Received: 13 January 2025 | Accepted: 16 March 2025

DOI: 10.1111/mice.13500

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING WILEY

RESEARCH ARTICLE

Excavator 3D pose estimation from point cloud with self-supervised deep learning

Mingyu Zhang[1] | **Wenkang Guo**[1] | **Jiawen Zhang**[1] | **Shuai Han**[1] | **Heng Li**[1] | **Hongzhe Yue**[2]

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

[2]School of Civil Engineering, Southeast University, Nanjing, China

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

Funding information
National Natural Science Foundation of China, Grant/Award Number: 42302322; Internal Research Fund of PolyU (UGC), Grant/Award Number: P0047899; Collaborative Research Fund (CRF) from the Research Grants Council (Hong Kong), Grant/Award Number: C6044-23GF

Abstract

Pose estimation of excavators is a fundamental yet challenging task with significant implications for intelligent construction. Traditional methods based on cameras or sensors are often limited by their ability to perceive spatial structures. To address this, 3D light detection and ranging has emerged as a promising paradigm for excavator pose estimation. However, these methods face significant challenges: (1) accurate 3D pose annotations are labor-intensive and costly, and (2) excavators exhibit complex kinematics and geometric structures, further complicating pose estimation. In this study, a novel framework is proposed for full-body excavator pose estimation directly from 3D point clouds, without relying on manual 3D annotations. The excavator pose is parameterized using pose parameters of geometric primitives under kinematic constraints. A unified deep network is designed to predict pose parameters from point clouds. The network is initially pre-trained on synthetic data to provide parameter initialization and then fine-tuned using real-world data. To facilitate label-free training, the self-supervised loss functions are designed by exploiting the geometric and kinematic consistency between point clouds and excavators. Experimental results on real-world construction sites demonstrate the effectiveness and robustness of the proposed method, achieving an average pose estimation accuracy of 0.26 m. The method also exhibits promising performance across various excavator operational scenarios, highlighting its potential for real-world applications.

1 | INTRODUCTION

Excavators are vital and ubiquitous equipment in the construction industry. The pose estimation of excavators has received considerable attention from both industry and academia due to its extensive applications in enhancing safety and productivity monitoring in construction and advancing autonomous systems, human–robot interactions, and virtual/augmented reality. Accurately identifying the full-body pose of an excavator in 3D space—including the positions and orientations of its components, as well as its occupancy volume—is crucial for these applications (Y. Liu & Jebelli, 2024). For example, in excavation safety monitoring, it helps estimate the minimal

distance between workers and the excavator, thereby preventing human–machine collisions (K. Kim et al., 2023). It also enables automated excavators to assess their spatial relationship with dump trucks, facilitating autonomous loading operations (Eraliev et al., 2022).

Current methods for pose estimation of excavators primarily include sensor-based and camera-based approaches. Sensor-based techniques have been thoroughly studied (D. Liu et al., 2023; Vahdatikhaki et al., 2015), yet they are susceptible to damage from intense vibrations and experience measurement drift over time. Camera-based methods have attracted substantial interest (Bang et al., 2021) with 2D-to-3D lifting or depth estimation techniques (Shen et al., 2024). However, estimating 3D poses using monocular cameras presents significant challenges, such as inadequate lighting, depth ambiguity, and scale variability arising from the absence of depth references. Existing camera-based approaches generally estimate 3D poses only under constrained conditions and with considerable inaccuracies (Assadzadeh et al., 2023). While stereo cameras can provide depth estimation, they compromise with calibration, installation, and computational efficiency (Soltani et al., 2018).

To overcome these limitations and better perceive spatial structures, light detection and ranging (LiDAR)-based methods have emerged as a promising paradigm for pose estimation (Y. Li & Ibanez-Guzman, 2020). The 3D point clouds generated by LiDAR effectively capture the detailed and long-range spatial geometry (H. S. Park et al., 2007), thus providing solid support for 3D pose. However, current LiDAR-based studies for excavator pose estimation remain significantly under-researched with a limited number (Cho & Gai, 2014; Kashani et al., 2010). These studies employed old techniques and incomplete pose representation, yielding inadequate accuracy.

This study focuses on the deep learning techniques for 3D pose estimation from point clouds (Yue et al., 2024). Although such methods have seen limited application for excavators, they are extensively employed in the studies of humans and rigid objects (Rafiei et al., 2024). Specifically, most 3D pose estimation methods rely on supervised learning (Wang et al., 2019; Zanfir et al., 2023), with 3D pose ground truth as supervision. However, they present several limitations. Annotating 3D poses is both costly and labor-intensive, requiring either precise manual adjustments (P. Sun et al., 2020) or the use of expensive motion capture equipment (Ionescu et al., 2014). In addition, manual 3D annotations are prone to errors, introducing bias into model learning. Furthermore, such methods often struggle to generalize effectively to scenarios not covered by the available annotations.

Data synthesis, that is, generating 3D data and annotations by simulation and computation, has facilitated the supervised training for 3D fitting tasks (Xiang et al., 2020). However, synthetic 3D data are characterized by overly idealized geometric shapes, the absence of natural signal noise (from systems, dust, or reflective surfaces), and inadequate coverage of corner cases, which differs from real-world distributions. This discrepancy leads to models trained solely on synthetic data struggling to generalize to real-world applications. It is noteworthy that large volumes of unlabeled 3D data inherently possess rich geometric features, such as alignment with known 3D primitives (Kluger et al., 2024) and kinematic consistency (Z. Weng et al., 2023). Based on this, unsupervised or self-supervised 3D data parsing methods have been an emerging field of research, which offers potential solutions to the challenges of annotation scarcity and performance gaps in real-world data.

In this study, an effective and flexible method is proposed for full-body pose estimation of excavators from 3D point clouds, which eliminates the need for manual 3D annotations. Our approach consists of two key stages to train a pose estimation network: supervised pretraining and self-supervised fine-tuning. First, the network is pre-trained on a synthetic dataset, where both excavator point clouds and pose labels are readily available. Then, the network is fine-tuned on unannotated real-world datasets using carefully designed self-supervised loss functions. This loss leverages the geometric and kinematic consistency of excavators to align the predicted poses with the actual point clouds, allowing the model to adapt to the real-world domain without requiring 3D pose labels. Our contributions are summarized as follows:

1. A full-body pose representation for excavators: The approach explicitly models the pose of all excavator components, considering their unique motion constraints and articulated structures.
2. A unified pose-estimation network: We present a deep neural network that predicts all pose parameters of the excavator, including positions, rotations, and dimensions, in a unified architecture.
3. Novel self-supervised losses for label-free training: By aligning the predicted poses with the observed point clouds, the self-supervised loss functions are designed to train the pose estimation network without pose annotations.
4. Validation on real-world data: Extensive experiments on real-world datasets demonstrate the effectiveness and robustness of the proposed method across various excavator operational scenarios.

The code will be available at: https://github.com/fishfen/EPEP.

2 | RELATED WORK

2.1 | Excavator pose estimation

Sensor-based methods determine the positions of excavator components by inertial measurement units (IMUs), global positioning system (GPS), or radiofrequency sensors such as ultra-wide band and radio frequency identification. For instance, GPS is employed to provide precise positioning of the excavator, while IMUs are mounted on various components to estimate tilt angles (D. Liu et al., 2023). However, sensor-based methods face several practical limitations. IMUs are prone to measurement drift (over 1 m) or even damage resulting from vibrations and magnetic interference (J. Park et al., 2017). GPS signals are easily obstructed in urban settings. Additionally, these methods necessitate the installation of tags on equipment and calibration, rendering the deployment of the entire system cumbersome and inflexible.

Camera-based methods often utilize depth, stereo, and monocular cameras. Depth cameras have a limited sensing range and are susceptible to strong light, making them unsuitable for outdoor environments (Shen et al., 2021). Stereo cameras estimate distance through triangulation from multiple viewpoints; however, issues with camera calibration and perspective distortion result in inevitable inaccuracies (Soltani et al., 2018). Monocular camera-based methods typically employ supervised learning techniques to estimate 3D coordinates. The 3D pose ground truth comes from robotic arm encoders (Liang et al., 2019), manual overlay of 3D models (J. Kim et al., 2023), and synthetic datasets (Torres Calderon et al., 2021). Among them, synthetic excavator images seem to be inexpensive, but the discrepancies between synthetic and real-world scenarios question its generalization (Mahmood et al., 2022). In addition, such approaches could be constrained by specific camera parameters (Assadzadeh et al., 2023). As depth information is absent in an image, estimating 3D coordinates from a single image presents an ill-posed problem, challenging the reliability of the 3D predictions.

Compared to other methods, LiDAR-based ones directly capture depth information through 3D point clouds, offering advantages like flexibility of deployment, as well as independent of lighting conditions. Few studies have explored the pose estimation of excavators from LiDAR point clouds. Some of them have simplified pose estimation to a point cloud registration problem, focusing solely on the pose of a rigid component rather than the full body (Cui et al., 2022; Xu et al., 2023). Cho and Gai (2014) exployed Computer-Aided Design (CAD) models of excavators for registration with point clouds to estimate pose. However, this method depends on pre-acquired CAD models and is hard to accommodate various models and sizes. Furthermore, point cloud registration-based methods generally suffer from large computational latency (Phillips & McAree, 2018). In this work, however, an effective and flexible framework is proposed to estimate the full-body pose of excavators using only LiDAR points as input.

