Received: 23 August 2023 | Accepted: 8 May 2024

DOI: 10.1111/mice.13238

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING WILEY

RESEARCH ARTICLE

Deep learning framework with Local Sparse Transformer for construction worker detection in 3D with LiDAR

Mingyu Zhang | **Lei Wang** | **Shuai Han** | **Shuyuan Wang** | **Heng Li**

Department of Building and Real Estate, Hong Kong Polytechnic University, Hong Kong, China

Correspondence
Shuai Han, Department of Building and Real Estate, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong 999077, China.
Email: shuaihan@polyu.edu.hk

Funding information
Start-up Fund for RAPs under the Strategic Hiring Scheme of the Hong Kong Polytechnic University, Grant/Award Number: P0042478; Internal Research Fund of PolyU (UGC), Grant/Award Number: P0047899; Guangdong-Hong Kong Technology Cooperation Funding Scheme, Grant/Award Number: GHP/166/21SZ

Abstract

Autonomous equipment is playing an increasingly important role in construction tasks. It is essential to equip autonomous equipment with powerful 3D detection capability to avoid accidents and inefficiency. However, there is limited research within the construction field that has extended detection to 3D. To this end, this study develops a light detection and ranging (LiDAR)-based deep-learning model for the 3D detection of workers on construction sites. The proposed model adopts a voxel-based anchor-free 3D object detection paradigm. To enhance the feature extraction capability for tough detection tasks, a novel Transformer-based block is proposed, where the multi-head self-attention is applied in local grid regions. The detection model integrates the Transformer blocks with 3D sparse convolution to extract wide and local features while pruning redundant features in modified downsampling layers. To train and test the proposed model, a LiDAR point cloud dataset was created, which includes workers in construction sites with 3D box annotations. The experiment results indicate that the proposed model outperforms the baseline models with higher mean average precision and smaller regression errors. The method in the study is promising to provide worker detection with rich and accurate 3D information required by construction automation.

1 | INTRODUCTION

Autonomous equipment, such as autonomous excavators (Eraliev et al., 2022) and construction robots (Allinson, 2022; Malewar, 2019), has played an increasingly important role in construction, greatly improving safety and productivity through the replacement of workers in dangerous or repetitive tasks (Business Research, 2023). Due to the growing demand for automation, intelligent technologies for sensing the surrounding environment have become more and more crucial to ensure efficient and safe operations. Autonomous equipment that cannot recognize and localize construction workers may become a source of collision safety accidents (W. Li et al., 2023) and are difficult to be involved in a human–robot collaboration system (Dogan et al., 2023). Therefore, it is necessary to provide them with sufficient capabilities to detect workers in construction sites.

Plenty of efforts have been made by researchers to recognize workers using RGB cameras. For example, several studies (M. Park et al., 2023; Son, Choi, et al., 2019) have proposed deep learning-based frameworks for worker

detection using cameras, and some have applied them for worker safety (Son & Kim, 2021; Son, Seong, et al., 2019). However, they cannot provide enough spatial information and thus cannot meet the spatial perception requirement of construction automation. To overcome the limitations of cameras, some studies tried to estimate spatial information from 2D images. However, existing depth estimation methods still face challenges such as low accuracy and resolution (Ming et al., 2021), as well as high computational complexity that hinders real-time operation (Laga et al., 2020). Although depth cameras can acquire spatial information, their working distance is often limited (typically not exceeding 5 m), and they are susceptible to strong light interference (Al-Naji et al., 2017), which restricts their application on construction sites.

In light of the successful application of light detection and ranging (LiDAR) in the field of autonomous driving, this device is regarded to be promising for spatial perception and scene understanding. LiDAR has a long period of application in construction, such as structural monitoring (H. S. Park et al., 2007). It enables real-time scanning of the 3D surroundings and generates point clouds. With direct access to spatial information, 3D object detection can be achieved, which involves predicting the locations, dimensions, and orientations of objects in 3D space (Mao et al., 2022). Additionally, LiDAR works well in low light or dust in air conditions, which gives it a great advantage over cameras. Therefore, employing LiDAR to facilitate detecting workers in 3D is considered both promising and necessary.

Few works inferred 3D detection boxes from images on construction sites (Shen et al., 2021; X. Yan et al., 2020), and no research directly detected workers in 3D using LiDAR. Although many 3D object detection approaches have been developed in fields such as autonomous driving (Yang et al., 2018; Zhou & Tuzel, 2018) and indoor detection (Qi et al., 2019), the primary challenge is that they are difficult to be directly applied to worker detection on construction sites. First, different from street scenes where the ground is usually flat and clean, the surface of construction sites is rough and uneven, which may contain excavated soil or under-construction floors. Second, unlike pedestrians on streets, construction workers exhibit various postures (Roberts et al., 2020), which can easily blend with the cluttered background, posing a challenge to detect. Third, they were scarcely designed for human detection, mostly concentrating on vehicle detection, leading to a relatively low accuracy of human detection (Y. Guo et al., 2021).

The purpose of this study is to develop a 3D object detection method that can efficiently and accurately detect workers in construction sites using LiDAR. We propose a deep-learning model that follows the modern 3D object detection paradigm, which adopts an anchor-free voxel-based detection framework. A Local Sparse Transformer (LST) block that applies multi-head self-attention on local voxel features in spatial grids is proposed, which can increase the model's capability of feature extraction. To reduce the computation cost, a downsampling sparse convolution operation is proposed, which can prune redundant features and maintain data sparsity. The backbone network of the model incorporates Transformer and convolution layers, efficiently processing sparse data. The contribution of the study can be summarized as follows:

1. LiDAR point cloud was leveraged to perform 3D detection for construction workers, which can provide accurate and rich spatial information about workers.
2. A deep learning model was developed for 3D worker detection. In the model, the LST block, 3D sparse convolution with XY dilation, and the backbone network combining LST with convolution were proposed to improve detection.
3. An annotated point cloud dataset with workers in various poses from construction sites was created. The method achieves improved performance in testing on the dataset.

2 | RELATED WORK

2.1 | Worker detection methods

Since the surge of deep learning (Martins et al., 2020; Pereira et al., 2020), numerous studies have investigated camera-based detection utilizing convolutional neural networks (CNNs), such as the early attempt (Fang et al., 2018; Son, Seong, et al., 2019) and recent refinement (M. Park et al., 2023; Son & Kim, 2021). However, they can only generate 2D bounding boxes on the image plane without any 3D clue. Many studies have attempted to infer spatial information from camera-based detection results, often relying on stereo vision, projection transformations, or depth estimation techniques. For instance, the stereo vision technique could be based on the 2D detection results from a single camera (Lee & Park, 2019). However, it may struggle to provide precise 3D information when objects are far from the camera. Projection transformation methods project each point from a 2D image to a 3D space (Son, Seong, et al., 2019), but they have inherent drawbacks, including the challenges of camera calibration and the limitations imposed by the assumption of planar surfaces. In a quite related study (X. Yan et al., 2020), workers' depth and distance information are estimated based on their size and location in 2D images, but it falls short of true 3D object detection since the reconstruction is based on 2D

detection results. In summary, existing worker detection methods to restore 3D information suffer from errors of transformation from 2D to 3D and are subject to camera inaccuracies.

2.2 | 3D object detection with LiDAR

Since LiDAR has been widely applied for autonomous driving, numerous 3D object detection frameworks were proposed based on the street point cloud dataset. The development of 3D object detection frameworks draws inspiration from the well-established 2D object detection techniques (W. Liu et al., 2016; Ren et al., 2017). For instance, they comprised a backbone and detection head network.

The backbone serves as the point cloud feature extractor. Based on how the point cloud is handled, the backbones can be categorized into point-based methods and discretization-based methods. Point-based methods, such as PointNet (Charles et al., 2017) and PointNet++ (Qi et al., 2017), directly process raw points. However, they require appropriate sampling methods, posing a challenge in balancing computational efficiency and performance. Discretization-based methods convert point clouds into discrete cubic voxels, pillars, or bird's-eye-view (BEV) images, where CNNs are mainly employed to extract features. The voxel-based methods (Zhou & Tuzel, 2018) mainly employ 3D CNNs as the backbone. SECOND (Y. Yan et al., 2018) employs 3D sparse convolution layers that operate solely on effective voxels, which substantially reduces the computational burden of the 3D backbone. It significantly advanced voxel-based methods and has emerged as the backbone network for many subsequent 3D object detection methods. In contrast, pillar-based (Lang et al., 2019) and BEV-based (Beltrán et al., 2018; Yang et al., 2018) methods process coarse features, and thus yield worse performance than voxels.

For detection heads, anchor-based methods were popular in the early years (Lang et al., 2019). However, detection with multiple classes in large scenes requires many anchors, which can make assigning ground truth labels to anchors a tedious task. An increasing number of models have been embracing simple and efficient anchor-free methods. Anchor-free manner combined with voxel-based methods (Q. Chen et al., 2020; Yang et al., 2018; Yin et al., 2021) has been investigated recently, where the detection boxes were generated directly from the voxel features. Among these models, CenterPoint (Yin et al., 2021) is perhaps the most efficient. It directly utilizes a CNN to generate a heat map that represents the object centers and regresses object properties from features at the centers.

2.3 | Transformer in computer vision

The remarkable success of the Transformer (Vaswani et al., 2017) in natural language processing has inspired numerous investigations into the potential of multi-head self-attention in computer vision. The pioneering work, ViT (Dosovitskiy et al., 2021), did not solve the unacceptable computational cost of the original Transformer due to quadratic computational complexity to the input size. To address this issue, Swin Transformer (Z. Liu, Lin, et al., 2021) applies the Transformer in non-overlapping local windows of an image and hierarchically processes the image, achieving linear computational complexity to the input size. Furthermore, transformers are well-suited for capturing long-range dependencies, while CNNs excel at extracting local features. As a result, some studies have designed hybrid models that combine Transformers and CNNs (J. Guo et al., 2022; J. Li et al., 2022), achieving better performance than using either Transformers or CNNs alone.

