-wise location*: As illustrated in Fig2, the CAD drawings supply precisely location of symbols in a structured form, while the image-based approach needs pixel-wisely to locate symbols from images which have the same size as CAD drawings, such as 4200×3000.

- *Efficient annotation*: The CAD drawings require annotators to skillfully use a CAD software to group graphical primitives (*i.e.*, arrows, lines, circles and their segmentations) into semantic symbols (see Fig. 2a). In contrast, the image-based approach only requires labelers annotate the keypoints or the lines of a symbol.

- *Feature representation*: The CAD drawing-based approach utilizes vector graphics extracted from CAD drawings as input, such as, clockwise angles, lengths and types. In contrast, the image-based approach relies solely on unstructured images themselves.

- *Data deficiency*: Nowadays many documents are still stored in paper format without the corresponding CAD

(a) A CAD drawing and its annotation

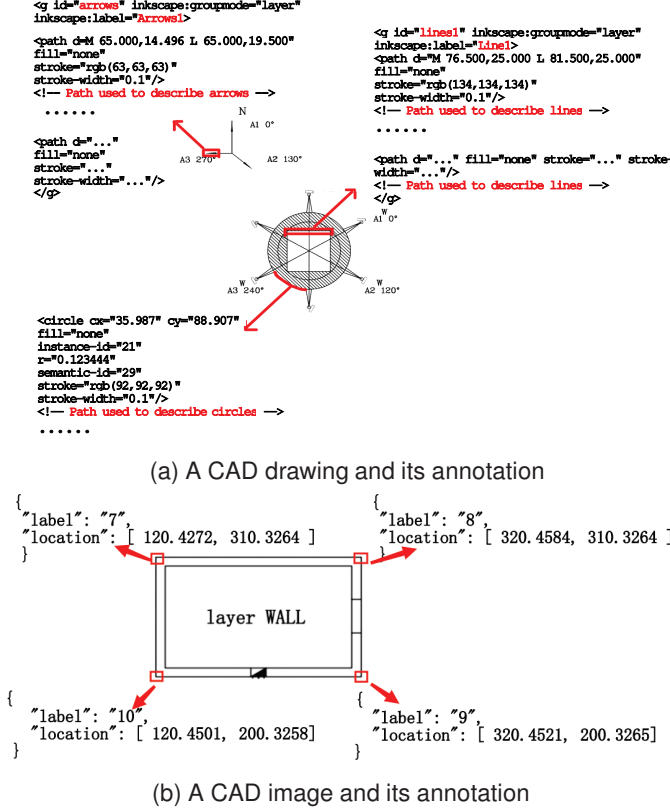(b) A CAD image and its annotation

Fig. 2: Comparison between a CAD drawing and a CAD image.

drawing files. Consequently, the image-based approach naturally handles the deficiency of the CAD drawings.

B. Symbol Spotting

Conventional symbol spotting methods are categorized as the vector-based methods [9] and the image-based methods [33][34][35]. Concretely, the vector-based methods utilize the structural relation among symbol primitives, while the image-based methods leverage the statistical properties of pixels. The challenge is that the handcrafted symbol descriptors barely cope with the graphical notation variability of all kinds [8].

Deep learning has been adapted to the symbol spotting in the vector-based methods for CAD drawings, where the popular detection models Faster-RCNN [36], YOLO [37], Graph Neural Network (GNN) [38], and Transformer [39] have been exploited. However, these methods require the expensive primitives-level annotations, making it difficult to generalize to any vector-based documents. In practice, a symbol maybe drawn from different layers in CAD drawings, requiring professionals with CAD software to label primitives in a time-consuming approach. On the contrary, the image-based method only requires the pixel-wise keypoint annotation, reducing the requirements of labelers.

The image-based methods often locate the bounding boxes of symbols in CAD images via the off-the-shelf object detec-

tors [36] [40] [41]. However, previous image-based methods focus on the types of symbols rather than pixel-based symbol spotting from CAD images.

Recently, a real-world floor plan CAD drawing dataset [5] has been released, contains 35 object classes of interest, including 30 countable symbols and 5 uncountable classes labeled with line-grained annotations in a vector-based document. As a comparison, our dataset contains 15 types of keypoints which are further organized into 12 semantic symbols for CAD images, where the image resolutions range from 1700×1200 to 4200×3000. To our best knowledge, we are the first to release the largest CAD images for symbol spotting via keypoint location.

C. Keypoint Location

Keypoint location methods are generally classified into two categories: the heatmap-based approach and the regression-based one. The heatmap-based approach [42] [43] fully utilized the spatial information around keypoints, thus achieving the higher accuracy than that of the regression-based one. Some studies [18] [23] [17] have proposed deep neural networks for feature extraction. In contrast, the regression-based method directly outputs the coordinates of keypoints for HPE. For example, CenterNet [44] and DirectPose [45] estimate multi-person HPE in a one-stage object detection framework. Recently, Residual Log-likelihood Estimation (RLE) [26] introduced a normalized flow model [46] to capture the underlying distribution of keypoints. Compared with the heatmap-based method, the regression-based approach has made great efforts to model the implicit relationships among keypoints, yet still lacks spatial generalization ability [24]. Experiments empirically show that the heatmap-based methods still obtain a better results than the regression-based ones [25].