2.2 | Pose estimation from point clouds

Estimating 3D poses from point clouds targets a variety of objects such as human bodies (X. Sun et al., 2018), rigid bodies (Wang et al., 2019), hands (Cheng et al., 2021), and mechanical structures (Cui et al., 2022). The majority rely on full-supervised learning, necessitating large datasets with delicate 3D annotations. For instance, models can learn the proximity to key points (Ge et al., 2018) or directly output coordinates using integral regression (X. Sun et al., 2018) under the supervision of key point ground truth. With 3D-oriented bounding box annotations, models can learn the point-wise position in the normalized object coordinate space (Wang et al., 2019), joint axis direction (X. Li et al., 2020), and rotation parameters (Y. Weng et al., 2021). Despite these advances in the supervised methods, annotating the 3D poses of real-world objects is costly and labor-intensive, requiring meticulous manual adjustments (P. Sun et al., 2020) or the use of expensive motion capture systems (Ionescu et al., 2014).

Recent research has explored pose estimation without manual 3D annotations. Weakly supervised approaches use 2D ground truth to supervise 3D (Zheng et al., 2022). Although 2D annotations are more accessible than 3D, aligning and synchronizing multiple modalities adds complexity. As an alternative, leveraging the geometric structure of point clouds to fit primitives presents a promising avenue for fully unsupervised pose estimation. For instance, PRIFIT (Sharma et al., 2022) fit ellipsoids to local point clusters by unsupervised reconstruction loss. Kluger et al. (2024) developed a sampling-based process to estimate cuboids in 3D scenes, but this method prioritizes simple scene descriptions over precise pose estimation. Recently, the geometry consistency inspired key point leaning (GC-KPL) model (Z. Weng et al., 2023) predicts key points of articulated objects by encouraging alignment with intrinsic kinematic structures. However, this approach is tailored toward humans, with shapes and poses distinct from excavators. In this work, a self-supervised method is proposed for excavator pose estimation based on the kinematics and geometric features of excavators.

3 | METHODOLOGY

3.1 | Preliminary formulation

3.1.1 | Excavator pose parametrization

A standard excavator consists of five primary components: the chassis, cab, boom, stick (also referred to as the dipper), and bucket, as shown in Figure 1. Each of these components is treated as a rigid body. The excavator operates under the following kinematic constraints: (1) The cab and chassis rotate around a shared main rotational axis that is perpendicular to the ground. (2) The main rotational axis passes through the geometric center of the chassis. (3) The joints connecting the three articulated arms (boom, stick, and bucket) allow rotation only along a single axis, restricting the arms to a single plane, referred to as the "arm plane."

In this study, the excavator's normalized coordinate system (NCS) is defined with the following parameters: The origin is at the center of the cab-chassis rotational joint, the z-axis aligns with the main rotational axis, and the y-axis is oriented perpendicular to the arm plane, illustrated in Figure 1. The transformation between the NCS and the sensor coordinate system—referred to as the LiDAR coordinate system (LCS)—is determined by a translation vector $\mathbf{T}$ and a rotation matrix $\mathbf{R}$. The full-body pose of an excavator can be described by the individual poses of its five main components. We propose to use geometric primitives (e.g., cuboids and links) to parametrize their poses as shown in Figure 1.

The cab and chassis are represented by oriented cuboids, with their poses parameterized by 3D translations, rotation matrices, and sizes. These pose parameters can be simplified using the excavator's kinematic constraints. The rotation matrix of the cab is identical to the $\mathbf{R}$ of NCS. Given the cab's rotation matrix, the rotation of the chassis is defined by a relative rotation angle θ about the main rotational axis with respect to the cab. The 3D dimensions of the cab and chassis are denoted as (d_{4x}, d_{4y}, d_{4z}) and (d_{5x}, d_{5y}, d_{5z}), respectively. Their translations can be calculated relative to the NCS. In this system, the (x, y, z) coordinates of the cab and chassis are $(l_{4x}, l_{4y}, d_{4z}/2)$ and $(0, 0, -d_{5z}/2)$, respectively. Since the vertical dimensions d_{4z} and d_{5z} are already considered in the size parameters, the translations can be reduced to two independent variables, l_{4x} and l_{4y}.

The three articulated arms are represented by three links (line segments) defined by four key points (K_1, K_2, K_3, K_4): the parent joint of the boom, the boom-stick joint, the stick-bucket joint, and the endpoint of the bucket, shown in Figure 1. The lengths of the boom, stick, and bucket links are denoted as l_1, l_2, and l_3, respectively. Using these key points, the lengths and angles of each arm can be computed, so the four key points are sufficient to parameterize the arm's pose. Furthermore, given the bucket's importance as an end effector, depth d_{3x} and width d_{3y} are introduced to consider its shape.

3.1.2 | Problem statement

Based on these definitions, this study formulates excavator pose estimation as follows: The input is a 3D LiDAR point cloud $P_L = \{p_i \in \mathbb{R}^3 | i = 1, \ldots, N\}$ containing the excavator in the LCS. The objective is to estimate the excavator's full body pose in LCS, which includes its global rotation $\mathbf{R} \in \mathbb{R}^{3\times3}$, translation $\mathbf{T} \in \mathbb{R}^3$, the coordinates of the four key points $\{K_k \in \mathbb{R}^3 | k = 1, 2, 3, 4\}$, the cab-to-chassis rotation angle θ, and the excavator's size parameters $S = \{l_{4x}, l_{4y}, d_{3x}, d_{3y}, d_{4x}, d_{4y}, d_{4z}, d_{5x}, d_{5y}, d_{5z}\}$.

First, the points of the excavator are detected and extracted from the scene point cloud by an existing 3D object detector. The rotation $\mathbf{R}_{LO}$ and translation $\mathbf{T}_{LO}$ of the 3D bounding box are used to define the object coordinate system (OCS). To handle excavators at arbitrary positions within the scene, the points within the 3D bounding box are extracted and then transformed from the LCS to the OCS via Equation (1), yielding P_O. Consequently, to obtain the pose in the LCS, it only needs to estimate the

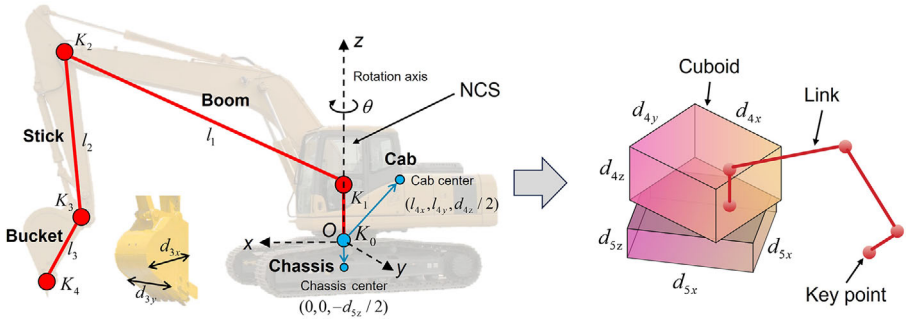

FIGURE 1 Excavator pose parametrization (left) and geometric primitive pose representation (right).

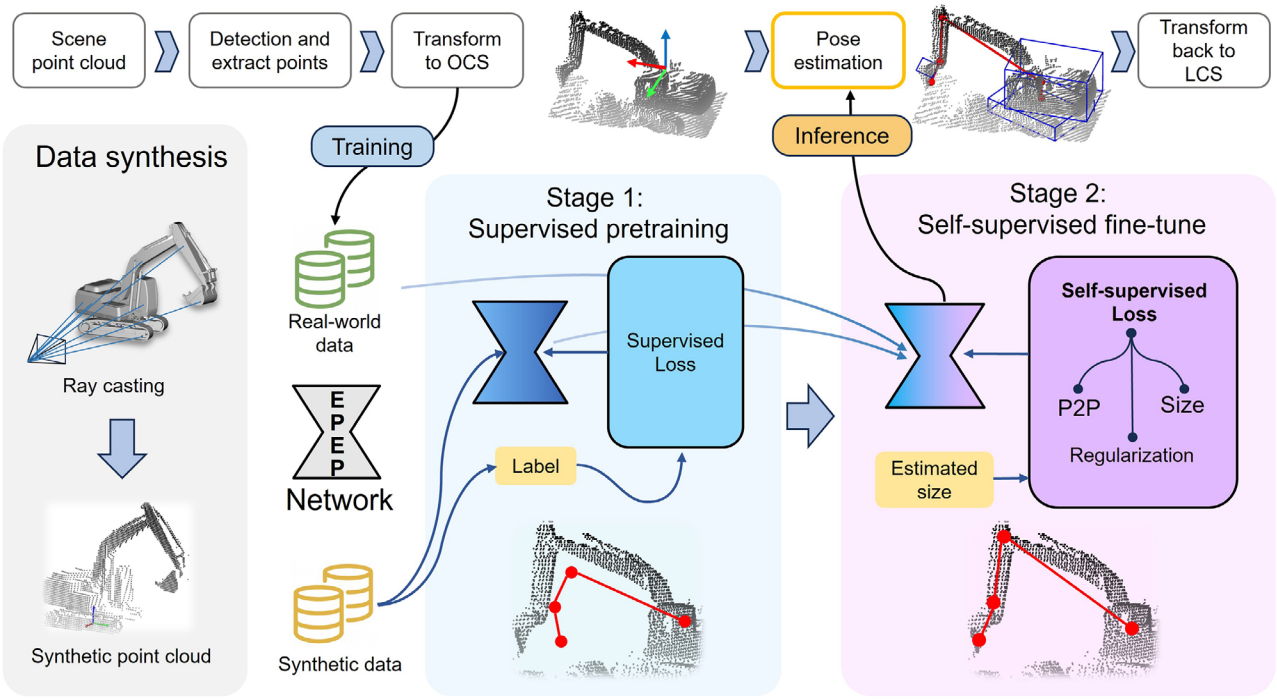

FIGURE 2 Method overview. EPEP, excavator pose estimation from point cloud; LCS, light detection and ranging (LiDAR) coordinate system; OCS, object coordinate system; P2P, point-to-primitive.