Several studies have investigated the applicability of Transformers in point cloud learning. Compared to 2D computer vision, Transformers in 3D cases are much more computationally intensive, given the larger data size of 3D data. Some studies have focused on 3D object detection in indoor scenes (Z.Liu, Zhang, et al., 2021; Misra et al., 2021). They are only suitable for indoor scenes, which typically have denser points and smaller scales than outdoor scenes. Recently, some research has proposed backbones based on the Transformer encoder for large-scale outdoor scenes. VoxSeT (He et al., 2022) reduces the computation of the Transformer by cross-attention for efficient object detection. VoTr (Mao et al., 2021) proposed a voxel self-attention method, where each voxel acts as a query and attends with its neighboring voxels. However, it also suffers from large computational costs and requires a significant amount of memory for training. In this scenario, relying solely on the Transformer to extract features may not be the most efficient solution.

2.4 | Research gaps

The following research gaps can be identified based on the above literature review:

1. The prevailing worker detection methods applied on construction sites are mainly based on cameras and pose challenges in conducting accurate 3D scene understanding due to the limitations of the hardware.
2. The existing LiDAR-based object detection methods mainly focus on simple scenes such as streets and

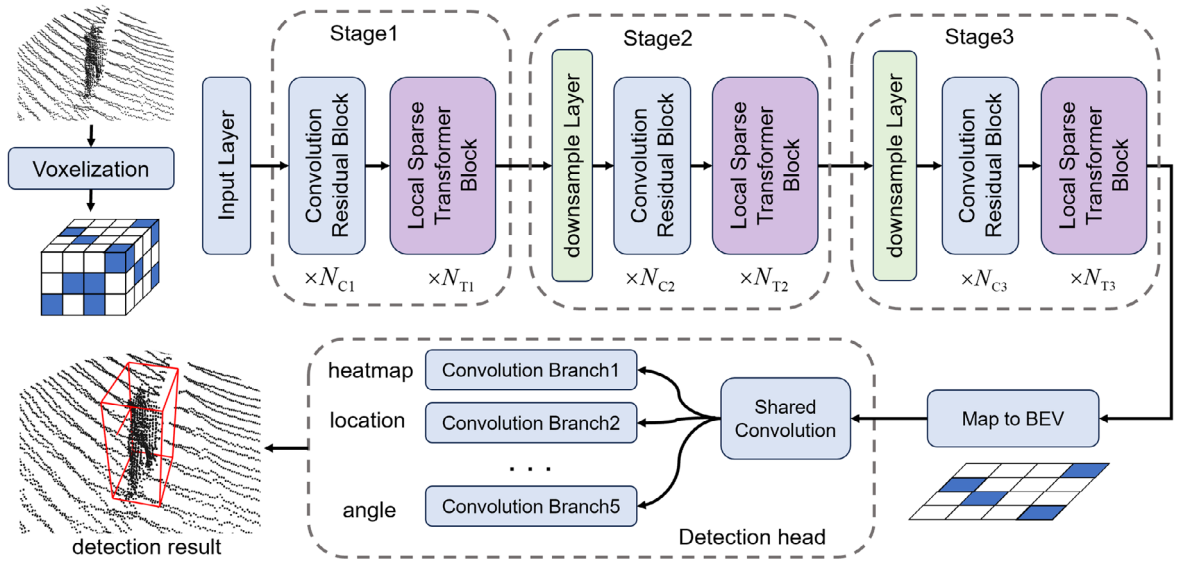


FIGURE 1 The model architecture. BEV, bird's-eye-view.

indoors, and their performances on construction sites have not been validated.

3. The Transformer model possesses a powerful feature extraction capability. However, there is a lack of research investigating the efficient implementation of Transformer and combination with CNN on large-scale complex point clouds.

3 | METHODOLOGY

In this section, we present the deep learning model designed for detecting construction workers in 3D. First, we introduce the overall architecture of the deep network. Next, we present the proposed LST block and innovation in convolution layers. Finally, the evaluation metrics are introduced.

3.1 | Overall architecture

The model's architecture is illustrated in Figure 1. The input point cloud goes through voxelization, a 3D backbone, a map-to-BEV module, and a detection head in sequence. The input point cloud contains n raw points with four-dimensional features (x, y, z, intensity). It is first cropped into a cubic space and then voxelized into N non-empty voxels with an initial voxel size (s_x, s_y, s_z) and spatial size $L_x \times L_y \times L_z$. The essence of voxelization is to convert the raw points into a regular cubic grid in the 3D space, determining which points are contained within each cube. The number of voxels N is not fixed and is limited by a maximum threshold. The number of points in a voxel is fixed to n_v, with zeros padding the voxel if the number of points is less than n_v. Each voxel's feature is encoded by calculating the mean value of all the points in it. The resultant features to be fed to the backbone network is $f_v \in \mathbb{R}^{N\times 4}$.

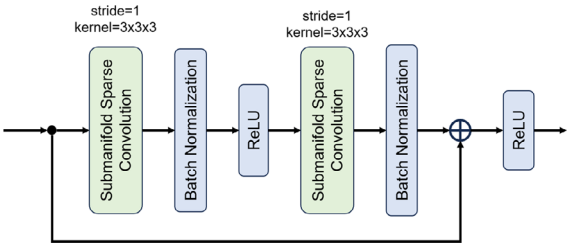


FIGURE 2 Convolution residual block.

The 3D backbone of the model expands the feature channels and downsamples the voxels in several stages. In each stage, several convolution residual blocks and LST blocks process voxel features in sequence with a fixed voxel and channel size. The structure of the convolution residual block is shown in Figure 2. This block has a residual connection that bypasses two 3D convolution layers. The LST blocks apply Transformer encoders to the voxel feature, and more details will be introduced in Section 3.2. All stages, except for the first, begin with a downsampling layer with stride 2 to downsample the voxels and increase the channels. Each convolution layer is followed by a batch normalization layer and ReLU activation function. There are two types of convolutions in the network: standard sparse convolution in the downsampling layers and submanifold sparse convolution in the other parts, which will be described in Section 3.3. Following all stages, the map-to-BEV module projects all the spatial voxel features

onto the XY plane by adding voxels with the same (x, y) coordinates. Thus, the output of the 3D backbone is a 2D feature map with resolution $L_x/d \times L_y/d$, where d is the downsampling factor.

The detection head has separate branches to regress each detection variable, namely, the confidence score of categories $\hat{Y}$, location refinement (dx, dy), Z coordinate z, object size (w, h, l), and orientation angle $(\sin(\theta), \cos(\theta))$. Each branch comprises a separate submanifold sparse convolution layer with kernel size 3 that maintains channel numbers and another convolution layer that alters the channel number to the output number. Additionally, a sparse convolution layer shared by all branches is adopted before separated branches. The anchor-free heatmap-based method (Yin et al., 2021) is adopted to regress the bounding box. The confidence score branch predicts a heatmap in which peaks indicate the presence of an object for each class.

The loss function comprises the classification loss and the regression loss of bounding boxes. The classification loss is a pixel-wise focal loss (Lin et al., 2018) given by:

$$L_C = -\frac{1}{N_{\text{gt}}} \sum_i \begin{cases} (1-\hat{Y}_i)^{\alpha} \log(\hat{Y}_i), \text{if } Y_i = 1 \\ (1-Y_i)^{\beta} \hat{Y}_i^{\alpha} \log(1-\hat{Y}_i), \text{otherwise} \end{cases} \quad (1)$$

where $\hat{Y}$ is the predicted heat map of confidence score, and i means each location on the feature map. α and β are hyper-parameters of the focal loss (Law & Deng, 2019). N_{gt} is the number of ground truths in a sample. The target heat map Y is given by:

$$Y(x,y) = \max_{i \in objects} \left[\exp\left(-\frac{(x-x_{c,i})^2 + (y-y_{c,i})^2}{2\sigma_i^2} \right) \right] \quad (2)$$

$$\sigma_i = \max(f(w_i, l_i), r) \quad (3)$$

where $(x_{c,i}, y_{c,i})$ is the center location of a ground truth, σ_i is the Gaussian radius determined by the object size (w_i, l_i) (Law & Deng, 2019), and the minimum radius r is adopted to ensure that small objects, like humans, have a large enough Gaussian peak for training. The regression loss is calculated by Equation (4) using the L1 loss. Notably, only the outputs at the ground truth center locations are taken into account for the regression loss. Finally, the overall loss function can be drawn as Equation (5), where λ_B is a hyper-parameter to balance the losses.

$$L_B = \frac{1}{N_{\text{gt}}} \sum_b \sum_{i=1}^{N} \left| \hat{b}_i - b_i \right|_{b \in \{dx,dy,z,w,l,h,\sin(\theta),\cos(\theta)\}} \quad (4)$$

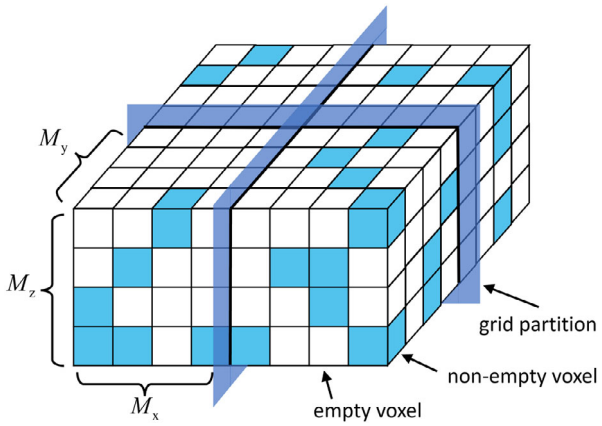


FIGURE 3 Voxels grouped into grids.

$$L_{\text{det}} = L_C + \lambda_B L_B \quad (5)$$

3.2 | LST block

Detecting workers in cluttered backgrounds containing numerous interference items (e.g., pillars, tools, and materials) that resemble human shapes can be challenging. Sparse convolution networks, which extract short-range features and focus on the local geometry of workers, could lead to poor detection. The model for construction detection is expected to have a large receptive field for long-range context modeling to understand the surroundings of a worker. Introducing the Transformer into the 3D worker detection framework is promising to solve this problem. The Transformer is supposed to be capable of attending to the most relevant region in the input data (Misra et al., 2021) and aggregate global information of large scenes (Mao et al., 2021). By connecting the features across a wide range, the model's receptive field can be enlarged. Giving larger weight to the more task-relevant part of the data may also help the model distinguish workers from the complicated environment. However, directly applying the Transformer to the global voxels is computationally expensive. Inspired by the Swin Transformer (Z.Liu, Lin, et al., 2021) that computes self-attention on dense pixels in non-overlapping windows, we apply the Transformer to sparse voxels in a local spatial grid to improve detection performance while maintaining computational efficiency.