It is important to note that the above keypoint location methods were primarily proposed for HPE, in which datasets include MPII [43] and COCO [47]. HPE considers the possible occlusion, object scale and complex backgrounds where the evaluation metric, Object Keypoint Similarity (OKS), is used to handle the uncertainty of keypoints. In contrast, symbol spotting for the CAD images requires the pixel-wise keypoint location. To obtain a pixel-base location for the CAD images, we propose to combine the heatmap-based method and regression-based one, utilizing local offsets to compensate for the quantization errors of keypoints.

III. PROGRESSIVE KEYPOINTS DETECTION

A. Heatmap based Coordinate Encoding-Decoding Revisited

The encoding of a keypoint is the process of converting the a coordinate into a Gaussian distribution. Specifically, given a heatmap $H \in \mathbb{R}^{h \times w}$ with the height h and the width w, we assume the Ground Truth (GT) of a keypoint to be $\boldsymbol{\mu} = [\mu_x, \mu_y]$. The heatmap of a GT keypoint is converted into a Gaussian distribution as follows:

$$ H(x, y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x - \mu_x)^2 + (y - \mu_y)^2}{2\sigma^2}\right) \quad (1) $$

(a) GT heatmap ($\sigma = 3$)   (b) GT heatmap ($\sigma = 1$)

(c) Predicted heatmap ($\sigma = 3$)   (d) Predicted heatmap ($\sigma = 1$)
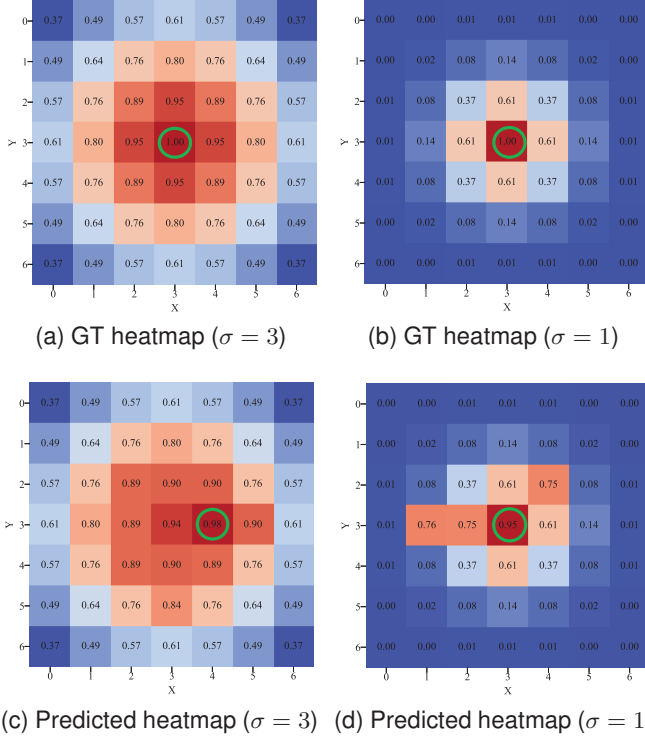
Fig. 3: Illustration of MVD problem by comprising between the GT heatmaps and the predicted heatmaps. (a) and (b) represent the GT heatmaps with the GKS $\sigma = 3$ and $\sigma = 1$, respectively. (c) and (d) show the predicted heatmaps with the corresponding GKS respectively. Color represents the magnitude of the heatmaps response. Circle indicates the position with the maximum value.

where (x,y) is the discrete pixel coordinate in the heatmap, and the parameter σ represents GKS, which models the uncertainly of an annotation for a keypoint.

Due to the inconsistency between the size of the heatmap $H \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times C}$ and the input image $I \in \mathbb{R}^{H \times W \times 3}$, where H and W are the height and width of an image respectively, R ($R > 1$) is the down-sampling rate, and C is the number of keypoint classes, quantization errors inevitably occur during the process of the coordinate encoding in Eq(1). For a GT keypoint $\boldsymbol{\mu} = [\mu_x, \mu_y]$ from the image, its corresponding coordinate in the heatmap is as follows:

$$\widetilde{\boldsymbol{\mu}} = \lfloor \frac{\boldsymbol{\mu}}{R} \rfloor \tag{2}$$

where the operation $\lfloor \cdot \rfloor$ causes a quantization error:

$$O = \frac{\boldsymbol{\mu}}{R} - \widetilde{\boldsymbol{\mu}} \tag{3}$$

in which the range of the quantization error O is bounded by $R \cdot [-1/2, 1/2) \times [-1/2, 1/2)$. As a result, the quantization error O deeply influences the accuracy of the pixel-wise point location.