excavator's rotation $\mathbf{R}_{\mathrm{ON}}$ and translation $\mathbf{T}_{\mathrm{ON}}$ with respect to the OCS. For simplicity, the subscripts are omitted in the subsequent discussion. P is used to denote P_{O}, and $\mathbf{R}$ and $\mathbf{T}$ are used to represent $\mathbf{R}_{\mathrm{ON}}$ and $\mathbf{T}_{\mathrm{ON}}$, respectively.

$$P_{\mathrm{O}} = \mathbf{R}_{\mathrm{LO}}^{-1}(P_{\mathrm{L}} - \mathbf{T}_{\mathrm{LO}}) \tag{1}$$

3.2 | Method overview

The framework overview is illustrated in Figure 2. A deep network, termed the excavator pose estimation from point cloud (EPEP) network, takes the point cloud containing an excavator as input and then outputs rotation, translation, key point coordinates, and size parameters. To enable pose estimation without relying on manual 3D annotations, a two-stage training framework is employed that includes (1) supervised pretraining and (2) self-supervised fine-tuning. In the first stage, the model is trained on synthetic point clouds, with all predicted variables supervised by ground truth. This stage ensures effective network initialization and enables the model to learn the correlation between excavator poses and the structures of point clouds. However, discrepancies in geometry, noise, and environment between synthetic and real-world data lead to a performance gap when the pre-trained model is applied to real-world scenarios. To bridge this gap, the second stage trains the model on unlabeled real-world data using self-supervised loss functions. This stage aims to align the real-world distribution by enforcing consistency between the estimated poses and the observed point cloud, as well as consistency with prior knowledge of geometry. The subsequent sections provide a detailed description of the EPEP network, followed by an in-depth explanation of the loss functions in the two training stages.

3.3 | Pose estimation network

Figure 3 illustrates the EPEP network. Its backbone adopts a U-Net architecture, consisting of an encoder followed by a decoder. The encoder includes four stages, which progressively downsample the point cloud while extracting point features. The decoder mirrors the encoder with four symmetric stages that progressively upsample the point cloud. It uses skip connections to transfer fine-grained spatial information from early layers in encoder to the corresponding decoder, where features are merged via element-wise addition. Each stage incorporates multiple point transformer layers (Wu et al., 2022; Zhao et al., 2021), featuring grouped vector attention that applies vector self-attention to the k-nearest neighbors of each point. Following the best practice, downsampling between stages is achieved through grid-based pooling (Wu et al., 2022).

Excavator pose estimation involves predicting many variables distinct from each other, which requires delicate regression strategies. A multi-task output head is built on top of the backbone network, which yields outputs at

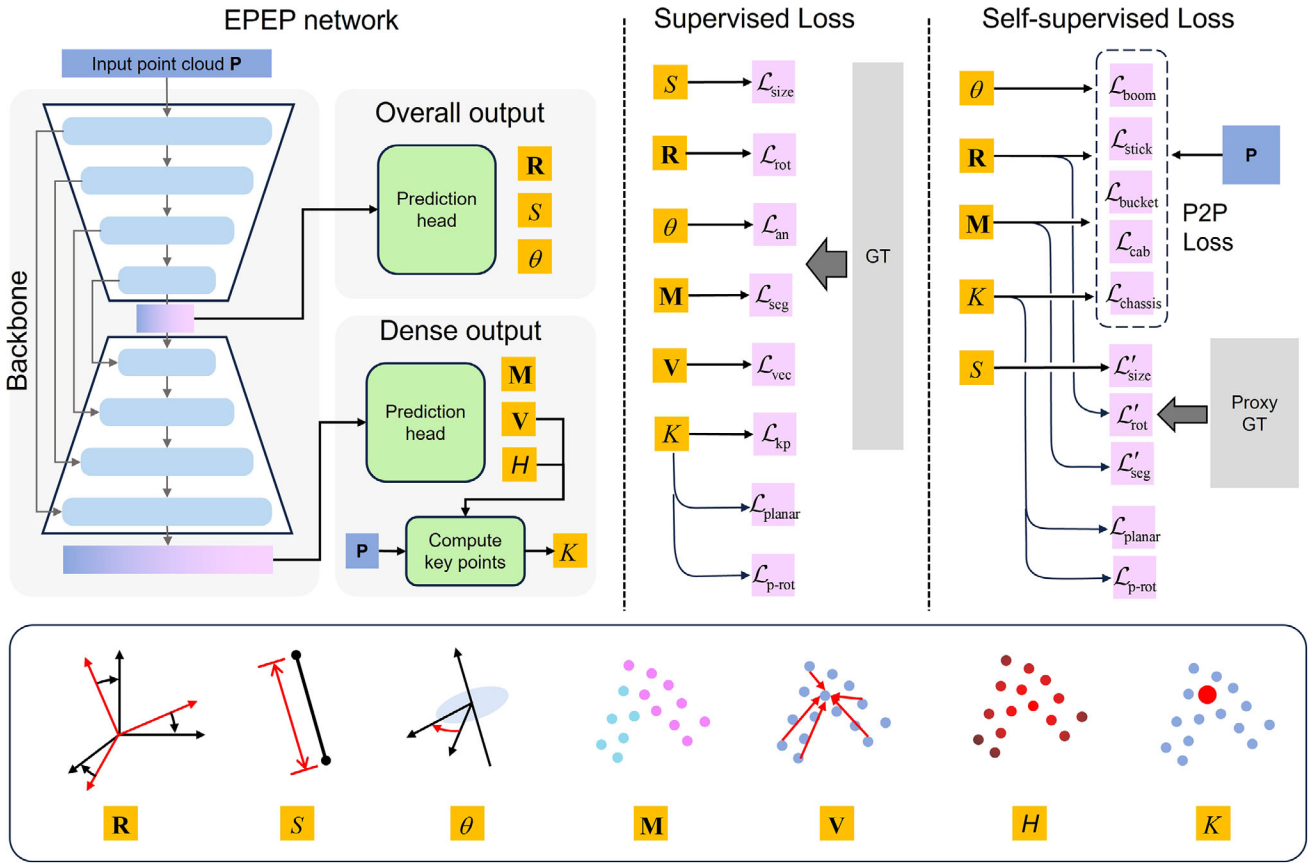

FIGURE 3 Excavator pose estimation from point cloud (EPEP) network and the loss functions in the training framework. At the bottom illustrates the output variables. GT, ground truth. P2P, point-to-primitive.

two levels: (1) Encoder-level overall output: This includes the direct predictions following the encoder, such as the rotation matrix **R**, angle θ, and size parameters S. For each head, a single multi-layer perceptron (MLP) is first applied, followed by parallel global average and max pooling. Finally, these features were concatenated along the channel dimension and projected to the output dimension through linear projection. (2) Decoder-level dense output: This generates point-wise predictions, including point cloud segmentation logits **M**, as well as heatmaps H and vectors **V** for key points regression. For each head, a sequence of an MLP and a linear projection are employed to obtain the output.

The regression of the rotation matrix leverages the 6D continuous rotation representation (Zhou et al., 2019), which predicts only the first two column vectors of the 3×3 rotation matrix. During inference, the predicted vectors are converted into a valid rotation matrix using the following equation:

$$\hat{\mathbf{R}} = [\mathrm{N}(\mathbf{a}_1), \mathrm{N}(\mathbf{a}_1) \times \mathrm{N}(\mathbf{a}_2), \mathrm{N}(\mathbf{a}_1) \times (\mathrm{N}(\mathbf{a}_1) \times \mathrm{N}(\mathbf{a}_2))] \quad (2)$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ denote the predicted first and second column vectors, respectively, and $\mathrm{N}(\bullet)$ denotes the normalization function to ensure orthonormality. For the angle θ, the network regresses two variables, $\cos(\theta)$ and $\sin(\theta)$. During inference, the angle is computed using the inverse trigonometric function.

Directly regressing key point coordinates from point cloud inputs is highly nonlinear (Ge et al., 2018). A point-wise ensemble approach is adopted to regress key points rather than direct regression. The dense head of the model predicts a point-wise vector $\hat{\mathbf{V}}_{ik} \in \mathbb{R}^3$, representing the offset from point p_i to key point K_k. In addition, the dense head predicts a point-wise scalar heatmap $\hat{H}_{ik}$, which serves as the weight to aggregate the dense vector prediction. A key point $\hat{K}_k$ is calculated from vectors $\hat{\mathbf{V}}_{ik}$ and points p_i using weighted average:

$$\hat{K}_k = \frac{\sum_{i=1}^{N} (\hat{\mathbf{V}}_{ik} + p_i) \cdot \sigma(\hat{H}_{ik})}{\sum_{i=1}^{N} \sigma(\hat{H}_{ik})} \quad (3)$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function that maps $\hat{H}_{ik}$ into range (0,1). As the translation **T** is inherently a 3D coordinate, it is treated as an additional key point (denoted as K_0) and predicted using this approach.