The LST block comprises two consecutive Transformer encoder modules. The voxel space is initially divided into uniform cubic grids with a size of $M_x \times M_y \times M_z$, and the sparse voxels are grouped into these grids. It should be noted that the grid size M represents the number of voxels in each direction. Figure 3 illustrates the cubic grid partition and how the voxels are grouped into these grids.

Multi-head self-attention is then applied to the voxels in each grid independently. Each voxel serves as a query and attends to all the voxels in the same grid. Let H denote the number of heads, N_k denote the number of voxels in the kth grid, and $f_v \in \mathbb{R}^{N_k \times C/H}$ denote the input voxel features in the kth grid of one head. Due to the nonuniformity of sparse voxels, the N_k may differ for two grids. Within each head, the multi-head self-attention first calculates the query Q, key K, and value V using linear projection:

$$Q = f_v W_Q, K = f_v W_K, V = f_v W_V \tag{6}$$

where W_Q, W_K, W_V are the weight matrix of the projection for query, key, and value, respectively. Then, self-attention f_{attn} can be calculated as a function of Q, K, V:

$$f_{\text{attn}}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + PE)V \tag{7}$$

where PE is the relative position encoding calculated from another linear projection (represented by matrix W_{pe}) of two voxels' coordinates (p_i and p_j) difference:

$$PE_{ij} = (p_i - p_j)W_{pe} \tag{8}$$

Each head processes C/H feature channels, and the outputs of the H heads are concatenated after self-attention, which restores the number of channels to C. Either encoder in the LST block can be formulated as

$$\begin{aligned} f' &= \text{MSA}(\text{LN}(f_v), PE) + f_v \\ f_{\text{out}} &= \text{MLP}(\text{LN}(f')) + f' \end{aligned} \tag{9}$$

where MSA(·) stands for the multi-head self-attention operation as Equations (6) and (7), LN(·) represents layer normalization, and MLP(·) represents multi-layer perceptron. Note that the above description shows the calculation of a single grid, and each grid works in the same way as Equation (9). However, if the grid partition pattern remains consistent in all blocks, the voxel cannot attend to the voxels in the neighboring grid. To facilitate information flow across the voxel space, shifting grids between consecutive layers is necessary. Therefore, the first encoder module of the LST block employs a regular grid pattern that starts from the voxel with the minimum coordinate, then the second encoder module shifts the grid pattern by $(M_x/2,\ M_y/2,\ M_z/2)$. The whole architecture of the LST block is illustrated in Figure 4.

Compared to the Transformer on global voxels with a computation complexity of $O(N^2)$, the LST block reduces it to $O(N \times \overline{N_k})$, where $\overline{N_k}$ is the mean number of voxels in each grid. While Swin Transformer computes all the windows in a batch, this approach is not viable for the LST block since the number of voxels in each grid varies. To reduce computations and memory usage for

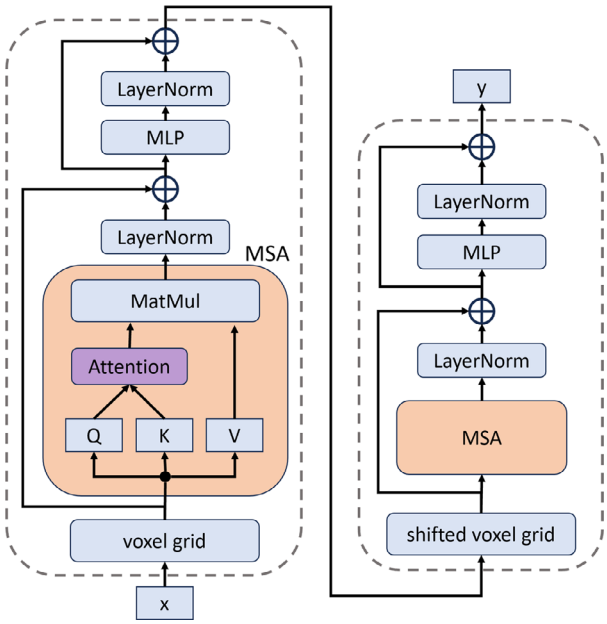


FIGURE 4 Local Sparse Transformer (LST) block.

the LST block, we opted to compute attention weight QK^T from all the query voxels in parallel, regardless of the grids. This involves $\sum_k N_k^2$ dot products, and the voxel pairs required in the products can be determined in advance. After obtaining the attention weight, scatter Softmax and scatter add (Fey, 2023) can be used to accelerate the remaining computation in Equation (7). Despite the improvements mentioned above, the Transformer still incurs more computation and memory costs than convolution. Thus, convolution and Transformer are used in the 3D backbone together. In particular, the attention mechanism cannot function effectively without the voxel feature embedding obtained from convolution (Lai et al., 2022). Therefore, another benefit of this design is that convolution can aid in calculating voxel feature embedding before the LST, which prevents the Transformer from dealing directly with the raw point feature.

3.3 | 3D sparse convolution networks

3.3.1 | Sparse convolution layers

In the voxel-based approach, directly using 3D convolution networks (Zhou & Tuzel, 2018) to process voxel features can be computationally time-consuming. Considering the sparsity of input data, where a significant proportion of voxels are empty, 3D sparse convolution (Graham & van der Maaten, 2017) can efficiently extract voxel features, compared to traditional convolution. Depending on the output location, sparse convolution can be classified

into standard sparse convolution (referred to as sparse convolution in the rest of the paper) and submanifold sparse convolution (Graham et al., 2017). In this study, submanifold sparse convolution constitutes most of the convolution operations in the backbone, and sparse convolution is primarily utilized in the downsampling layers.

Sparse convolution first dilates the input voxels with spatial locations V_{in} to their neighborhood within the kernel range. The output spatial locations V_{out} is the resultant voxel locations after dilation. This voxel dilation increases the number of non-empty voxels and can be expressed as follows:

$$V_{out} = \bigcup_{p \in V_{in}} \mathrm{D}(p) \tag{10}$$

where $D(\cdot)$ means the dilation operation that finds extra voxels according to a given voxel. In common sparse convolution practice, $D(i)$ finds all the voxels in the kernel range of a voxel with location p. Sparse convolution will only compute the output features for these active voxels in V_{out} as:Li

$$y_j = \sum_{k \in K(j)} w_k x_{j+k}, p_j \in V_{out} \tag{11}$$

where y_j is the output at voxel j with location p_j, x_{j+k} is the input voxel at location p_{j+k}, w_k is the kernel weight at offset k, $K(j)$ is the set of kernel offset. For example, for kernel size $3 \times 3 \times 3$ the size of $K(j)$ is 27. The term "$j + k$" means applying the kernel offset to voxel location p_j.

In contrast, submanifold sparse convolution regards a voxel as active and computes its output feature only if the voxel is non-empty in the input. The convolution computation is still expressed by Equation (11) except that $V_{out} = V_{in}$, which means it does not have a voxel dilation operation and thus does not change the non-empty voxel number.

3.3.2 | XY dilation strategy

Employing common sparse convolutions in construction scenarios is challenging and problematic. The uneven and cluttered construction surface has been identified as a major challenge. Moreover, it is discovered that the point cloud of a construction site has a long span in the X and Y directions but a short span in the Z direction, while workers mainly have large Z spans rather than X or Y. Therefore, the common dilation operation in all three directions (denoted as XYZ dilation) mainly increases the voxel number in the Z direction. One remarkable example is that the construction surface has around a three-time voxel increase in the Z direction. On one hand, it adds a significant amount of useless background information to the scene, which adversely affects the foreground object detection performance. On the other hand, a dilation operation is necessary for the sparse convolution in downsampling layers; otherwise, a significant amount of voxel information will be lost. However, the dilation destroys the sparsity of data and increases the number of non-empty voxels, which leads to high computation costs.

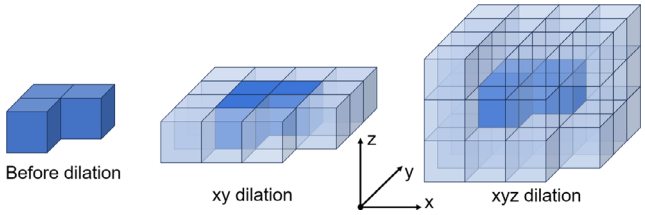


FIGURE 5 Dilation in sparse convolution.

Therefore, a dilation strategy named XY dilation is proposed to suppress the redundant XY-distributed background features while maintaining the important Z-distributed worker features. J. Liu et al. (2022) prune a ratio of dilated voxels that have small feature magnitude, but its voxel selection additionally costs computation. Instead, our XY dilation strategy is quite simple yet efficient, which only dilates the voxel in the X and Y directions, as illustrated in Figure 5. The proposed operation for each voxel at position $p_i = (x_i, y_i, z_i)$ can be formulated as Equation (12), where x_k and y_k represent the X and Y components of the kernel, respectively. The offset component in Z is zero, which means the voxels do not expand in Z direction. When this strategy is used in downsampling layers, the voxel number in subsequent stages becomes much smaller, and the model can save much computation and memory in training, particularly when involved with the Transformer. Furthermore, the XY dilation strategy has no adverse effect on the detection performance, and more details are shown in Section 5.2.

$$\mathrm{D}_{\mathrm{XY}}(p_i) = p_i \cup \{p_i + (x_k, y_k, 0)\}, k \in K(i) \tag{12}$$

3.4 | Evaluation metrics

The mean average precision (mAP) was used to evaluate the detection performance of the model. Following the practice of Nuscenes (Caesar et al., 2020), the predictions and ground truths are matched using 2D center distance on the ground plane rather than intersection over union (IOU). Specifically, a prediction is matched with a ground truth that has the smallest center distance up to a certain distance threshold d_{Thre}. For each class, precision and recall can be calculated from the number of assigned positive labels and ground truth numbers as Equation (13). For

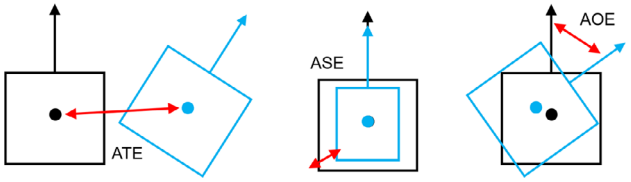


FIGURE 6 True positive metrics. AOE, average orientation error; ASE, average scale error; ATE, average translation error.

a given distance threshold the average precision (AP) of one class is calculated by integrating the precision–recall curve PR(•) given by Equation (14). The mean AP of all classes at a given distance threshold x is denoted as AP_x. The mAP is the mean value of the APs of all the classes and distance thresholds. The match threshold distances are set as 0.1, 0.15, 0.25, and 0.5 m to get metrics under different task difficulties.