In this paper, we introduce a local offset in the position encoding process to reduce the quantization errors. Specifically,
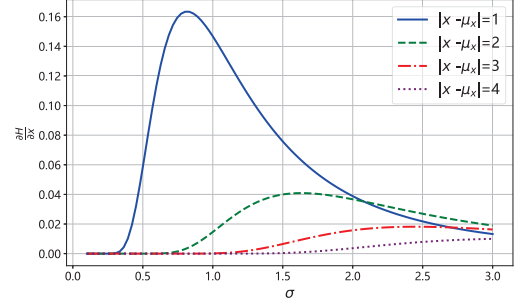

Fig. 4: The relationship between the gradients (6) and the distance $|x - \mu_x|$.

we use a deep model to predict the Gaussian heatmap $\hat{H}$ and the quantization error $\hat{O}$ as follows:

$$L = \frac{1}{N} \sum_{k=1}^{N} \|H_i - \hat{H}_i\|^2 + \frac{\lambda}{N} \sum_{k=1}^{N} \|O_i - \hat{O}_i\|^2 \tag{4}$$

where λ is a hyperparameter, i ($1 \leq i \leq N$) is the index of a sample. The loss function (4) consists of two parts: one computes the mean square error between the GT heatmap H and the predicted heatmap $\hat{H}$; the other computes the mean square error between the real quantization error O and the predicted quantization error $\hat{O}$.

Once the Eq(4) is optimized, the predicted coordinate $(\hat{x}, \hat{y})$ is decoded as follows:

$$(\hat{x}, \hat{y}) = R \cdot \left(\text{Argmax}_{(x,y)}(\hat{H}) + \hat{O} \right) \tag{5}$$

where the function $\text{Argmax}(\cdot)$ locates the spatial coordinate by finding the maximum value.

B. Progressive Change of the GKS

Heatmaps in Eq.(1) have a significant impact on the accuracy of coordinate decoding for each point. As illustrated in Fig.3, the larger GKS is, the smoother the heatmap distribution is. Consequently, during the training stage, a small disturbance would have a higher chance to change the decoded position for the larger GKS than the smaller one [24], although the larger GKS has a lower training loss in Eq(4) than for the smaller one. As illustrated in Fig. 3(c) and (d), for the pixel-wise location tasks, it is suitable to utilize a smaller GKS than a larger one due to the MVD problem.

On the other hand, a larger GKS would bring more supervised signal to train a deep model. Specially, the gradient of H in Eq(1) with respect to each pixel coordinate is as follows:

$$\frac{\partial H}{\partial x} = \frac{(x - \mu_x)}{\sqrt{2\pi}\sigma^3} \exp\left(-\frac{(x - \mu_x)^2 + (y - \mu_y)^2}{2\sigma^2} \right) \tag{6}$$

Fig.4 illustrates the relationship between the a spatial point x and a GT one μ_x. We have drawn two key observations from Fig.4 as follows: 1) For a fixed GKS, the farther a spatial point is away from the GT, the smaller the supervised signal is; 2) For a fixed spatial point, the larger a GKS is, the larger the supervised signal is.
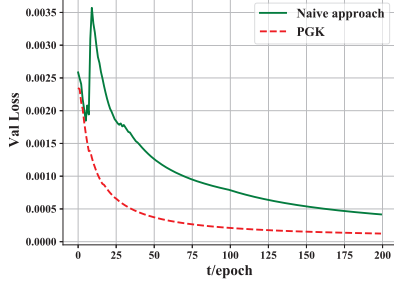
Fig. 5: The loss curve of the validation set for PGK and naive approach. At 100 epochs, the naive method changes the GKS from 3 to 1.

Fig. 6: The curves of GKS in Eq (7) with different values of α, the other hyper-parameters are $\sigma_{max} = 3$, $\sigma_{min} = 1$, and $M = 200$.

Therefore, a naive idea to utilize the above observations is to switch the GKS in the training of a model. That is, in the early stage of training, in order to obtain a large supervision signal for these points away from the GT ones, a larger GKS is used to help the model converge quickly; while, in the later stage of training, a smaller GKS makes the model focus on the local area around the GT points for the detailed information. However, the naive approach faces the "switch problem": the weights from the larger GKS is not a good initialization for the smaller GKS. As illustrated in Fig.5, the naive approach did not accelerate the convergence of training process.

To handle the switch problem, we propose a progressive keypoint location method referred to as point location via Progressive Gaussian Kernels in an annealing approach as follows:

$$\sigma_t = (\sigma_{max} - \sigma_{min})\alpha^{t/M} + \sigma_{min}\big(1 - (\sigma_{max} - \sigma_{min})\alpha\big)^{t/M}$$
(7)

where the parameters σ_{max} and σ_{min} ($\sigma_{max} > \sigma_{min}$) represent the initial GKS and the target one, respectively. σ_t is the annealing GKS at the t-th epoch, α ($0 < \alpha < 1$) is the annealing rate, and M ($1 \leq t \leq M$) is the total number of epochs during training.

The progressive Gaussian kernel is shown in Fig.6. Following the idea of the annealing approach [48], the function $\alpha^{t/M}$ progressively increases with respect to the number of the epochs. Specifically, the change rate of σ_t is controlled by $(\sigma_{max} - \sigma_{min})\alpha^{t/M}$, which reduces the GKS at each epoch.

C. Grouping Keypoints into Rectangle Symbols

The located keypoints have two types of errors: 1) the type of a keypoint is misclassified; 2) the location of a keypoint still has an error. We propose a strategy with the error-correction mechanism to group the keypoints into rectangle symbols for the classification error.