While point cloud segmentation is not directly required for pose estimation, it plays a crucial role in the

subsequent self-supervised method. Since there possibly be background points in the input, identifying them also benefits the pose estimation, which also requires point cloud segmentation. Therefore, the dense head additionally outputs the point-wise membership $\hat{\mathbf{M}}_i \in \mathbb{R}^6$, which represents the probability of each point belonging to one of six categories, including five excavator components and the background.

3.4 | Supervised pretraining

At this stage, the model is trained in a fully supervised manner using synthetic data. As the ground truth is obtained with synthetic data, no manual labeling is required. This provides sufficient supervision for the smooth initialization of the EPEP network. The subsequent sections will describe the loss functions, which are also shown in Figure 3.

Mean-square loss $\mathcal{L}_{kp}$ is employed to supervise key points:

$$\mathcal{L}_{\text{kp}} = \sum_{k} \|\hat{K}_k - K_k\|_2^2 \tag{4}$$

Since the computation of $\hat{K}_k$ in Equation (3) is differentiable, it enables end-to-end training through Equation (4). The model can learn appropriate heatmap values during training, so the heatmap serves as an intermediate variable, which does not require a loss function. Furthermore, supervising the point-wise vector with ground truth enhances the stability and efficiency of training, allowing quicker convergence (Y. Weng et al., 2021). So, the loss of point-wise vector is adopted, given by:

$$\mathcal{L}_{\text{vec}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=0}^{4} \|\hat{\mathbf{V}}_{ik} - \mathbf{V}_{ik}\|_2^2 \tag{5}$$

The following variables are also trained with mean-square loss: the rotation loss $\mathcal{L}_{rot}$ for 6D rotation parameters, the angle loss $\mathcal{L}_{an}$ for $\{\cos(\theta), \sin(\theta)\}$, and the size loss $\mathcal{L}_{size}$ for all the size parameters S. The segmentation loss is given by cross-entropy:

$$\mathcal{L}_{\text{seg}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{6} \mathbf{M}_{i,j} \log\left(\hat{\mathbf{M}}_{i,j}\right) \tag{6}$$

where $\hat{\mathbf{M}}_{i,j}$ is the predicted probability for the jth class of the ith point, and $\mathbf{M}_i$ is the one-hot ground truth label for ith point.

In addition, geometry constraint-based losses are introduced as regularization terms to encourage the predictions to align the excavator kinematics. Considering the arm plane constraint, the planarity loss $\mathcal{L}_{\text{planar}}$ is introduced to encourage the four key points to lie on the same plane. Specifically, each key point is treated as a test point iteratively, and its distance to the plane defined by the other three key points is computed as given by:

$$\mathcal{L}_{\text{planar}} = \frac{1}{4} \sum_{k=1}^{4} \frac{\|((\hat{K}_b - \hat{K}_a) \times (\hat{K}_c - \hat{K}_a)) \cdot (\hat{K}_k - \hat{K}_a)\|}{\|(\hat{K}_b - \hat{K}_a) \times (\hat{K}_c - \hat{K}_a)\|} \tag{7}$$

where $\hat{K}_k$ is the test point indexed by k and $(\hat{K}_a, \hat{K}_b, \hat{K}_c)$ are the other three points. The cross product calculates the normal vector of the plane and the dot product calculates the projected length, that is, the distance to the plane.

The planarity loss alone is not enough, as it does not guarantee that the plane's orientation aligns with the rotation matrix $\hat{\mathbf{R}}$. Therefore, the plane-rotation loss $\mathcal{L}_{\text{p-rot}}$ is introduced to encourage the four key points to lie on the x-z plane defined by $\hat{\mathbf{R}}$, given by:

$$\mathcal{L}_{\text{p-rot}} = \frac{1}{3} \sum_{k=1}^{3} \frac{\left\|\mathbf{e}_y^R \cdot \left(\hat{K}_{k+1} - \hat{K}_k\right)\right\|}{\left\|\mathbf{e}_y^R\right\|} \tag{8}$$

where $\mathbf{e}_y^R$ is the second column vector of $\hat{\mathbf{R}}$, which corresponds to the y-axis vector of the NCS. This loss computes the projection length of the vectors defined by two adjacent key points onto $\mathbf{e}_y^R$.

The overall training objective for this stage is to minimize the total loss:

$$\begin{aligned}\mathcal{L}_1 = {} & \lambda_{\text{kp}}\mathcal{L}_{\text{kp}} + \lambda_{\text{rot}}\mathcal{L}_{\text{rot}} + \lambda_{\text{an}}\mathcal{L}_{\text{an}} + \lambda_{\text{size}}\mathcal{L}_{\text{size}} + \lambda_{\text{vec}}\mathcal{L}_{\text{vec}} \\ & + \lambda_{\text{seg}}\mathcal{L}_{\text{seg}} + \lambda_{\text{planar}}\mathcal{L}_{\text{planar}} + \lambda_{\text{p-rot}}\mathcal{L}_{\text{p-rot}}\end{aligned} \tag{9}$$

3.5 | Self-supervised fine-tuning

In the absence of pose ground truth, it is essential to design self-supervised loss functions to optimize the network effectively. Excavator pose estimation exhibits the following characteristics: (1) The estimated pose should ensure that the geometric primitives of excavator components closely align with the point cloud. For example, the link representing the stick should not deviate from the point cloud. (2) The dimensions (e.g., length of articulated arms) of the same excavator remain constant over time. The first characteristic is the core principle, as it provides an essential learning objective for pose estimation, supervised by the point cloud itself. The second is also indispensable, as fixed dimensions eliminate potential ambiguities in pose estimation. By integrating these two principles, self-supervised losses can be designed to

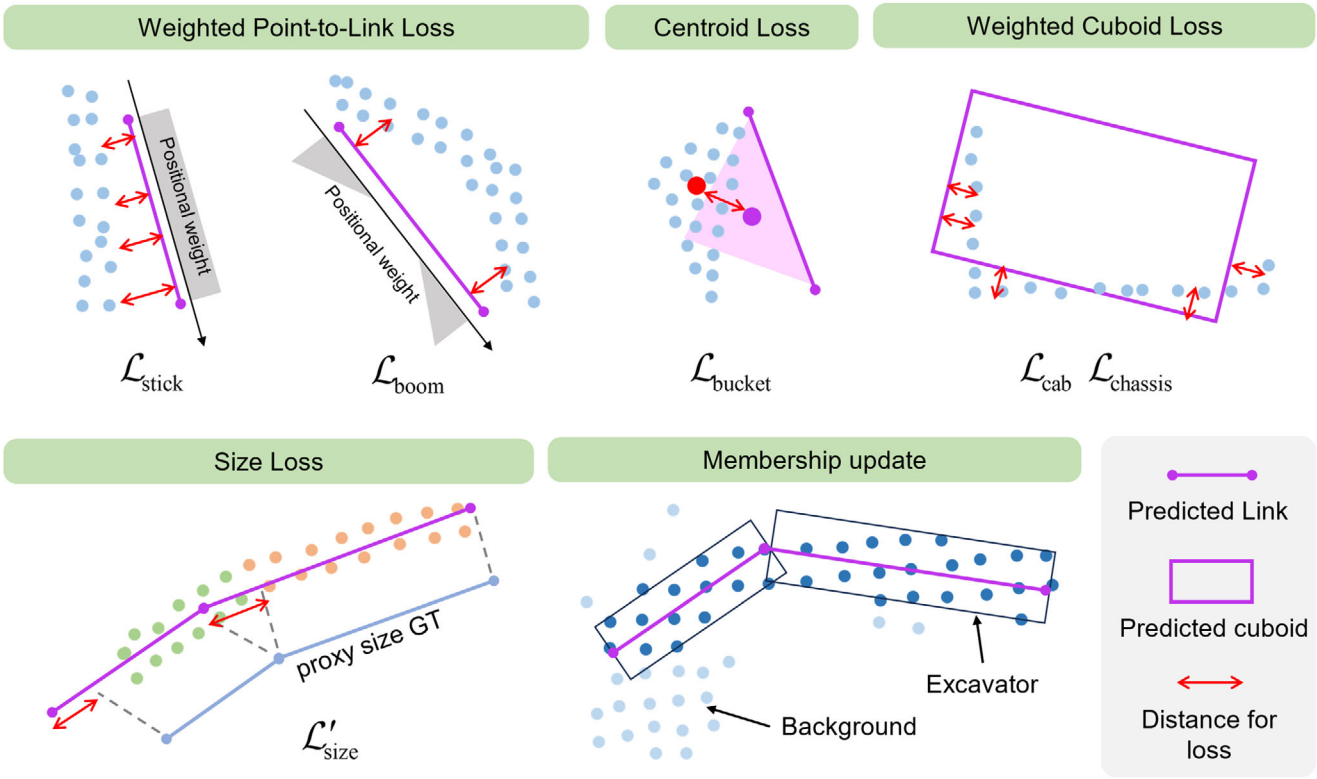

FIGURE 4 Self-supervised loss. The first row illustrates the P2P loss (Section 3.5.1). The second row illustrates the size loss (Section 3.5.2) and membership update (Section 3.5.3). GT, ground truth.

guide the convergence of pose estimation toward the correct direction. These loss functions are summarized in Figure 3 and detailed in Figure 4.

3.5.1 | Point-to-primitive (P2P) loss

In Section 3.1, the poses of the excavator's five components are represented by geometric primitives. Here, these primitives are further refined into links, triangles, and cuboids. The key insight underlying this loss is that the points of the excavator should lie near or be distributed around these primitives. We propose the P2P loss $\mathcal{L}_{p2p}$, which penalizes the distances between points and their corresponding geometric primitives. To make the loss component-aware, the estimated membership $\hat{\mathbf{M}}$ is used to identify points belonging to a specific component, ensuring that loss is only calculated between a component and the points assigned to it. Optimizing $\hat{\mathbf{M}}$ and the distances simultaneously can lead to instability during training, so $\hat{\mathbf{M}}$ is detached from the computational graph during training and excluded from backpropagation.