Besides, the distance match does not consider the box matching degree, so three true positive metrics are introduced to better understand the magnitude of detection box errors: (1) average translation error (ATE), which measures Euclidean center distance in meters; (2) average scale error (ASE), which is calculated as 1 – IOU after aligning centers and orientations; and (3) average orientation error (AOE), which measures the smallest heading angle difference between prediction and ground-truth in radians, as illustrated in Figure 6. To obtain these metrics, the cumulative mean at each recall is first calculated and then the average at all recalls. These true positive metrics are calculated under the distance match threshold of 0.5 m.

$$\begin{cases} precision = \dfrac{N_{\text{prediction}}(distance<d_{\text{thre}})}{N_{\text{prediction}}} \\ recall = \dfrac{N_{\text{prediction}}(distance<d_{\text{thre}})}{N_{\text{ground truth}}} \end{cases} \tag{13}$$

$$\text{AP} = \int_0^1 \text{PR}(recall)\text{d}recall \tag{14}$$

4 | EXPERIMENT

4.1 | Dataset

In order to train and evaluate the proposed model, a point cloud dataset was created, where each sample is a scan of construction sites. The equipment employed for data collection is Robosense RS-LiDAR-M1 LiDAR (Robosense, 2023) as shown in Figure 7. Its horizontal and vertical field of view are 120 and 25 degrees (±12.5 degrees), respectively. Its horizontal and vertical resolution can both reach 0.2 degrees, with a frame rate of 10~20 Hz. The data were collected at construction sites in China with various construction types and stages, such as excavation, building foundations, and steel and concrete substructures. There are complex disturbances in the scene, such as dynamic equipment, excavated ground, fences, and scattered materials. During data collection, the LiDAR was installed on a tripod and moved around manually to scan the varying areas and angles of the sites. The workers engaged in construction activities without interference, exhibiting various poses of a construction nature. The workers' privacy was protected as their faces were not visible. After data collection, point cloud frames were carefully selected from the raw stream to ensure that there are many workers in the field of view and to cover different construction scenarios as much as possible. Finally, 3040 point-cloud samples were obtained, which were randomly divided into 2280 frames for training and 760 frames for testing. Figure 8 shows two examples of the point cloud in the dataset.

FIGURE 7 LiDAR setup.

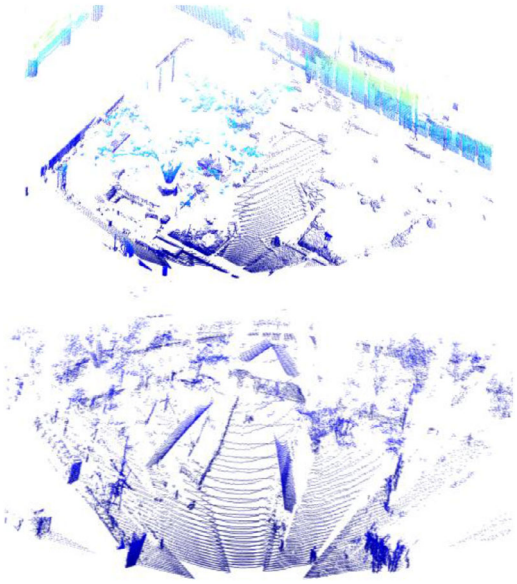
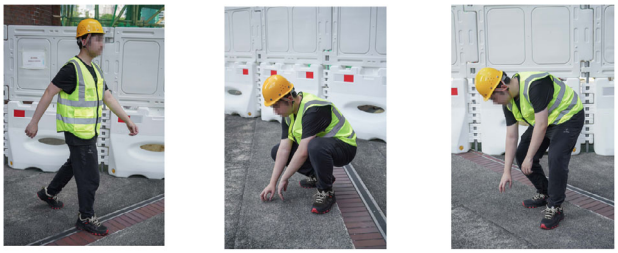


FIGURE 8 Examples of point clouds.

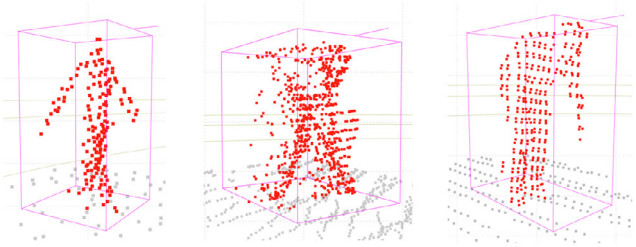
TABLE 1 Sample and object number in the dataset.

	Samples	All workers	Walking worker	Stooping worker	Squatting worker
Training set	2280	7098	2606	2375	2117
Test set	760	2369	872	795	702
Total	3040	9467	3478	3170	2819



(a) Walking (b) Squatting (c) Stooping

FIGURE 9 Worker postures in the dataset.



(a) Walking (b) Squatting (c) Stooping

FIGURE 10 Point cloud examples for workers in three postures. LiDAR, light detection and ranging.

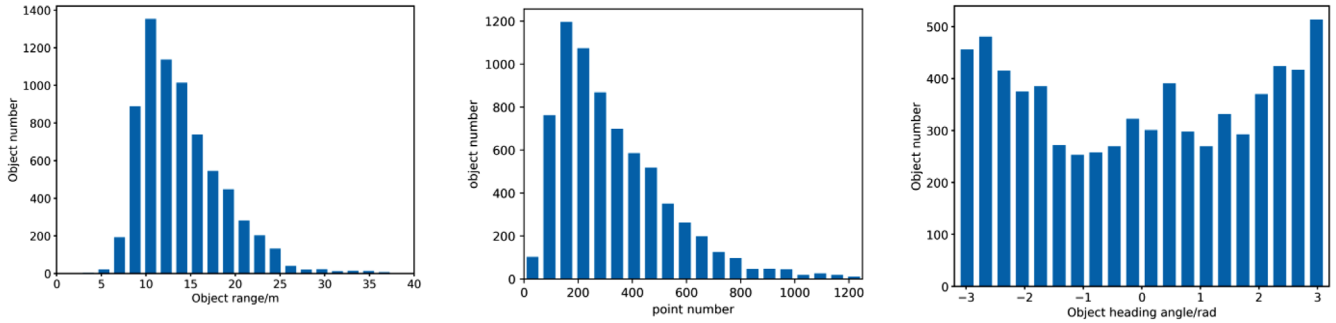
The SUSTechPOINTS open-source annotation tool (E. Li et al., 2020) was utilized to annotate the dataset, which involves assigning a 3D box and a posture label for each worker. For each worker's annotation box, the rotation needs to be manually adjusted to match the worker's orientation, and the size needs to be resized to fit profiles. Each worker is labeled with one of the three postures walking, stooping, and squatting. These three postures are representative of construction activities, and they are distinctive from each other as presented in Figures 9 and 10. It is worth noting that most construction activities, such as carrying, bricklaying, plastering, and rebar binding, can be categorized into one of the three postures by the relative angle between the torsos and legs. This ensures that the model can recognize workers in different postures rather than only detecting the common walking ones.

The quantity information of the dataset is shown in Table 1. There are 9467 worker instances in the dataset. The number of workers in different postures is balanced. Figure 11 illustrates the distribution of ground truths in the dataset, including sensing distance, point number, and heading angle. The heading angle is defined as the angle between the direction in which the worker is facing and the scanning direction of the sensor as illustrated in Figure 12. The worker instances cover distance in the range of [5, 35 m] and point number in the range of [10, 1300]. Considering that longer sensing distance of objects will result in sparser points and increase detection difficulty, various sensing distances were contained in the dataset. The dataset still contains difficult instances whose range is beyond 30 m and point number less than 70, which can help the model learn to recognize workers from very sparse points. The heading angles of the workers are uniformly distributed in all directions (from $-\pi$ to π), which helps the model to detect workers with different orientations. The dataset also contains around 200 occluded workers whose bodies are only partly visible from the sensor. In a word, the dataset involves a variety of compositions in terms of sensing distance, point sparsity, heading angle, and occlusion.

4.2 | Implementation details

4.2.1 | Network setting

The X direction heads the center of the LiDAR's field of view, and the Z direction is vertical up. The cubic range to crop the point cloud is [6, 42 m], [−18, 18 m], [−2, 1 m] for X, Y, and Z axes, respectively. The default voxel size (s_x, s_y, s_z) is set as (0.075, 0.075, 0.15 m), resulting in 480 × 480 × 20 voxels. The voxel size follows the optimized setting of existing methods (Yin et al., 2021), and it is set smaller due to the higher resolution of the LiDAR. The maximum number of voxels is 12,000, and no more than 10 points are allowed for each voxel. The detailed network hyper-parameter settings are listed in Table 2. Three stages were adopted in the backbone to downsample input by a factor of 4. As a result, the input spatial size 480 × 480 × 20 of voxels is changed to 120 × 120 × 5 after the backbone. The numbers of residual convolution blocks used in the three stages are {1, 1, 1} while that of LST blocks are {2, 2, 2}. The channel sizes in stages are {64, 128, 128}, which are the same for the residual convolution block and LST block. The head numbers of multi-head self-attention are set as {4, 8, 8} for the three stages, and the grid size $M_x \times M_y \times M_z$ for LST is 10 × 10 × 10 in all stages. The above hyper-parameter



(a) Objects' range (b) Objects' point number (c) Objects' heading angle

FIGURE 11 Distribution of the objects in the dataset.

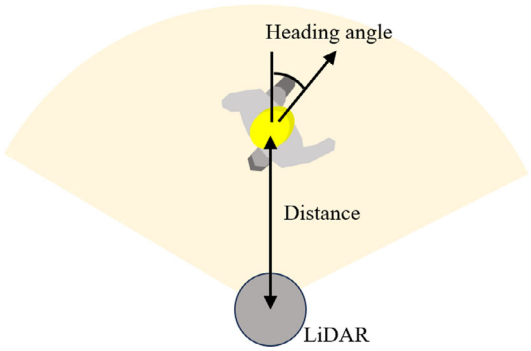


FIGURE 12 Heading angle definition.

setting mainly follows the common practice (Z.Liu, Lin, et al., 2021; B. Zhu et al., 2019) and is adjusted appropriately to suit the dataset and worker detection task.