Concretely, we utilize the rectangle shape as a prior to design multiple grouping routes to remove classification errors according the consistence between the grouping point and the candidate one. We formally represent the consistence between points as: $1 == \text{consistence}(p_i, p_j)$. For example, point 7 and point 8, 12, 14 and 15 in Fig. 7 have the consistent group direction to group two points into one third rectangle symbol
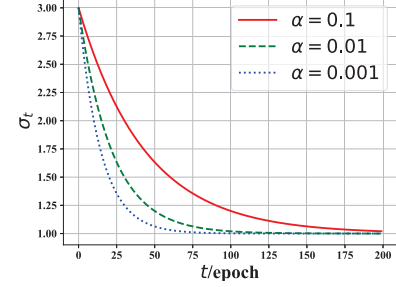
if we group a rectangle from the right of point 7. An example of grouping keypoints into rectangle symbols is illustrated in Alg. 1.

Algorithm 1 Grouping keypoints into the predefined symbols

Input: A start point set $\mathcal{K}$ and the bounding box set of symbols $B_i \in \mathcal{B}$
Output: The rectangle symbol set $V_i \in \mathcal{V}$ where $|\mathcal{V}| = |\mathcal{B}|$
1: **for** $B_j \in \mathcal{B}$ **do**
2:     Select a start point $p_i \in \mathcal{K}$ as the grouping point from B_j
3:     Follow the grouping direction of the point p_i and calculate Euclidean distance d_{ij} between the point p_i and the candidate ones p_j
4:     **if** $l = \min_j d_{ij}$ and 1==consistence(p_i, p_l) **then**
5:         Add the point p_l into the matched list V_i
6:         Use the point p_l as the new start point, $k_i \leftarrow p_l$
7:     **else**
8:         Retrieve anther keypoint from V_i as new start point
9:     **end if**
10:     Goto Step 3
11: **end for**
12: **return** V_i.

IV. EXPERIMENTS

A. Dataset for CAD images

The dataset[1] consists of 300 images for the training and 60 images for the testing, respectively. The image resolutions range from 1700×1200 to 4200×3000. Some examples are illustrated in Fig.10. We use Labelme as the annotation software. Because it supports various annotation methods, including point annotation, rectangle annotation, and polygon annotation. These CAD images about the layout of the equipment rooms are from the telecommunication industrial CAD drawings.

Fig.1 shows 12 types of semantic symbols, including 3 types of rectangle symbols (*i.e.*, Scales, Blocks, and Walls) and 9 types of region symbols (The types of symbols are illustrated in Table I). The size of these symbols in CAD images are

---
[1]https://github.com/pangjunbiao/CAD-dataset/

significantly different from each other. The region symbols are efficiently located by object detection methods for the OBS task. Specifically, the sizes of the rectangle symbols are smaller than those of the region symbols, requiring the pixel-wise location.

To locate the rectangle symbols, the first step involves coarsely localizing region symbols, followed by the point-wise keypoint location. Fig. 7 shows the 15 types of keypoints that compose the rectangle symbols. Concretely, for scales, 6 types of keypoints are defined based on the direction of the scale's endpoints. For blocks and walls, 9 types of keypoints are defined based on the shape of the corners.

In summary, parsing rectangle symbols poses a significant challenge to the pixel-wise point location for the symbol spotting task. Because the symbol grouping algorithm in Alg. 1 requires both the type and the location of keypoints should be correctly classified and located.

B. Evaluation Metrics

In our experiment, we used two kinds of evaluation metrics to evaluate the accuracy of the keypoint location and the symbols spotting as follows:

F1 score: The F1 score offers a balanced evaluation between Precision and Recall as follows:

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (8)$$

where Precision and Recall are respectively defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (9)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (10)$$

in which TP is the number of the successfully detected keypoints or region symbols, FP is the number of incorrectly detected keypoints or region symbols, and FN is the number of keypoints or region symbols that were not detected when they should have been. Specifically, whether a keypoint or region symbol has been correctly detected is further defined as follows:

- For region symbols, Intersection over Union (IoU) evaluates the ratio of the intersection and union between the predicted region symbols B and true symbols G:

$$\text{IOU} = \frac{\text{Intersection}(B, G)}{\text{Union}(B, G)} \quad (11)$$

When IOU is larger than a predefined threshold τ_o(i.e., IOU $> \tau_o$), we consider that a region symbol is correctly detected. In this paper, τ_o is set as 0.5.

- For keypoints, Euclidean distance ℓ_i between the i-th predicted keypoint and the corresponding GT point is used:

$$\ell_i = \sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2} \quad (12)$$

When ℓ_i is smaller than a predefined threshold τ_p (i.e., $\ell_i < \tau_p$), we consider that a keypoint is correctly detected. In this paper, τ_p is set as 2.

Fig. 7: 15 kinds of keypoints are defined to group the rectangle symbols.

TABLE I: The results of the object-based symbol spotting by object detection method

Region Symbol	Precision	Recall	F1
Main	1.00	1.00	1.00
Main title	1.00	0.92	0.96
Title bar	1.00	1.00	1.00
Material list	1.00	1.00	1.00
Legend	1.00	1.0	1.00
Description text	1.00	1.00	1.00
Direction angle	1.00	0.90	0.95
Text with directional lines	1.00	0.98	0.99
Scale text	0.97	0.97	0.97

Averaged Pixel Error for Keypoints (APEK): APEK evaluates the accuracy of the pixel-wise keypoint location as follows:

$$\text{APEK} = \frac{1}{N} \sum_{i=1}^{N} \ell_i \quad (13)$$

where ℓ_i is the Euclidean distance in Eq(12), N is the number of all the detected keypoints (including the wrongly detected ones).