Weighted point-to-link loss

This loss is designed for boom and stick that are represented by links. It measures the average distance between a set of points and a link, incorporating positional weighting to emphasize specific points along the link. Let $\hat{K}_a$ and $\hat{K}_b$ denote the estimated start and end key points of a link, respectively. The distance from point p_i to the link is computed using the cross product:

$$d_i = \mathrm{d}(p_i, \hat{K}_a, \hat{K}_b) = \frac{\|(p_i - \hat{K}_a) \times (\hat{K}_b - \hat{K}_a)\|}{\|\hat{K}_b - \hat{K}_a\|} \tag{10}$$

Although excavator arms have been represented by links, their actual geometry may deviate from straight lines; for example, the boom often has a curved shape. As a result, it is not necessary for the link to stay close to the point cloud at every position. Therefore, a positional weight $w(t)$ is introduced for this loss based on the relative projection t of a point p_i onto the line. The projection t is computed by:

$$t_i = \frac{(p_i - K_a) \cdot (K_b - K_a)}{\|K_b - K_a\|^2} \tag{11}$$

The positional weight determines how much influence a point has on the loss. Sticks use uniform weight function:

$$w(t) = \begin{cases} 1, & \text{if } 0 \le \mathrm{t} \le 1, \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Booms use a two-ends weight function, emphasizing the points near the start and end of the line, and diminishing toward the middle. A parameter η is used to define the influence region. The function is given by:

$$w(t) = \begin{cases} (\eta - t)^2/\eta^2, & \text{if } 0 \le t \le \eta, \\ (t - (1-\eta))^2/\eta^2, & \text{if } 1-\eta \le t \le 1, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

The overall weighted point-to-link loss for component j is computed as the average of the distances, weighted by the product of $w(t)$ and point-wise membership $\hat{\mathbf{M}}$:

$$\mathcal{L}_{\text{p2p}}^{\text{link}}(j) = \frac{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,j} w(t_i) d_i}{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,j} w(t_i)} \tag{14}$$

Centroid loss

This loss is specifically designed for the bucket, whose shape resembles a triangle (when viewed from the side). The objective is to ensure that the estimated centroid of the bucket is close to the centroid of its associated points. In this setup, the line segment connecting K_3 and K_4 is treated as the base of the triangle, while the triangle's height is represented by the size parameter d_{3x}. Although the position of the centroid varies for different triangles, the distance from the centroid to the base remains constant at $2d_{3x}/3$. Therefore, instead of explicitly calculating the centroid's position, the centroid-to-base distance is used as the loss function:

$$c_3 = \frac{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,3} p_i}{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,3}} \tag{15}$$

$$\mathcal{L}_{\text{p2p}}^{\text{centroid}} = \left\| \frac{2}{3} d_{3x} - \text{d}(c_3, \hat{K}_3, \hat{K}_4) \right\| \tag{16}$$

where $\hat{\mathbf{M}}_{i,3}$ denotes the points' membership for the bucket, c_3 is the centroid of bucket's points, and $\text{d}(\cdot)$ is the point-to-link distance calculated using Equation (10).

Weighted cuboid loss

This loss is designed for the cab and chassis that are represented as cuboids. The goal is to minimize the distance between the estimated cuboid and the associated points, while balancing the influence of points in different regions using weights. It begins by transforming the excavator's point cloud into the NCS:

$$p_i^{\text{N}} = \mathbf{R}^{-1}(p_i - \mathbf{T}) \tag{17}$$

Based on the positions of the components in the NCS (as described in Section 3.1), the points are then further transformed into the cuboid-centric coordinate system. The distances between the normalized points p_i^{N} and the surfaces of the cuboid are computed as follows:

$$d_i^{\text{in}} = \max\left(\min\left(1 - \frac{2\,|x_i|}{d_x}, 1 - \frac{2\,|y_i|}{d_y}, 1 - \frac{2\,|z_i|}{d_z}\right), 0\right) \tag{18}$$

$$d_i^{\text{ex}} = \|(\max(0, x_i - d_x/2), \max(0, y_i - d_y/2), \max(0, z_i - d_z/2))\|_2 \tag{19}$$

$$\mathcal{L}_{\text{p2p}}^{\text{cuboid}}(j) = \frac{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,j}(d_i^{\text{in}} + \mu d_i^{\text{ex}})}{\sum_{i=1}^{N} \hat{\mathbf{M}}_{i,j}} \tag{20}$$

where $(x_{\text{i}}, y_{\text{i}}, z_{\text{i}})$ are the 3D coordinates of p_i^{N}, $(d_{\text{x}}, d_{\text{y}}, d_{\text{z}})$ are the sizes of the cuboid, d_i^{in} denotes the interior distance, and d_i^{ex} denotes the exterior distance. Since the cab and chassis are not perfect cuboids and the estimated cuboid is expected to fully enclose the point cloud of the component, points are allowed to lie within the cuboid. However, points belonging to the component should not fall outside the cuboid. Consequently, the loss allows for a certain tolerance of d_i^{in}, while enforcing d_i^{ex} to approach zero. To balance the supervision of d_i^{in} and d_i^{ex}, a parameter μ is introduced, setting $\mu = 5$.

3.5.2 | Size loss

Using only the P2P loss is insufficient to ensure accurate pose estimation. As illustrated in Figure 4, despite the small P2P loss, there is a considerable error in the link size, and the estimated pose has a large error. Therefore, incorporating the size loss can effectively address pose ambiguity in self-supervision.

Before self-supervised training, the network trained in the first stage is used to estimate the precise size parameters $\bar{S}$ for each training sample. These sizes are used as ground truth to supervise size estimation in this stage. Here, $\bar{S}$ includes not only all the previous size parameters but also the lengths of the three articulated arms $(\hat{l}_1, \hat{l}_2, \hat{l}_3)$, which are the Euclidean distances between their respective pairs of estimated key points. To minimize errors of $\bar{S}$ and ensure consistent size for the same excavator, it post-processes the sizes for all samples belonging to the same excavator. An interquartile range (IQR) filter is applied to detect and remove outliers (Vinutha et al., 2018), followed by averaging the results to obtain $\bar{S}$. Using the IQR filter reduces sensitivity to extreme estimation errors, resulting in more accurate and robust averaged size estimations. The mean-square error between the predicted size $\hat{S}$ and the target size $\bar{S}$ is employed to compute the size loss $\mathcal{L}'_{\text{size}}$.

3.5.3 | Regularization and sum-up

The effectiveness of P2P loss for self-supervised training relies heavily on the accuracy of the part segmentation predictions $\hat{\mathbf{M}}$. Errors in the predicted membership, such as the frequent misclassification of background points as belonging to the bucket, result in inaccurate supervision signals. Additionally, relying solely on the P2P loss can be insufficient and even lead to degenerate solution. For example, the model might trivially minimize P2P loss by predicting all points as background, or by converging to incorrect poses that happen to loosely align with the point cloud but are geometrically or kinematically implausible. To solve this problem, additional regularization techniques are proposed to constrain and guide the optimization.

Our solution is to include the rotation matrix loss $\mathcal{L}'_{\text{rot}}$ and the part segmentation loss $\mathcal{L}'_{\text{seg}}$ as regularization terms in this stage, using the same formulas as in the first stage. They should be treated as soft constraints, rather than strict ground truth supervision, guiding the self-supervised learning process in a more stable and meaningful direction. The key lies in how to obtain the targets for supervision. The predictions from the pre-trained model are used as proxy ground truth for rotation and segmentation in this stage. For $\mathcal{L}'_{\text{rot}}$, the proxy ground truth encourages the model in the fine-tuning stage to maintain a stable global orientation, preventing drastic shifts in rotation.

For $\mathcal{L}'_{\text{seg}}$, before proxy membership targets can serve as ground truth, they require an update to correct the inconsistency with the predicted pose. Specifically, for each excavator component, a 3D bounding box is constructed based on the predicted pose as illustrated in Figure 4. For the three arms, the bounding box is aligned along the axis of the link, while for the chassis and cab, the bounding box corresponds to their cuboid. The membership update rule is that points falling outside these bounding boxes should belong to the background.