4.2.2 | Training setup

During the training process, multiple data augmentation techniques were employed to improve the model's generalization. The data augmentation used includes:

1. Translation with a vector randomly sampled from the range [−0.5, 0.5 m], [0, 0], [−0.25, 0.25 m] for X, Y, Z axes.
2. Flip along the Y axis.
3. Scale with a factor sampled from the range [0.95, 1.05].
4. Rotation along the Z axis between the range of [−0.3925, 0.3925 rads].
5. Ground truth augmentation (Y. Yan et al., 2018).

The ground truth augmentation involves placing additional ground truths into the samples. Before training, a ground truth database including the annotations and points of all the ground truths in the training set was created. Then, during training, several ground truths were randomly selected from the database and placed into the sample. The maximum number of ground truths of workers in one posture for a sample was set to 10.

The model was implemented with PyTorch (Paszke et al., 2019), and the 3D sparse convolution network was implemented by Spconv (Y. Yan, 2023). The model was trained on an NVIDIA 4090 graphic card for 100 epochs with a batch size of 8. Adam method was used to optimize the model's parameters. The learning rate was adjusted following a one-cycle policy (Smith, 2018) with a maximum of 0.4, a division factor of 10, and a momentum from 0.95 to 0.85. Weight decay for regularization was fixed at 0.01. Loss balance parameter λ_B was set at 0.25.

5 | RESULT AND DISCUSSION

5.1 | Test result

5.1.1 | Overall result

The main detection performance on the test set is shown in Table 3, where the AP for a worker posture is the mean value across all distance thresholds. To demonstrate the benefits of the LST block, an ablated model that replaces all the LST blocks with the same number of sparse convolution layers is included. Furthermore, several voxel-based baseline models were selected for comparison as shown in Table 3. SECOND (Y. Yan et al., 2018) is the pioneer of sparse convolution networks; CenterPoint (Yin et al., 2021) has been a widely recognized benchmark in recent years; VoxelNeXt (Y. Chen et al., 2023) is the latest model modified from CenterPoint. These models adopted the same input voxel setting as the proposed method for a fair comparison and were trained and tested on the proposed dataset.

Remarkably, our model achieves a high $AP_{0.5}$ of 0.885 and an $AP_{0.25}$ of 0.877. The mAP reaches 0.807, demonstrating the model's strong ability to detect workers with

TABLE 2 Details of network setting.

Stages or branches	Operation and parameters	Channel size	Spatial size
Input	/	4	/
Preprocessing	Voxelization, mean	4	480 × 480 × 20
Input layer	3D SMSpconv, $k = 4 \times 64 \times (3,3,3)$, $s = 1$	64	480 × 480 × 20
Backbone Stage 1	3D SMSpconv, $k = 64 \times 64 \times (3,3,3)$, $s = 1$	64	480 × 480 × 20
Backbone Stage 1	LST, $g = 10 \times 10 \times 10$, $H = 4$, $C = 64$	64	480 × 480 × 20
Backbone Stage 2	3D Spconv, $k = 64 \times 128 \times (3,3,3)$, $s = 2$, dil = XY	128	240 × 240 × 10
Backbone Stage 2	3D SMSpconv, $k = 128 \times 128 \times (3,3,3)$, $s = 1$	128	240 × 240 × 10
Backbone Stage 2	LST, $g = 10 \times 10 \times 10$, $H = 8$, $C = 128$	128	240 × 240 × 10
Backbone Stage 3	3D Spconv, $k = 128 \times 128 \times (3,3,3)$, $s = 2$, dil = XYZ	128	120 × 120 × 5
Backbone Stage 3	3D SMSpconv, $k = 128 \times 128 \times (3,3,3)$, $s = 1$	128	120 × 120 × 5
Backbone Stage 3	LST, $g = 10 \times 10 \times 10$, $H = 8$, $C = 128$	128	120 × 120 × 5
Map to BEV	Feature add	128	120 × 120
Det head shared	2D SMSpconv, $k = 128 \times 128 \times (3,3,3)$, $s = 1$	128	120 × 120
Det head heat map	2D SMSpconv, $k = 128 \times 3 \times (3,3,3)$, $s = 1$	3	120 × 120
Det head x, y	2D SMSpconv, $k = 128 \times 2 \times (3,3,3)$, $s = 1$	2	120 × 120
Det head z	2D SMSpconv, $k = 128 \times 1 \times (3,3,3)$, $s = 1$	1	120 × 120
Det head w, l, h	2D SMSpconv, $k = 128 \times 3 \times (3,3,3)$, $s = 1$	3	120 × 120
Det head angle	2D SMSpconv, $k = 128 \times 1 \times (3,3,3)$, $s = 1$	1	120 × 120

Note: SMSpconv denotes submanifold sparse convolution, and Spconv denotes sparse convolution. k denotes kernel = in channel × out channel × (x, y, z). s denotes stride. H, g are head number and grid size. Spatial size is in $x \times y \times z$ or $x \times y$.
Abbreviation: BEV, bird's-eye-view.

TABLE 3 Detection performance comparison.

Model	mAP	$AP_{0.1}$	$AP_{0.15}$	$AP_{0.25}$	$AP_{0.5}$	Walking AP	Stooping AP	Squatting AP	ATE (cm)	ASE	AOE (rad)
SECOND	0.721	0.538	0.731	0.806	0.811	0.771	0.765	0.629	6.20	0.212	0.168
CenterPoint	0.739	0.560	0.765	0.814	0.819	0.770	0.771	0.678	6.07	0.213	0.173
VoxelNeXt	0.756	0.581	0.788	0.823	0.831	0.774	0.788	0.705	6.09	0.205	0.182
Model without LST	0.753	0.545	0.775	0.843	0.848	0.810	0.747	0.703	6.16	0.203	0.187
Ours	0.807	0.638	0.827	**0.877**	**0.885**	0.831	0.834	0.755	5.68	0.184	0.165

Abbreviations: AP, average precision; AOE, average orientation error; ASE, average scale error; ATE, average translation error; LST, Local Sparse Transformer; mAP, mean average precision.

both high recall and precision. Since the APs from threshold 0.5 to 0.1 m reflect increasing detection difficulty, there is a significant drop in $AP_{0.1}$, compared to other APs. For different postures, the detection performance of stooping and walking workers is excellent, while that of detecting squatting workers is relatively low. Squatting workers have fewer points than other types of workers and are more likely to generate false positive predictions. The true positive metrics (ATE, ASE, AOE) suggest that the matched predicted boxes have high quality and precisely restore the location, height, width, and orientation of workers. In addition to the high performance, the inference time for one frame is 70 ms (14.2 Hz) tested on an NVIDIA RTX3070 laptop graphics processing unit (GPU) and 36 ms (27.8 Hz) tested on an NVIDIA RTX 4090 GPU. This reaches near real-time detection, considering that a LiDAR typically operates at 10–20 Hz (Lang et al., 2019). The results demonstrate the high efficiency and capability of the proposed model.

Compared to the other models, the proposed model achieves the highest mAP and performance at different matching distance thresholds. It is noteworthy that the improvement in $AP_{0.1}$ is larger than that in other distance thresholds. This indicates that the proposed model has a greater advantage in difficult detection cases. In addition to the precision score, the errors of the true positive

TABLE 4 mAP within different distance range.

Distance/m	[4, 12]	[12, 20]	[20, 28]	[28, 36]
mAP	0.792	0.814	0.801	0.687

TABLE 5 mAP within different heading angle range.

Angle/rad	$[-\pi, -3\pi/4] \cup [3\pi/4, \pi]$	$[-3\pi/4, -\pi/4]$	$[-\pi/4, \pi/4]$	$[\pi/4, 3\pi/4]$
mAP	0.807	0.782	0.797	0.792

detection of the proposed model are slightly lower than in other models. It is worth noting that the model without the LST block still achieves slightly better performance than the baseline models, which suggests the superior performance of the optimized network architecture for worker detection. When the LST block is ablated, the proposed model experiences a significant drop in all APs, especially a huge performance drop in detecting stooping and squatting workers. This demonstrates that the LST block can effectively enhance worker detection performance, especially in scenarios where worker samples have smaller sizes and fewer points.

5.1.2 | Result in different conditions

The detection performance of the model can be further evaluated under different conditions, such as object's distance, heading angle, and occlusion. The whole distance and heading angle range are divided into several bins, and detailed mAPs are computed within the objects in each range bin. The results of our method are shown in Tables 4 and 5. The results indicate that, for most distance ranges, the detection performance remains at the same level as the overall mAP (Table 3). As the detection distance increases from 4 to 28 m, there is no decline in detection performance. However, in the range of the farthest detection distance (>28 m), the performance drops by 11% relative to the overall mAP. As for different heading angles, the detection performance shows no significant variation across the four intervals of π /4, indicating that the method can detect workers with various heading angles.

To better understand the strength of the model under different conditions, Figure 13 illustrates the prediction in several samples. In the figure, the blue and orange boxes represent the proposed model and the model without the LST block, respectively. The green boxes represent the ground truth. Workers in Figure 13a–c are relatively close to the LiDAR (<10 m) and have more points, where both models generate accurate bounding boxes. Workers in Figure 13d,c,f are far from the LiDAR (>20 m) and have fewer points. In this case, the model without LST blocks struggles to predict the accurate orientation and size, while the predictions of the proposed model still have high quality. The advantage of the proposed model can also be revealed where confusing items are involved in the detection. For example, the ablated model has a large error (Figure 13g,h) and generates false positive predictions (Figure 13i), while our model does not make this mistake. The local Transformer can endow the model with the ability to perceive a larger range of environmental context, which is useful for inferring the true result when the input is ambiguous.