C. Implement Details

We use the Stacked Hourglass network [18] as the backbone network. The initial learning rate for the ADAM optimizer was set to 10^{-4}, and the hyper-parameter λ was set to 0.1. The model was optimized for 200 epochs. In the training stage, we first locate the region symbols, the detected region symbols are cropped into a set of the 256×256 pixel patches by a sliding window approach with the step size is equal to 0. In the inference stage, we divide a complete CAD image into pixel patches of size 256×256, and then feed each patch into the model for keypoint localization, where Non-Maximum Suppression (NMS) is applied to obtain the keypoints. We detected a keypoint by judging whether the max value of the heatmap $\hat{H}$ in Eq(5) is larger than the threshold $\varsigma = 0.6$. Finally, we stitch the results from each patches into the coordinates of the CAD images.

D. Experimental Results

The rectangle symbols usually are contained within these region symbols. Rather than detecting rectangle symbols across the whole CAD images, it is reasonable to locate the rectangle symbols within the region symbols.

1) Performances of OBS: We located the region symbol by YOLOX [13]. Table I demonstrates that YOLOX achieves an excellent performance for the object-based symbol spotting. Serving as the layout analysis for CAD images, the accuracy of the OBS task is critical to improve the inference speed for the PBS task. Because rather than locating the small rectangle symbols in a high-resolution image (see Fig.10), we need to further finely locate the rectangle symbols within these detected region symbols.

In this work, the comparisons with these advanced methods (such as [49] [50]) are not conducted. Because the OBS task is not our main contribution in this work; besides, we believe that any advanced object detection methods could be used to the OBS task.

2) Performances of PBS: **Baselines** There are many heatmap-based methods and the regression-based methods for point location for HPE. To show the superiority of the pixel-wise location ability of the proposed method for parsing CAD drawings, we choose two State-Of-The-Art (SOTA) methods (*i.e.*, soft-argmax [28]) from the former and one (*i.e.*, RLE [26]) from the latter as our baselines as follows:

- **Soft-argmax [51]** assumes that a model learns a discrete probability map π_{y_i} which indicates the probability of the predicted target point at y_i. The quantization error is avoided by an elegant approximation, *i.e.*, $\hat{y} = $ soft-argmax$(\pi) = \sum_i \pi_{y_i} y_i$, where π is a normalized distribution. Soft-argmax avoids the quantization errors by the expectation of the probability map π.
- **RLE [26]:** RLE utilizes the normalizing flows [52] to capture the underlying output distribution and makes the regression-based methods match the accuracy of SOTA heatmap-based methods. RLE, as a regression-based method, naturally avoids the quantization errors.

To achieve the pixel-based symbol spotting, we firstly locate keypoints in Fig.7 via PGK in subsection III-B, and then apply Alg. 1 to group these detected keypoints into rectangle symbols. In the following experiments, we report the performances of keypoint location and then the accuracy of the Alg. 1.

Performances of Keypoint Location. Table II shows the consistent performance superiority over the heatmap-based counterpart, *i.e.*, soft-argmax [28] and the regression-based method, *i.e.*, RLE [26]. Note that the evaluation and network training are conducted under the same input size. Table II also shows the offset component in Eq(4) uniformly increase the performances keypoint location in terms of both F1 and AEPK measurements. For example, our method outperforms heatmap-based counterpart by +0.03 F1 and +0.41 AEPK, respectively.

Performances of Symbol Grouping Algorithm.

Table II also shows consistent performance gains over the heatmap-based counterparts, *i.e.*, soft-argmax [28] and the regression-based method, *i.e.*, RLE [26]. Results presented in Table II demonstrate that:1) the accuracy of point location has a great influence on the final grouped symbols; 2) the symbol grouping alg. 1 is able to partially rectify some misclassified points.

TABLE II: The results of point location and the symbol grouping algorithm

Methods	Point		Scale	Block and Wall
	F1 ↑	AEPK ↓	F1 ↑	F1 ↑
soft-argmax	0.76	4.92	0.77	0.76
RLE	0.78	4.85	0.79	0.78
Our method	**0.79**	**4.51**	**0.79**	**0.79**

In summary, according to the results presented in Table II, we can further draw the following conclusions: 1) Soft-argmax still produces an unconstrained probability map for CAD images, which also suffers the bias problem [53]. 2) Our proposed method combines the characteristics of the heatmap-based method and the regression-based one, leading to a simpler and more efficient scheme.

3) Ablation Studies for KeyPoint Location: In order to demonstrate the effectiveness of the local offset in Eq(4) and the PGK in Eq(7), we conducted ablation studies respectively.