Additionally, this stage maintains the geometric constraint losses $\mathcal{L}_{\text{planar}}$ and $\mathcal{L}_{\text{p-rot}}$ from the first stage. Based on these definitions, the total loss function for the self-supervised training stage is:

$$\begin{aligned}\mathcal{L}_2 = {} & \lambda_{\text{p2p}}\mathcal{L}_{\text{p2p}} + \lambda'_{\text{rot}}\mathcal{L}'_{\text{rot}} + \lambda'_{\text{size}}\mathcal{L}'_{\text{size}} + \lambda'_{\text{seg}}\mathcal{L}'_{\text{seg}} \\ & + \lambda_{\text{planar}}\mathcal{L}_{\text{planar}} + \lambda_{\text{p-rot}}\mathcal{L}_{\text{p-rot}} \end{aligned} \tag{21}$$

where $\mathcal{L}_{\text{p2p}}$ is the P2P loss composed of five components:

$$\begin{aligned}\mathcal{L}_{\text{p2p}} &= \mathcal{L}_{\text{boom}} + \mathcal{L}_{\text{stick}} + \mathcal{L}_{\text{bucket}} + \mathcal{L}_{\text{cab}} + \mathcal{L}_{\text{chassis}} \\ &= \mathcal{L}^{\text{link}}_{\text{p2p}}(1) + \mathcal{L}^{\text{link}}_{\text{p2p}}(2) + \mathcal{L}^{\text{centroid}}_{\text{p2p}} \\ &\quad + \mathcal{L}^{\text{cuboid}}_{\text{p2p}}(4) + \mathcal{L}^{\text{cuboid}}_{\text{p2p}}(5) \end{aligned} \tag{22}$$

4 | DATASET

4.1 | Synthetic data for posed excavators

A synthetic dataset of excavators was created to pre-train the pose estimation network under full supervision. The 3D models of the excavator's five components were first created and then were assembled and randomly scaled, rotated, and translated to enhance the diversity. The Open3D library (Open3D, 2024) was used to generate point clouds via ray casting. A virtual LiDAR for casting rays was randomly placed within a distance range of 10–50 m, a height range of 1.5–6 m, and with random angular orientations. This setup synthesized point clouds from diverse distances, angles, and perspectives. To simulate real-world conditions, the point clouds were further augmented by adding random Gaussian noise and environmental elements such as ground surfaces and clutter. For each synthetic point cloud, its pose labels, including all the output variables described in Section 3.1, were computed and saved. In total, 16,800 synthetic excavator point cloud samples were generated and split into 12,000 for training, 3200 for validation, and 1600 for testing.

4.2 | Real-world data

A real-world excavator point cloud dataset was created to fine-tune and evaluate the pose estimation model. The RS-M1 LiDAR (Robosense, 2023), identical to the model used in (Zhang et al., 2024), was used for data collection. This LiDAR features a 120-degree horizontal field of view, an angular resolution of 0.2 degrees, and a working frame rate of 10 Hz. Data were collected at multiple excavation sites where various models of excavators (e.g., Komatsu, XCMG, SANY, Kobelco, and CAT) were performing diverse tasks, including ground leveling, surface digging, slope excavation, material transportation, and truck loading, as shown in Figure 5. To capture a variety of data, the LiDAR was frequently repositioned to scan excavators from different distances and angles. A total of 2810 point cloud samples were collected, which were split into 2020 samples for training and 790 samples for testing. While the training samples do not require annotations, the test samples were manually annotated. An open-source annotation tool (E. Li et al., 2020) was used to label the excavator pose, which includes the four key points and oriented bounding boxes of cabs and chassis. Figure 6 illustrates the distribution of distances and view angles between the LiDAR and the excavators in the real-world dataset, showing a diverse range.

FIGURE 5 Photos of excavators during data collection.

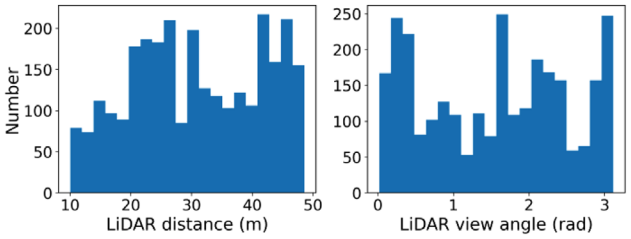

FIGURE 6 Distribution of excavator samples in real-world data. LiDAR, light detection and ranging.

5 | EXPERIMENTS AND RESULTS

5.1 | Evaluation metrics

The following evaluation metrics are employed for excavator pose estimation:

1. Mean per joint position error (MPJPE): This metric quantifies the deviation of the predicted key point $\hat{K}_k$ from the ground truth using the Euclidean distance. Lower values indicate more accurate predictions. In this study, MPJPE is reported in meters.
2. Joint position accuracy (JPA): This metric represents the percentage of key points with prediction errors smaller than 0.3 m. Unlike MPJPE, which captures errors of any magnitude, JPA evaluates whether the error falls within an acceptable range.
3. 3D intersection-over-union (IoU): This metric computes the IoU between the predicted and ground truth 3D bounding boxes. It primarily evaluates the accuracy of the position and size for cabs and chassis. A value of 1 indicates a perfectly accurate prediction.
4. Angle error E_{A}: This metric calculates the absolute error of the predicted cab-to-chassis rotation angle θ.
5. Rotation error E_{R}: This metric converts the predicted rotation matrix into three rotation angles (x, y, and z axes) and computes the absolute error for each axis. Both angle error and rotation error are reported in degrees.

5.2 | Implementation details

An existing 3D object detector (Zhang et al., 2024) was used to perform 3D object detection for excavators. Most of the configurations are consistent with Zhang et al. (2024), except for using a larger initial voxel size of 0.3 m to accommodate the excavator scenes. Since the primary focus of this study is pose estimation, the object detection evaluation and its impact on pose estimation fall outside the scope of this work.

TABLE 1 The loss weights in pretraining.

Loss	$\mathcal{L}_{\mathrm{kp}}$	$\mathcal{L}_{\mathrm{rot}}$	$\mathcal{L}_{\mathrm{an}}$	$\mathcal{L}_{\mathrm{size}}$	$\mathcal{L}_{\mathrm{vec}}$	$\mathcal{L}_{\mathrm{seg}}$	$\mathcal{L}_{\mathrm{planar}}$	$\mathcal{L}_{\mathrm{p-rot}}$
Weight	2	5	2	0.5	0.5	1	0.1	0.2

The hyperparameters of the EPEP network were determined based on the practices in related networks (Wu et al., 2022) and performance observations on the validation set. The encoder and decoder stages of the network are configured with Point Transformer layers in numbers of [2, 2, 6, 2] for the encoder and [1, 1, 1, 1] for the decoder. The initial feature dimension is set to 32, which is embedded from the input feature dimension (i.e., 3) through a Point Transformer layer. Subsequently, the feature dimensions for each encoder stage are set to [64, 128, 256, 256], with the decoder's feature dimensions designed symmetrically. The grid sizes for each stage are set to [0.4, 0.8, 1.6, 3.2 m]. The number of nearest neighbors for the self-attention is set to 16.

The model was implemented using PyTorch (Paszke et al., 2016/2019). All the training and inference were conducted on an NVIDIA 4090 graphics card. In the first stage, the model was trained from scratch on the training set of the synthetic dataset. The training process spanned 150 epochs with a batch size of 32, requiring approximately 2 h for a complete session. The Adam optimizer was employed with a weight decay of 0.001. To facilitate model convergence, the 1cycle learning rate scheduling strategy (Smith & Topin, 2018) was adopted, with the initial learning rate, maximum learning rate, and final divided factor set to 0.0001, 0.001, and 1000, respectively. The weights of each loss term, determined through experimental tuning, are provided in Table 1.

In the second stage, the model was trained on the real-world dataset and initialized using weights from the first stage. In addition, each sample was applied random transformation and rotation to simulate inaccuracy from object detection. The model was trained for 50 epochs with an initial learning rate of 0.0001, using an exponential learning rate scheduler with a gamma value of 0.9. All other parameter settings remain consistent with the first

TABLE 2 The loss weights in self-supervised fine-tuning.

Loss	$\mathcal{L}_{p\text{-}p}$	$\mathcal{L}'_{rot}$	$\mathcal{L}'_{size}$	$\mathcal{L}'_{seg}$	$\mathcal{L}_{planar}$	$\mathcal{L}_{p-rot}$
Weight	1	2	5	1	0.1	0.2

TABLE 3 Key points localization results.

Metrics	K_0 (translation)	K_1	K_2	K_3	K_4	Overall
Mean per joint position error (MPJPE; m)	0.17	0.19	0.24	0.26	0.42	0.26
Joint position accuracy (JPA; %)	99.2	93.7	84.0	81.5	56.6	83.0

stage. The weights of each loss term, determined through experimental tuning, are provided in Table 2

5.3 | Results

5.3.1 | Quantitative results

Overall accuracy

The localization accuracy for individual and overall key points is presented in Table 3, including MPJPE and JPA. The whole translation is denoted by K_0. The results show that the proposed method achieves an overall MPJPE of 0.26 m. The translation error of excavators is remarkably small—only 0.17 m—showing the method's precise localization capabilities. The accuracy varies across different components. The parent joint of the boom (K_1), the boom-stick joint (K_2), and the stick-bucket joint (K_3) exhibit MPJPE close to the average value, whereas the MPJPE for the bucket end (K_4) is slightly higher. This is primarily due to the bucket end, as the end effector, frequently interacting with objects (e.g., soil, rocks, and materials), which can alter the point cloud's shape or cause missing points. Other pose estimation metrics are presented in Table 4. JPA indicates that more than 80% of the key points have errors less than 0.3 m. The 3D IoU and E_A metrics demonstrate that the proposed method achieves decent size and position estimation for the cab and chassis. The E_R metric with approximately 3 degrees indicates excellent 3D rotation estimation performance.

Inference speed

The computational efficiency of the proposed method was tested, achieving a runtime of 30 ms (33 fps) on an NVIDIA RTX 3070 graphics card. Given that most modern LiDAR systems operate at a frame rate of 10–20 fps, this demonstrates that the method preliminarily satisfies the real-time performance requirements for engineering applications.

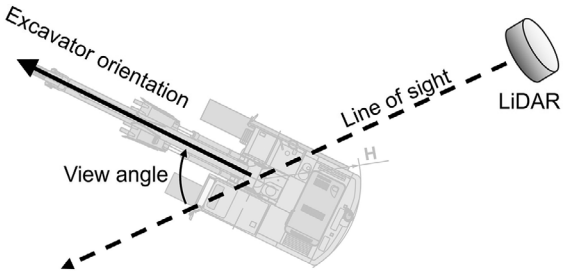

FIGURE 7 Definition of view angle. LiDAR, light detection and ranging.