When workers are occluded, only a partial body is visible in the point cloud, which results in limited information for the model and poses challenges for accurate worker detection. Some predictions of the model in case of occlusion are shown in Figure 14. In Figure 14a, different degrees of occlusion between workers are included, and the sample before the occurrence of occlusion is shown for comparison. The results demonstrate that the model is able to detect workers in most occlusion cases. The impact on the model's predictions is minimal when the occlusion is mild. However, severe occlusion can affect the quality of the model's predictions, especially in terms of angle estimation. In extremely severe occlusion scenarios, the model may miss some workers (Figure 14c). In conclusion, having sufficient information is crucial for the model's predictions. When the point cloud of a worker is incomplete, the model can extract contextual information from the remaining points and make comprehensive inferences.

Construction sites are one of the most complex and disorderly scenes in the industry, and each construction site may be vastly different from another. Therefore, it is important to evaluate the generalization of the model across different construction sites. All the above experiments were performed on the randomly separated training and test set. Here, the data from one specific construction site were reserved for testing, whereas the data from the rest of the sites were used to train the model. This leads to the model encountering previously unseen scenarios during the testing phase. In this case, the model achieved 0.785 mAP in the test. Although the performance slightly dropped, compared to that in Section 5.1.1, it is still a remarkable result. This shows that the proposed model has the potential for generalization across different construction sites.

5.2 | Ablation study

5.2.1 | Influence of XY dilation

As introduced in Section 3, an XY dilation strategy is proposed in sparse convolution downsampling layers to reduce the redundant voxels that may not contribute to

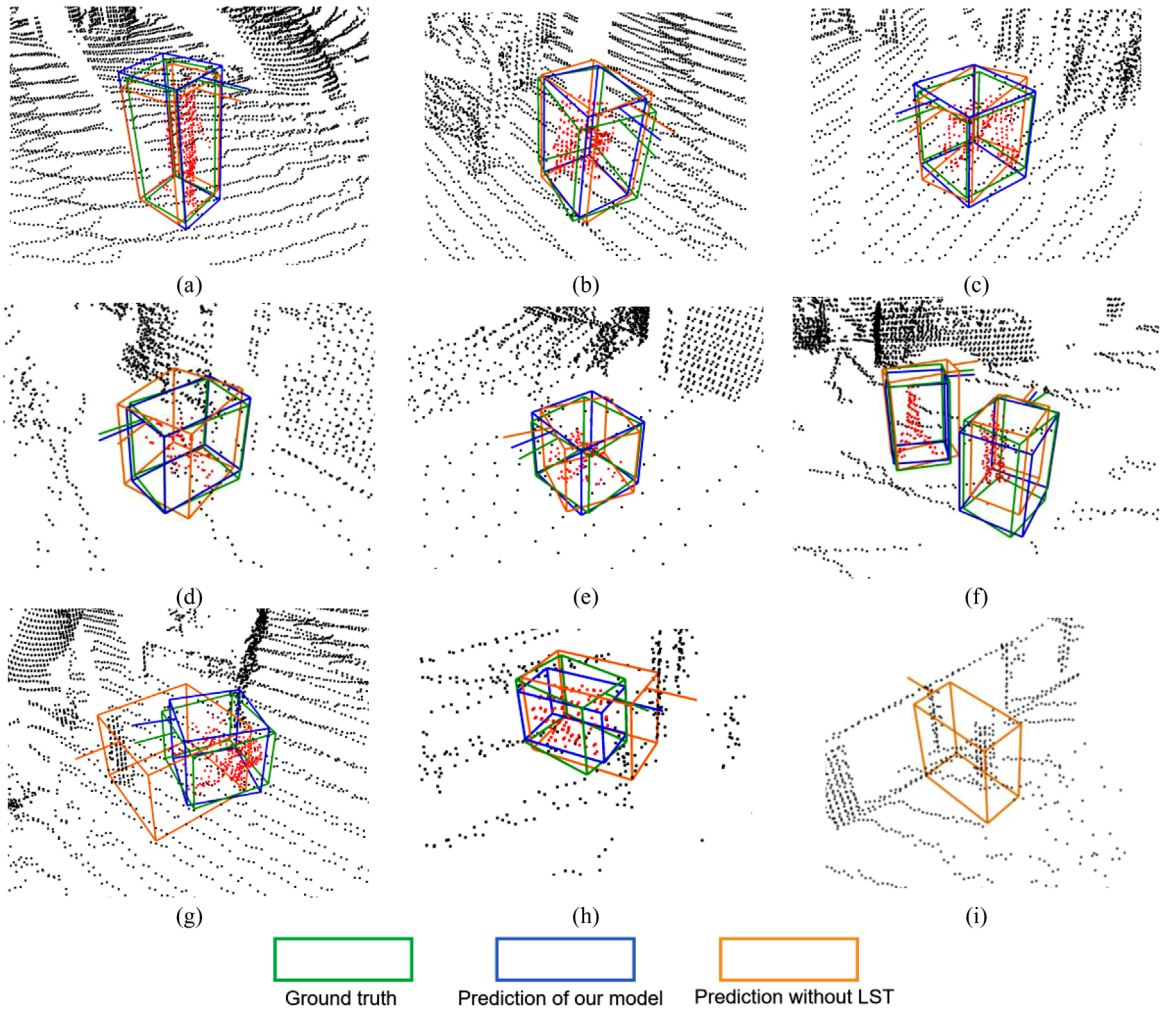


FIGURE 13 Detection results of the proposed model and that without LST block. (a), (b), and (c) workers in short distance; (d), (e), and (f) workers in far distance; (g), (h), and (i) mistakes with confusing items. Points in the ground truth boxes are visualized in red.

TABLE 6 Comparison of the efficiency of the dilation strategy.

Model	Dilation	Voxel number			mAP	Memory (GB)
		Stage 1	Stage 2	Stage 3		
D1	{XY, XY}	12k	6.2k	2.2k	0.801	3.0
D2	{XY, XYZ}	12k	6.2k	4.3k	0.807	5.1
D3	{XYZ, XYZ}	12k	12.9k	6.4k	0.792	9.5

the prediction. To demonstrate the effectiveness and to assess the influence on performance, three models with different dilation were tested and the results are shown in Table 6. The memory column denotes the GPU memory cost during training, and {XY, XYZ} refers to the first downsampling layer using XY dilation, the second downsampling layer using XYZ dilation, and so on. All the models have the same input voxel number (12k).

It is evident that when XY dilation is used, the voxel number is greatly reduced. When XY dilation is only used in the first downsampling layer (D2), the memory cost has already decreased to an adequate degree, meanwhile achieving the highest performance. Therefore, we consider XY dilation in the first downsampling layer to be necessary for effortless training. The performance change demonstrates that the voxels that are dilated in the Z direction are less important features and may adversely influence the detection. Figure 15 further compares the voxels that are obtained after XY and XYZ dilation. In XYZ dilation, the voxels representing the ground get a nearly three-fold increase in the Z direction, which is the majority of the voxel increase and an unnecessary burden on the data. In

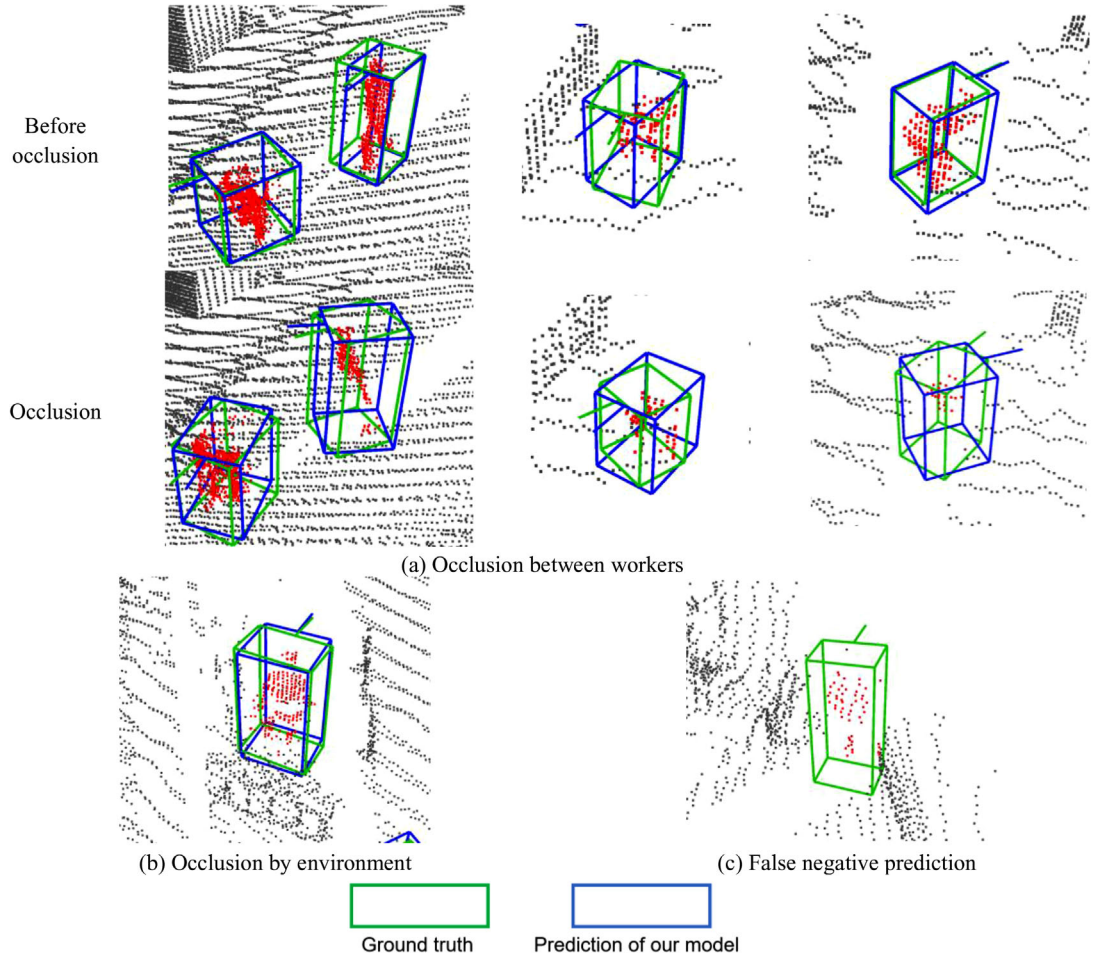


FIGURE 14 Predictions in case of occlusion.

contrast, in XY dilation, these voxels do not expand in the *Z* direction and are likely to be eliminated by the down-sampling operation. The implication of this section can be theoretically generalized to the scenes where the majority of point cloud distribution is in the XY direction.