Effectiveness of Local Offset: Table III shows that the local offset boosts performances of different types of keypoints in terms of both F1 and AEPK metrics. Specifically, the performances of point 3 and point 4 in Fig. 7 from the scale symbol are significantly improved from 0.72 to 0.74 and 0.83 to 0.85 respectively in terms of F1 score. Interestingly, The AEPK scores of the point 3 and point 4 are also reduced from 4.99 to 4.56 and 4.76 to 4.19, respectively. The results mean that the offset term not only improves the classification accuracy of points but also reduces the spatial location errors.
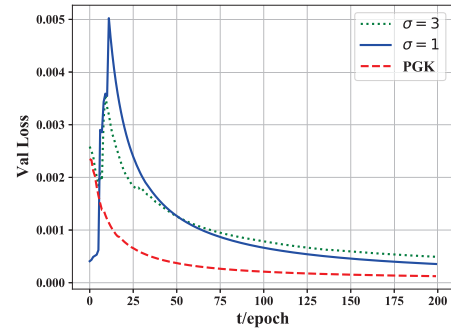
Fig. 8: Validation set loss curve with different Gaussian kernel.

Effectiveness of PGK: We discuss the impact of the proposed PGK by comparing the following baselines:

1. GKS with $\sigma = 3$: $\sigma = 3$ is a common setting to handle the uncertainty in the label errors for HPE. However, GKS with $\sigma = 3$ would incur location errors due to the MVD problem in Fig.3.
2. GKS with $\sigma = 1$: $\sigma = 1$ is barely used for HPE. We verify that a smaller GKS would obtain a lower location error than that of GKS with a larger kernel for symbol spotting from CAD images.

Fig.8 shows that the proposed PGK enables a model to not only have a "smooth" training process but also a faster convergence speed than the counterparts, *i.e.*, GKS with $\sigma = 3$ and GKS with $\sigma = 1$. The comparison shows that the PGK

TABLE III: The effectiveness of the local offset for keypoint location

	local offset	Scale						Block and Wall									Average
		point1	point2	point3	point4	point5	point6	point7	point8	point9	point10	point11	point12	point13	point14	point15	
F1↑	✗	0.70	0.70	0.72	0.83	0.78	0.84	0.79	0.81	0.75	0.80	0.83	0.78	0.90	0.77	0.70	0.77
	✓	0.72	0.71	0.74	0.85	0.79	0.85	0.80	0.81	0.76	0.82	0.86	0.81	0.92	0.80	0.72	0.79
AEPK↓	✗	5.00	4.88	4.99	4.76	4.9	4.51	4.59	4.94	4.74	4.89	4.73	5	4.84	4.91	4.92	4.88
	✓	4.78	4.58	4.56	4.19	4.34	4.23	4.5	4.47	4.34	4.72	4.44	4.73	4.51	4.69	4.77	4.51

TABLE IV: The effectiveness of PGK for keypoint location

|	PGK	Scale						Block and Wall									Average
		point1	point2	point3	point4	point5	point6	point7	point8	point9	point10	point11	point12	point13	poin14	point15	
F1 ↑	✗	0.72	0.71	0.74	0.85	0.79	0.85	0.80	0.81	0.76	0.82	0.86	0.81	0.92	0.80	0.72	0.79
	✓	0.75	0.79	0.81	0.9	0.87	0.89	0.83	0.87	0.77	0.83	0.90	0.86	0.95	0.83	0.78	0.84
AEPK↓	✗	4.78	4.58	4.56	4.19	4.34	4.23	4.5	4.47	4.34	4.72	4.44	4.73	4.51	4.69	4.77	4.51
	✓	4.62	4.56	4.41	3.95	4.33	4.18	4.33	4.29	4.08	4.51	4.43	4.43	4.26	4.48	4.52	4.36
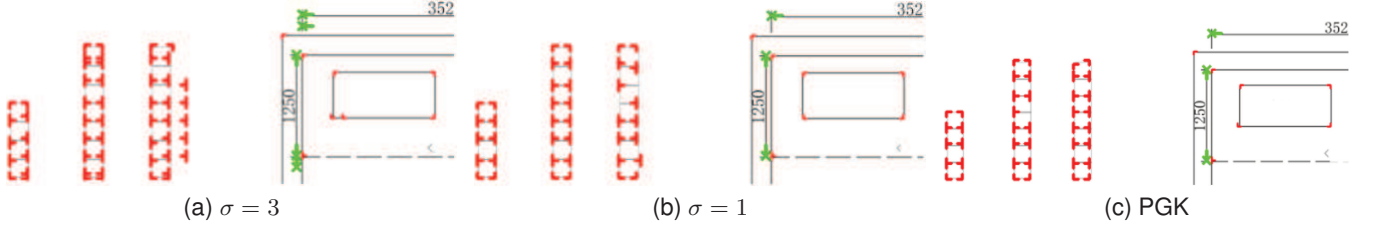
Fig. 9: Visualization results of the detected keypoint for the different σ.
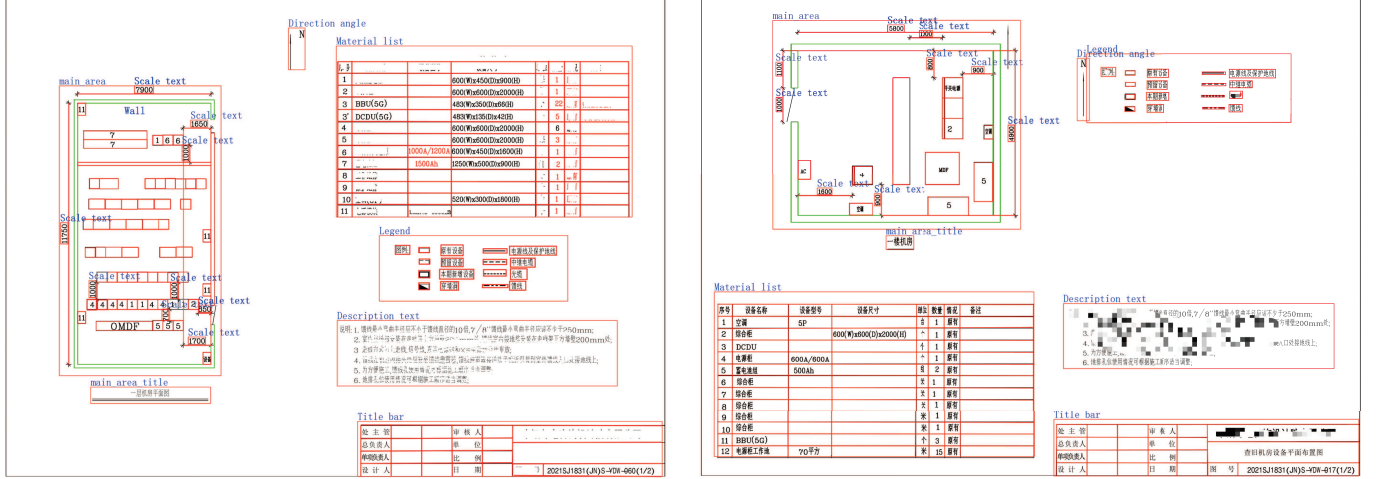
Fig. 10: sample results of the symbol spotting for CAD images.