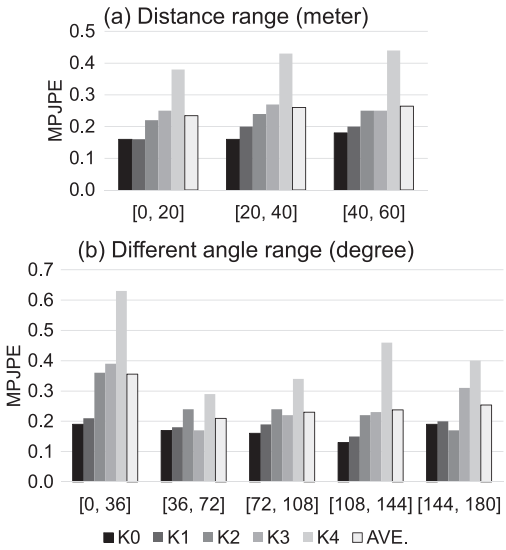

FIGURE 8 Mean per joint position error (MPJPE) in different distance and angle range.

View angles and distances

The method was also evaluated across various view angles and distances. The definition of view angles is illustrated in Figure 7. Changes in view angle affect the shape of the point cloud and the level of self-occlusion, while distance impacts the sparsity of the point cloud. View angles were divided into five bins, ranging from 0 to 180 degrees in 36-degree intervals. Distances were divided into three bins: 0–20 m, 20–40 m, and 40–60 m. As shown in Figure 8, The localization error remains largely unaffected by increasing distance, maintaining consistent performance within the 0–60 m range. Translation errors are minimal across all view angle ranges. However, the errors at the endpoints of the three arms (K_2, K_3, K_4) increase significantly in the 0–36 degree range, primarily due to self-occlusion caused by the excavator's structure. Additional qualitative results on occlusion are provided in later sections. In summary, the results highlight the robustness of the model across varying distances and view angles.

Generalization across different excavators

Table 5 presents evaluation results across different excavator models. The variation in shape and size of the

TABLE 4 Results of excavator pose estimation and ablation studies.

Model/variation	MPJPE (m)↓	JPA (%)↑	3D intersection-over-union (IoU)↑		E_A (degree)↓	E_R (degree)↓		
			Cab	Chassis		x	y	z
Full model	0.26	83.0	0.75	0.68	6.3	2.3	1.7	2.3
W/o fine-tuning	0.36	71.9	0.69	0.51	9.4	3.4	2.8	3.2
W/o size loss	0.32	77.3	0.31	0.25	6.7	2.3	1.8	2.3
W/o seg loss	0.51	48.2	0.56	0.49	12.5	2.4	1.8	2.3

TABLE 5 Pose estimation results for different excavator models.

Model (size)	MPJPE (m)↓	JPA (%)↑	3D IoU↑		E_A (degree)↓
			Cab	Chassis	
Komatsu (small)	0.25	84.0	0.68	0.64	6.4
SANY (mid)	0.29	77.8	0.78	0.62	5.0
XCMG (large)	0.22	89.2	0.74	0.70	8.1

excavators provides a test for generalization capability. Three brands of excavators were selected: Komatsu, SANY, and XCMG, which are also in small, medium, and large sizes, respectively. The results demonstrate that the proposed method achieves satisfactory performance across all excavator models, showcasing good pose estimation ability that is not restricted to a specific type. This robustness is primarily due to the method's independence from fixed size priors. Unlike previous approaches that are constrained to specific models and sizes (Cho & Gai, 2014), the proposed network directly estimates size parameters, enabling it to accommodate various sizes of excavators.

Comparison with previous methods

Our method is compared with other non-contact approaches that share similar principles. Given the outdated nature of existing LiDAR-based methods, we focused on two recent camera-based 3D pose estimation methods for comparison. The first method (J. Kim et al., 2023) directly estimates 3D poses from 2D images, yielding an average error exceeding 1 m. The second method (Assadzadeh et al., 2023) refines 2D pose estimations into 3D, achieving an average error of 0.7 m; however, this method has not been tested in construction sites. In contrast, our method demonstrates a significant improvement in accuracy (0.26 m) when applied to real construction site scenarios, outperforming existing methods.

5.3.2 | Qualitative results

This section presents the qualitative results. Figure 9a illustrates the results under various view angles and distances, showing that the estimated poses align closely with the ground truth. Figure 9b provides additional examples, highlighting the robustness of the method in diverse scenarios. Notably, the proposed method demonstrates accurate predictions even in challenging conditions, such as severe occlusions, incomplete point cloud of the working arm, very sparse points on excavators, and interactions with other machinery.

Figure 10 illustrates several error cases in pose estimation. The first type of error involves incorrect localization of the excavator's stick and bucket. This typically occurs when the view angle is close to 0 degrees (with the excavator facing away from the LiDAR), resulting in the stick and bucket being occluded by the excavator itself. However, it is notable that these error cases are rare as shown in Figure 9b where only a small portion of points remain, correct pose estimation could still be achieved. The second type of error involves significant misorientation of the chassis, indicated by a large E_A. These errors are primarily caused by occlusion of the chassis, leaving insufficient points for accurate pose inference. Such errors are also commonly observed when the θ angle approaches 90 degrees.

The results described above utilize geometric primitives to abstractly represent the excavator, yet a more intuitive and realistic visualization is achievable. By scaling, rotating, and translating the 3D model of each excavator component based on predicted parameters, the excavator's full-body 3D model is generated as illustrated in Figure 11. This process achieves mapping from on-site point clouds to a 3D geometric model. Due to the precise and comprehensive pose parameters, the resulting 3D model aligns closely with the actual on-site dimensions and posture.

5.4 | Ablation studies

This section further explores the impact of self-supervised loss, with results listed in Table 4. The first experiment removes the self-supervised fine-tuning (w/o fine-tuning). The results demonstrate that the absence of fine-tuning obviously increases errors in pose estimation across

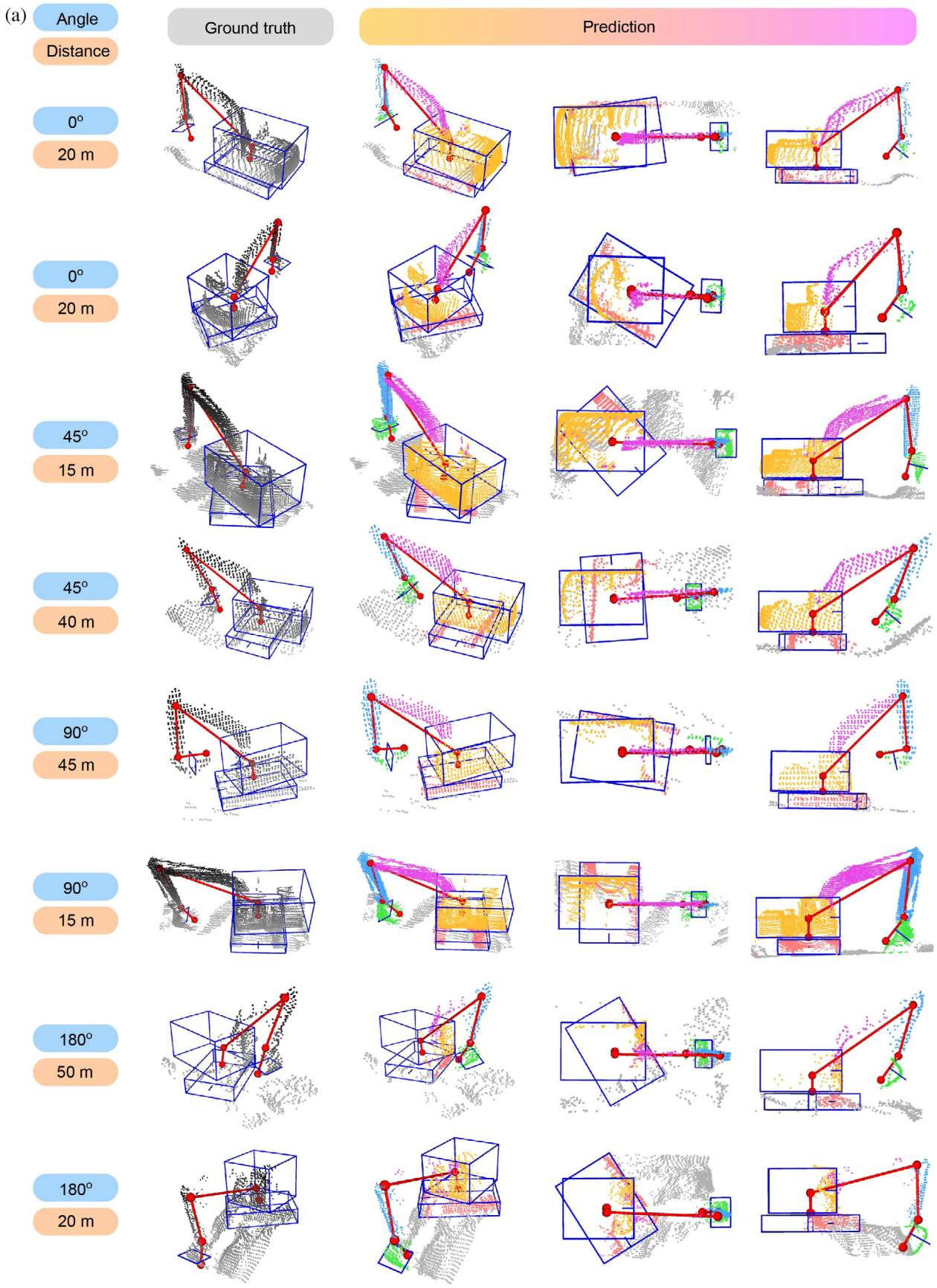

FIGURE 9 Visualization of the pose estimation results: (a) different view angles and distances and (b) diverse conditions.