5.2.2 | Influence of grid size

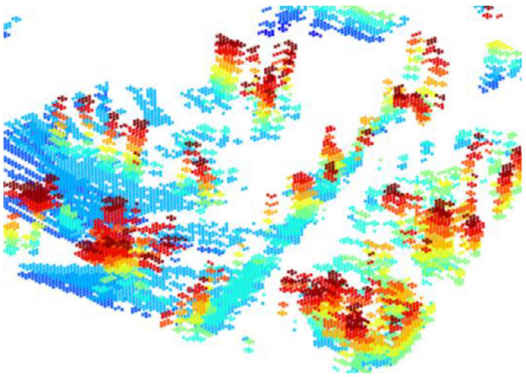
The most critical hyper-parameter of the LST block is the grid size, which determines the attention range of a query voxel. The testing results with changed grid size are shown in Table 7. The grid size (in voxel number) remains constant in all stages, but the actual attention range in the real world (in meters) varies due to down-sampling. The results demonstrate that the performance does not significantly vary with input sizes, and larger grid sizes do not always yield better performance. Specifically, model G3 achieved the highest mAP with a middle grid size. As the grid size increases from G1 to G3, the performance gradually improves, which could be attributed to the enlargement of the receptive field. However, when the grid size increases further to G4 and G5, the performance drops. It is probably because the large grids enclose much-cluttered background information, which may distract attention from the worker's features. Thus, the grid size should be carefully selected to ensure that the model incorporates the background context within an appropriate range. In addition to the performance drop, another consequence of large grid size is the unaffordable memory cost since the attention size in a grid is related to the cube of the grid size. For instance, G5 costs 12.2 GB of memory, which is 2.5 times that of G3. Therefore, the proposed model adopts a grid size of $10 \times 10 \times 10$, which is a compromise between performance and computation cost.

5.3 | Discussion

This study innovatively introduces the Transformer in the voxel-based deep-learning network to improve the performance of worker detection with LiDAR. Given the challenges of using LiDAR for worker detection in

TABLE 7 Influence of the grid size.

Model	Grid size	Grid actual range (m)			mAP	Memory (GB)
		Stage 1	Stage 2	Stage 3		
G1	5 × 5 × 5	0.375 × 0.375 × 0.75	0.75 × 0.75 × 1.5	1.5 × 1.5 × 3	0.803	2.4
G2	5 × 5 × 10	0.375 × 0.375 × 1.5	0.75 × 0.75 × 3	1.5 × 1.5 × 3	0.805	2.4
G3	10 × 10 × 10	0.75 × 0.75 × 1.5	1.5 × 1.5 × 3	3 × 3 × 3	0.807	5.1
G4	15 × 15 × 20	1.125 × 1.125 × 3	2.25 × 2.25 × 3	4.5 × 4.5 × 3	0.793	9.1
G5	20 × 20 × 20	1.5 × 1.5 × 3	3 × 3 × 3	6 × 6 × 3	0.794	12.2



(a) XYZ dilation

(b) XY dilation

FIGURE 15 The effect of XY dilation. The voxels at lower elevation (representing ground) are visualized in blue.

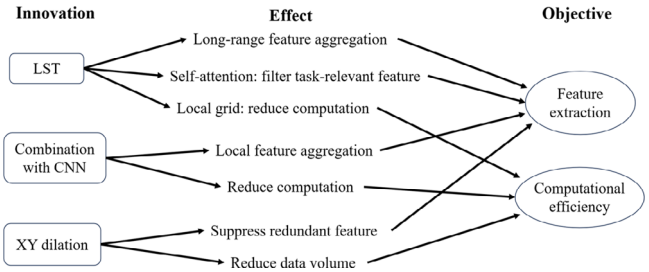


FIGURE 16 Map of main methodology innovations. CNN, convolutional neural network.

complex construction scenes, the 3D object detector needs to have strong feature extraction capabilities to aggregate task-relevant features (i.e., workers) from noisy inputs (i.e., complex sites). Additionally, computational efficiency is also a concern in this study due to the large amount of data in large-scale point clouds. In this context, the main innovations are summarized in Figure 16, and our knowledge contributions to the methodology include the following key components:

First, the innovative LST block is proposed that applies a Transformer encoder to sparse voxels in a local spatial grid. The LST block can connect all voxel features in a local grid in one operation. With proper grid size, it aggregates features of workers and semantic information about their surrounding complex environment. This greatly improves the receptive field of the backbone network and the efficiency of feature extraction. Furthermore, the multi-head self-attention mechanism in the LST block allows the model to adjust weights based on the computed feature similarity, ensuring that task-relevant features are given more attention. In addition to enhancing feature extraction capabilities, the proposed LST block reduces the computational complexity from quadratic to nearly linear by applying the Transformer in a local region, ensuring the computational efficiency of the model.

Second, the combination of CNN and Transformer is introduced at each stage. The Transformer is proficient at long-range associations but inefficient for extracting features of individual workers. On the other hand, CNN excels at short-range modeling, which helps identify the local geometric features of workers. Therefore, this hybrid network combines the benefits of capturing both broad semantic information and local details, which is particularly useful for detecting less obvious workers in complex scenes. Additionally, as the computational cost of CNN is much lower than that of the Transformer, the hybrid network alleviates overall computational overhead.

Third, the proposed XY dilation strategy replaces the original dilation in sparse convolution operations. The proposed XY dilation only expands voxels in the X and Y directions. Although this change is simple, it is highly effective. Since the background of a construction site is mainly distributed in the X and Y direction, while workers are distributed in the Z direction, XY dilation can adaptively prune unimportant features belonging to the

background and promote features belonging to workers to dominate the task. Furthermore, by suppressing the growth of redundant features, computational costs are significantly reduced.

Although initial progress has been achieved in this study, some limitations are worth noting. When the distance is large (>28 m), the detection performance decreases, which is mainly due to the lack of samples with distances greater than 28 m in the dataset. This problem can be hopefully improved by increasing the occurrence of samples from larger distances using dataset resampling techniques (B. Zhu et al., 2019). Another issue that affects detection performance is occlusion. In fact, when occlusion is extremely severe, it is difficult to infer full object information based on the current frame alone. Combining detection results from different time frames has the potential to address this issue (Z. Zhu et al., 2017). Last, although the dataset in this study attempts to cover a wide range of scenes, it is still relatively preliminary. Increasing the number of samples will allow fully exploiting the potential of deep learning. More construction sites of different types and in different regions need to be included to promote the applicability of the method. The samples of workers in far distances and occlusions also need to be expanded to enhance the robustness of the method.

6 | CONCLUSION

This study proposes an improved deep-learning model for detecting construction workers in 3D with LiDAR. To enhance worker detection performance in complex environments by enabling the model to understand contextual information surrounding workers, a novel LST block that applies multi-head self-attention in a local grid region is proposed. Moreover, to suppress redundant noisy features in large construction scenes, a novel XY dilation operation in sparse convolution layers is proposed, which can reduce the computation and memory cost while maintaining detection performance. In addition, a hybrid network architecture is adopted, which includes CNNs and Transformers, to aggregate local and global feature extraction while achieving a trade-off in computational cost. The model's architecture is also optimized for worker detection with delicate hyperparameter selection, such as layers, stages, and resolutions. To train and test the proposed model, a high-quality point cloud dataset was established.

The evaluation results indicate that the proposed model yields promising performance for worker detection over the baseline models. It can generate high-quality detection boxes with accurate location, size, and orientation predictions, while still maintaining a reasonable inference time. Compared to the model without LST blocks, the proposed model achieves better performance with fewer detection mistakes. Further ablation experiments investigated the operating principles of the LST block and demonstrated the effectiveness of XY dilation.

ACKNOWLEDGMENTS

This research was funded by the Start-up Fund for RAPs under the Strategic Hiring Scheme of the Hong Kong Polytechnic University (Grant Number P0042478), Internal Research Fund of PolyU (UGC) (Grant Number P0047899), and Guangdong-Hong Kong Technology Cooperation Funding Scheme (Grant Number GHP/166/21SZ).

REFERENCES

Allinson, M. (2022). *Construction robotics startup Canvas launches drywall finishing robot*. Robotics and Automation News. https://roboticsandautomationnews.com/2022/01/27/construction-robotics-startup-canvas-launches-drywall-finishing-robot/48705/

Al-Naji, A. A., Gibson, K., Lee, S.-H., & Chahl, J. (2017). Real time apnoea monitoring of children using the Microsoft Kinect sensor: A pilot study. *Sensors*, *17*, 286. https://doi.org/10.3390/s17020286

Beltrán, J., Guindel, C., Moreno, F. M., Cruzado, D., García, F., & De La Escalera, A. (2018). BirdNet: A 3D object detection framework from LiDAR information. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI (pp. 3517–3523). https://doi.org/10.1109/ITSC.2018.8569311

Business Research. (2023). *Autonomous construction equipment market size, trends and global forecast To 2032*. The Business Research Company. https://www.thebusinessresearchcompany.com/report/autonomous-construction-equipment-global-market-report

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., & Beijbom, O. (2020). nuScenes: A multimodal dataset for autonomous driving. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA (pp. 11618–11628). https://doi.org/10.1109/CVPR42600.2020.01164

Charles, R. Q., Su, H., Kaichun, M., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI (pp. 77–85). https://doi.org/10.1109/CVPR.2017.16

Chen, Q., Sun, L., Wang, Z., Jia, K., & Yuille, A. (2020). *Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots*. arXiv. http://arxiv.org/abs/1912.12791

Chen, Y., Liu, J., Zhang, X., Qi, X., & Jia, J. (2023). *VoxelNeXt: Fully sparse VoxelNet for 3D object detection and tracking*. arXiv. http://arxiv.org/abs/2303.11301

Dogan, F. I., Melsion, G. I., & Leite, I. (2023). Leveraging explainability for understanding object descriptions in ambiguous 3D environments. *Frontiers in Robotics and AI*, *9*, 937772. https://doi.org/10.3389/frobt.2022.937772

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An image is worth 16×16*

words: Transformers for image recognition at scale. arXiv. http://arxiv.org/abs/2010.11929

Eraliev, O. M. U., Lee, K.-H., Shin, D.-Y., & Lee, C.-H. (2022). Sensing, perception, decision, planning and action of autonomous excavators. *Automation in Construction*, *141*, 104428. https://doi.org/10.1016/j.autcon.2022.104428