enjoys the advantages of both the large GKS and the small one. Moreover, the smallest validation loss indicates that the proposed PGK has a best generalization ability compared to these counterpart methods. Fig.9 vividly compares the detection results among different kernel sizes. Fig.9(a) shows that the GKS with $\sigma = 3$ tends to predict duplicated keypoints due to the MVD problem and the NMS operation. Comparing between Fig.9(b) and Fig.9(c), the proposed PGK yields more accurate results than that of GKS with $\sigma = 1$.

Results presented in Table IV further demonstrate that the proposed PGK shows consistent performance superiority over GKS with $\sigma = 1$. For example, F1 score is increased from 0.79 to 0.84, and the averaged value of AEPK is decreased from 4.51 to 4.36.

E. Visualization results

Fig.10 illustrates that the visual result of the pixel-wise symbol spotting via both PGK and the symbol grouping algorithm. By comparing the spotting position with the original points, our system successfully parses the complex symbols for the telecommunication industrial drawings.

CONCLUSION

In this paper, we propose the PGK approach to achieve both fast training speed and the good results in keypoint location. It adjusts the GKS in a progressively annealing approach to avoid the inefficiency of the small GKS yet enjoys the fast training speed of a large GKS. To enhance the location precision, we introduce a local offset into the heatmap based method.

This approach enhances the accuracy of point location, setting a new benchmark in pixel-wise keypoint detection for CAD image parsing. We further propose a symbol grouping method with the error-correction ability, which is a significant advancement in the field of symbol spotting in CAD images.

Our method is computationally simple, coupled with its outstanding performance, underscoring its potential as a transformative tool for professionals engaged in CAD image analysis.By setting a precedent in the use of PGK and localized position encoding, this work paves the way for future investigations into efficient and accurate symbol spotting methodologies for parsing CAD images.

In the future work, we aim to enhance our work as follows: 1) how to group the non-rectangle symbols, such as, the symbols grouped by arcs as illustrated in Fig. 2; 2) verify advantageous backbones, such as, HRNet [17] and Token-Pose [54], in terms of the training speed, inference speed and accuracy.

REFERENCES

[1] Z. Zheng, J. Li, L. Zhu, H. Li, F. Petzold, and P. Tan, "Gat-cadnet: Graph attention network for panoptic symbol spotting in cad drawings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 747–11 756.

[2] Z. Fan, T. Chen, P. Wang, and Z. Wang, "Cadtransformer: Panoptic symbol spotting transformer for cad drawings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 986–10 996.

[3] W. Liu, T. Yang, Y. Wang, Q. Yu, and L. Zhang, "Symbol as points: Panoptic symbol spotting via point-based representation," *arXiv preprint arXiv:2401.10556*, 2024.

[4] C. Li, H. Pan, A. Bousseau, and N. J. Mitra, "Free2cad: Parsing freehand drawings into cad commands," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–16, 2022.

[5] Z. Fan, L. Zhu, H. Li, X. Chen, S. Zhu, and P. Tan, "Floorplancad: A large-scale cad drawing dataset for panoptic symbol spotting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 128–10 137.

[6] A. Rezvanifar, "Analyzing symbols in architectural floor plans via traditional computer vision and deep learning approaches," Ph.D. dissertation, 2021.

[7] H. Bhanbhro, Y. Kwang Hooi, W. Kusakunniran, and Z. H. Amur, "A symbol recognition system for single-line diagrams developed using a deep-learning approach," *Applied Sciences*, vol. 13, no. 15, p. 8816, 2023.

[8] A. Rezvanifar, M. Cote, and A. B. Albu, "Symbol spotting on digital architectural floor plans using a deep learning-based framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 568–569.

[9] A. Rezvanifar, M. Cote, and A. Branzan Albu, "Symbol spotting for architectural drawings: state-of-the-art and new industry-driven developments," *IPSJ Transactions on Computer Vision and Applications*, vol. 11, pp. 1–22, 2019.

[10] S. Sarkar, P. Pandey, and S. Kar, "Automatic detection and classification of symbols in engineering drawings," *arXiv preprint arXiv:2204.13277*, 2022.

[11] X. Wu, Y. Zheng, T. Ma, H. Ye, and L. He, "Document image layout analysis via explicit edge embedding network," *Information Sciences*, vol. 577, pp. 436–448, 2021.

[12] G. M. Binmakhashen and S. A. Mahmoud, "Document layout analysis: a comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–36, 2019.

[13] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[14] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1653–1660.

[15] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 466–481.

[16] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5693–5703.

[17] C. Yu, B. Xiao, C. Gao, L. Yuan, L. Zhang, N. Sang, and J. Wang, "Lite-hrnet: A lightweight high-resolution network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 10 440–10 450.

[18] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*. Springer, 2016, pp. 483–499.

[19] H. Qu, L. Xu, Y. Cai, L. G. Foo, and J. Liu, "Heatmap distribution matching for human pose estimation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 327–24 339, 2022.

[20] W. Liu, M. Salzmann, and P. Fua, "Counting people by estimating people flows," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 11, pp. 8151–8166, 2021.

[21] Y. Wang, M. Li, H. Cai, W.-M. Chen, and S. Han, "Lite pose: Efficient architecture design for 2d human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 126–13 136.

[22] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 124–12 134.

[23] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[24] A. Nibali, Z. He, S. Morgan, and L. Prendergast, "Numerical coordinate regression with convolutional neural networks," *arXiv preprint arXiv:1801.07372*, 2018.

[25] Y. Li, S. Yang, P. Liu, S. Zhang, Y. Wang, Z. Wang, W. Yang, and S.-T. Xia, "Simcc: A simple coordinate classification perspective for human pose estimation," in *European Conference on Computer Vision*. Springer, 2022, pp. 89–106.

[26] J. Li, S. Bian, A. Zeng, C. Wang, B. Pang, W. Liu, and C. Lu, "Human pose regression with residual log-likelihood estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 11 025–11 034.

[27] K. Gu, L. Yang, and A. Yao, "Removing the bias of integral pose regression," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 067–11 076.

[28] X. Sun, B. Xiao, F. Wei, S. Liang, and Y. Wei, "Integral human pose regression," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 529–545.

[29] K. Gu, L. Yang, and A. Yao, "Dive deeper into integral pose regression," in *International Conference on Learning Representations*, 2021.

[30] L. W. Jianwu, L. S. Yew, L. K. On, T. C. Keong, R. T. Yuan Sheng, S. B. Sani, and T. H. Juan Agnes, "Artificial intelligence-enabled evaluating for computer-aided drawings (amcad)," *International Journal of Mechanical Engineering Education*, vol. 52, no. 1, pp. 3–31, 2024.

[31] Y. You, G. Liu, and L. Liu, "Art design method of interior space layout based on cad drawing," 2022.

[32] Q. Lu, L. Chen, S. Li, and M. Pitt, "Semi-automatic geometric digital twinning for existing buildings based on images and cad drawings," *Automation in Construction*, vol. 115, p. 103183, 2020.

[33] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 224–236.

[34] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017.

[35] Y. Ko, Y. Lee, S. Azam, F. Munir, M. Jeon, and W. Pedrycz, "Key points estimation and point instance segmentation approach for lane detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8949–8958, 2021.

[36] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[37] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia computer science*, vol. 199, pp. 1066–1073, 2022.

[38] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[41] R. G. Schneider and T. Tuytelaars, "Example-based sketch segmentation and labeling using crfs," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 5, pp. 1–9, 2016.

[42] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 648–656.

[43] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, "2d human pose estimation: New benchmark and state of the art analysis," in *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, 2014, pp. 3686–3693.

[44] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578.

[45] Z. Tian, H. Chen, and C. Shen, "Directpose: Direct end-to-end multi-person pose estimation," *arXiv preprint arXiv:1911.07451*, 2019.

[46] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," *arXiv preprint arXiv:1605.08803*, 2016.

[47] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[48] M. S. Barkhordari and M. Tehranizadeh, "Response estimation of reinforced concrete shear walls using artificial neural network and simulated annealing algorithm," in *Structures*, vol. 34. Elsevier, 2021, pp. 1155–1168.

[49] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7464–7475.

[50] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-time flying object detection with yolov8," *arXiv preprint arXiv:2305.09972*, 2023.

[51] J. Li, T. Chen, R. Shi, Y. Lou, Y.-L. Li, and C. Lu, "Localization with sampling-argmax," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 236–27 248, 2021.

[52] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538.

[53] K. Gu, L. Yang, M. B. Mi, and A. Yao, "Bias-compensated integral regression for human pose estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[54] Y. Li, S. Zhang, Z. Wang, S. Yang, W. Yang, S.-T. Xia, and E. Zhou, "Tokenpose: Learning keypoint tokens for human pose estimation," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 11 313–11 322.