Fang, W., Ding, L., Zhong, B., Love, P. E. D., & Luo, H. (2018). Automated detection of workers and heavy equipment on construction sites: A convolutional neural network approach. *Advanced Engineering Informatics*, *37*, 139–149. https://doi.org/10.1016/j.aei.2018.05.003

Fey, M. (2023). *torch-scatter: PyTorch extension library of optimized scatter operations (2.1.1)* [Python]. https://github.com/rusty1s/pytorch_scatter

Graham, B., Engelcke, M., & van der Maaten, L. (2017). *3D semantic segmentation with submanifold sparse convolutional networks*. arXiv. http://arxiv.org/abs/1711.10275

Graham, B., & van der Maaten, L. (2017). *Submanifold sparse convolutional networks*. arXiv. http://arxiv.org/abs/1706.01307

Guo, J., Han, K., Wu, H., Tang, Y., Chen, X., Wang, Y., & Xu, C. (2022). *CMT: Convolutional neural networks meet vision transformers*. arXiv. http://arxiv.org/abs/2107.06263

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., & Bennamoun, M. (2021). Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(12), 4338–4364. https://doi.org/10.1109/TPAMI.2020.3005434

He, C., Li, R., Li, S., & Zhang, L. (2022). *Voxel set transformer: A set-to-set approach to 3D object detection from point clouds*. arXiv. http://arxiv.org/abs/2203.10314

Laga, H., Jospin, L. V., Boussaid, F., & Bennamoun, M. (2020). A survey on deep learning techniques for stereo-based depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(4), 1738–1764. https://doi.org/10.1109/TPAMI.2020.3032602

Lai, X., Liu, J., Jiang, L., Wang, L., Zhao, H., Liu, S., Qi, X., & Jia, J. (2022). *Stratified Transformer for 3D point cloud segmentation*. arXiv. http://arxiv.org/abs/2203.14508

Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (2019). PointPillars: Fast encoders for object detection from point clouds. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA (pp. 12689–12697). https://doi.org/10.1109/CVPR.2019.01298

Law, H., & Deng, J. (2019). *CornerNet: Detecting objects as paired keypoints*. arXiv. http://arxiv.org/abs/1808.01244

Lee, Y.-J., & Park, M.-W. (2019). 3D tracking of multiple onsite workers based on stereo vision. *Automation in Construction*, *98*, 146–159. https://doi.org/10.1016/j.autcon.2018.11.017

Li, E., Wang, S., Li, C., Li, D., Wu, X., & Hao, Q. (2020). SUSTech POINTS: A portable 3D point cloud interactive annotation platform system. *2020 IEEE Intelligent Vehicles Symposium (IV)*, Las Vegas, NV (pp. 1108–1115). https://doi.org/10.1109/IV47402.2020.9304562

Li, J., Xia, X., Li, W., Li, H., Wang, X., Xiao, X., Wang, R., Zheng, M., & Pan, X. (2022). *Next-ViT: Next generation vision transformer for efficient deployment in realistic industrial scenarios*. arXiv. https://arxiv.org/abs/2207.05501v4

Li, W., Hu, Y., Zhou, Y., & Pham, D. T. (2023). Safe human-robot collaboration for industrial settings: A survey. *Journal of Intelligent Manufacturing*. Advance online publication. https://doi.org/10.1007/s10845-023-02159-4

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2018). *Focal loss for dense object detection*. arXiv. http://arxiv.org/abs/1708.02002

Liu, J., Chen, Y., Ye, X., Tian, Z., Tan, X., & Qi, X. (2022). *Spatial pruned sparse convolution for efficient 3D object detection*. arXiv. http://arxiv.org/abs/2209.14201

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In B. Leibe, J. Matas, N. Sebe, & M. Welling (Eds.), *Computer vision–ECCV 2016* (Vol. 9905, pp. 21–37). Springer International Publishing. http://link.springer.com/10.1007/978-3-319-46448-0_2

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin Transformer: Hierarchical vision Transformer using shifted windows. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada (pp. 9992–10002). https://doi.org/10.1109/ICCV48922.2021.00986

Liu, Z., Zhang, Z., Cao, Y., Hu, H., & Tong, X. (2021). *Group-free 3D object detection via Transformers*. arXiv. http://arxiv.org/abs/2104.00678

Malewar, A. (2019). *Spot robot is ready for on-site inspection at a large construction site*. InceptiveMind. https://www.inceptivemind.com/spot-robot-ready-site-inspection-large-construction-site/10359/

Mao, J., Shi, S., Wang, X., & Li, H. (2022). *3D object detection for autonomous driving: A review and new outlooks*. arXiv. http://arxiv.org/abs/2206.09474

Mao, J., Xue, Y., Niu, M., Bai, H., Feng, J., Liang, X., Xu, H., & Xu, C. (2021). *Voxel Transformer for 3D object detection*. arXiv. http://arxiv.org/abs/2109.02497

Martins, G. B., Papa, J. P., & Adeli, H. (2020). Deep learning techniques for recommender systems based on collaborative filtering. *Expert Systems*, *37*(6), e12647. https://doi.org/10.1111/exsy.12647

Ming, Y., Meng, X., Fan, C., & Yu, H. (2021). Deep learning for monocular depth estimation: A review. *Neurocomputing*, *438*, 14–33. https://doi.org/10.1016/j.neucom.2020.12.089

Misra, I., Girdhar, R., & Joulin, A. (2021). *An end-to-end Transformer model for 3D object detection*. arXiv. http://arxiv.org/abs/2109.08141

Park, H. S., Lee, H. M., Adeli, H., & Lee, I. (2007). A new approach for health monitoring of structures: Terrestrial laser scanning. *Computer-Aided Civil and Infrastructure Engineering*, *22*(1), 19–30. https://doi.org/10.1111/j.1467-8667.2006.00466.x

Park, M., Tran, D. Q., Bak, J., & Park, S. (2023). Small and overlapping worker detection at construction sites. *Automation In Construction*, *151*, 104856. https://doi.org/10.1016/j.autcon.2023.104856

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Garnett, R. (2019). PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, *32*, 8024–8035.

Pereira, D. R., Piteri, M. A., Souza, A. N., Papa, J. P., & Adeli, H. (2020). FEMa: A finite element machine for fast learning. *Neural Computing and Applications*, *32*(10), 6393–6404. https://doi.org/10.1007/s00521-019-04146-4

Qi, C. R., Litany, O., He, K., & Guibas, L. J. (2019). *Deep Hough voting for 3D object detection in point clouds*. arXiv. https://doi.org/10.48550/arXiv.1904.09664

Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, Long Beach, CA.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

Roberts, D., Torres Calderon, W., Tang, S., & Golparvar-Fard, M. (2020). Vision-based construction worker activity analysis informed by body posture. *Journal of Computing in Civil Engineering*, *34*(4), 04020017. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000898

Robosense. (2023). *Automotive grade LiDAR RS-LiDAR-M1 RoboSense LiDAR for autonomous driving, robots*. Robosense. https://www.robosense.cn/en/rslidar/RS-LiDAR-M1

Shen, J., Yan, W., Li, P., & Xiong, X. (2021). Deep learning-based object identification with instance segmentation and pseudo-LiDAR point cloud for work zone safety. *Computer-Aided Civil and Infrastructure Engineering*, *36*(12), 1549–1567. https://doi.org/10.1111/mice.12749

Smith, L. N. (2018). *A disciplined approach to neural network hyper-parameters: Part 1—Learning rate, batch size, momentum, and weight decay*. arXiv. https://doi.org/10.48550/arXiv.1803.09820

Son, H., Choi, H., Seong, H., & Kim, C. (2019). Detection of construction workers under varying poses and changing background in image sequences via very deep residual networks. *Automation In Construction*, *99*, 27–38. https://doi.org/10.1016/j.autcon.2018.11.033

Son, H., & Kim, C. (2021). Integrated worker detection and tracking for the safe operation of construction machinery. *Automation in Construction*, *126*, 103670. https://doi.org/10.1016/j.autcon.2021.103670

Son, H., Seong, H., Choi, H., & Kim, C. (2019). Real-Time vision-based warning system for prevention of collisions between workers and heavy equipment. *Journal of Computing in Civil Engineering*, *33*(5), 04019029. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000845

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, Long Beach, CA.

Yan, X., Zhang, H., & Li, H. (2020). Computer vision-based recognition of 3D relationship between construction entities for monitoring struck-by accidents. *Computer-Aided Civil and Infrastructure Engineering*, *35*(9), 1023–1038. https://doi.org/10.1111/mice.12536

Yan, Y. (2023). *spconv: Spatial sparse convolution (2.3.6) [Python]*. https://github.com/traveller59/spconv

Yan, Y., Mao, Y., & Li, B. (2018). SECOND: Sparsely embedded convolutional detection. *Sensors*, *18*(10), 3337. https://doi.org/10.3390/s18103337

Yang, B., Luo, W., & Urtasun, R. (2018). PIXOR: Real-time 3D object detection from point clouds. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT (pp. 7652–7660). https://doi.org/10.1109/CVPR.2018.00798

Yin, T., Zhou, X., & Krahenbuhl, P. (2021). Center-based 3D object detection and tracking. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN (pp. 11779–11788). https://doi.org/10.1109/CVPR46437.2021.01161

Zhou, Y., & Tuzel, O. (2018). VoxelNet: End-to-end learning for point cloud based 3D object detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT (pp. 4490–4499). https://doi.org/10.1109/CVPR.2018.00472

Zhu, B., Jiang, Z., Zhou, X., Li, Z., & Yu, G. (2019). *Class-balanced grouping and sampling for point cloud 3D object detection*. arXiv. http://arxiv.org/abs/1908.09492

Zhu, Z., Ren, X., & Chen, Z. (2017). Integrated detection and tracking of workforce and equipment from construction jobsite videos. *Automation in Construction*, *81*, 161–171. https://doi.org/10.1016/j.autcon.2017.05.005

How to cite this article: Zhang, M., Wang, L., Han, S., Wang, S., & Li, H. (2024). Deep learning framework with Local Sparse Transformer for construction worker detection in 3D with LiDAR. *Computer-Aided Civil and Infrastructure Engineering*, *39*, 2990–3007. https://doi.org/10.1111/mice.13238