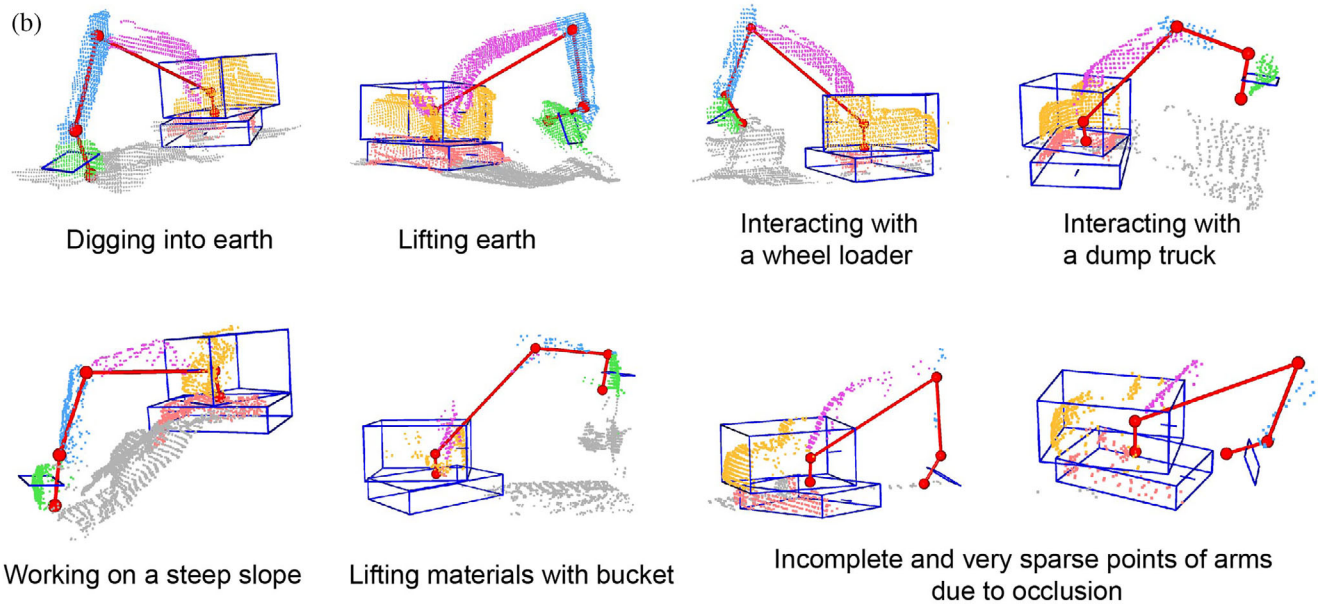

FIGURE 9 Continued

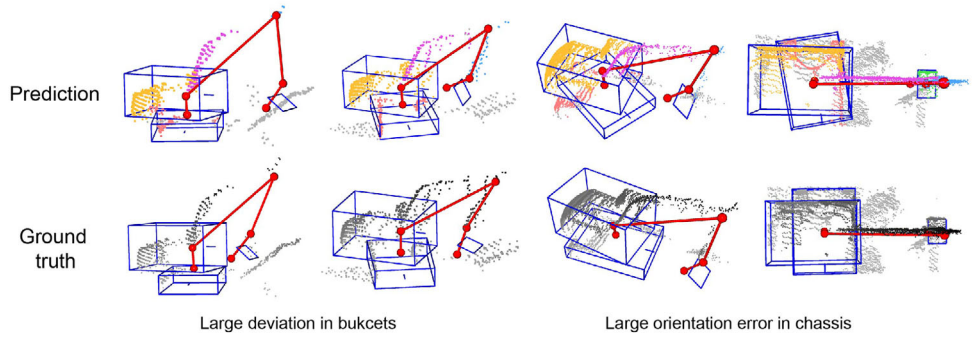

FIGURE 10 Error cases.

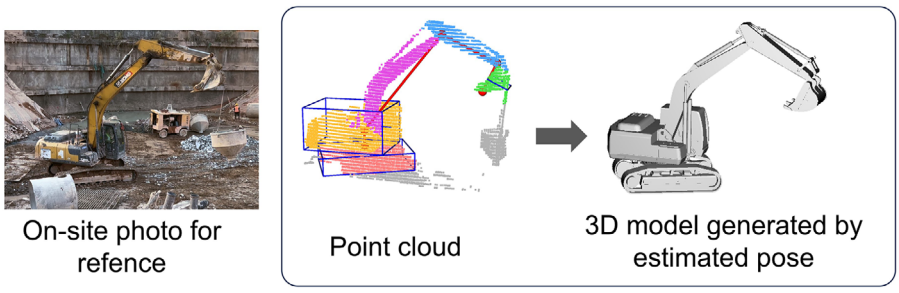

FIGURE 11 The predicted 3D excavator model compared with on-site photo.

various metrics. Figure 12 illustrates examples where predictions before fine-tuning only roughly align the point cloud, whereas after fine-tuning, the alignment becomes accurate. The second experiment involves removing size supervision (w/o size loss). The findings indicate that size loss enhances the accuracy of key points. Furthermore, eliminating the size loss results in a significant decrease in 3D IoU, primarily due to significant deviations in the sizes of the cab and chassis. The third experiment removes point cloud segmentation loss (w/o seg loss), revealing that nearly all performance metrics degrade considerably, coupled with notable instability during training. Thus,

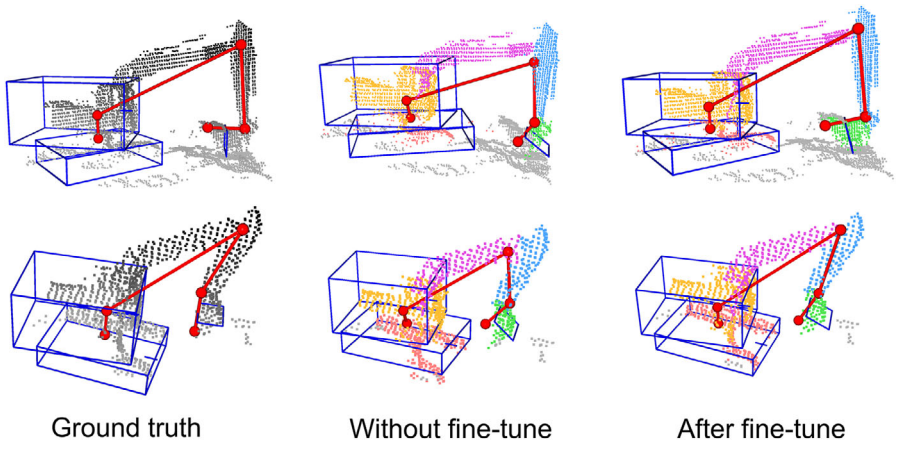

FIGURE 12 Effect of fine-tuning.

accurate segmentation is essential for effective and stable training.

The ablation study reveals the contributions and dependencies of the loss functions. The P2P loss is the core supervision, driving the pose estimation to align with the observed point cloud shape. Without this loss, the model would lack the fundamental guidance to learn meaningful poses from unlabeled data. While segmentation is not the primary task, the segmentation loss is crucial for the component-aware P2P loss. Without reliable segmentation, the P2P loss might misapply geometric constraints, hindering effective pose learning. Furthermore, the size loss works in conjunction with the P2P loss to resolve pose ambiguities. While P2P loss aligns the pose with the point cloud shape, size loss complements P2P by ensuring that the estimated pose is not only geometrically aligned but also dimensionally consistent, leading to more accurate and stable pose estimations.

6 | LIMITATIONS AND FUTURE WORK

As illustrated in the failure cases in the results section, occlusion can negatively impact the performance of pose estimation. This issue primarily stems from self-occlusion, particularly when the view angle is close to 0 degrees and the excavator cab occludes the arms. To address this, future work could explore deploying multiple LiDARs at the site with orthogonal viewpoints, thereby minimizing self-occlusion caused by the excavator. Furthermore, pose estimation robustness in occluded frames could be enhanced by leveraging temporal information, for example, by utilizing results from un-occluded frames in previous time steps. Incorporating a cross-frame position consistency loss and applying post-processing methods, such as Kalman filtering, are promising avenues to significantly reduce pose estimation errors in occluded frames and will be considered in our future research.

In addition, the orientation prediction of the chassis occasionally shows significant errors. While multi-frame fusion can help alleviate this issue, it may be worthwhile to explore more robust angle regression methods. For instance, adopting an angle bin classification approach could simplify the learning process instead of direct regression. Furthermore, relying solely on cosine and sine losses could lead to a discontinuous loss function (Yu & Da, 2024), which needs further improvement in future work.

The proposed approach operates independently of object detection, requiring sequential execution of the object detection and pose estimation models during deployment. The performance of the pose estimation method is therefore dependent on the accuracy of the upstream object detection. To improve efficiency, future research should explore end-to-end pose estimation methods that combine these tasks within a single network.

Last, it is notable that the proposed training framework and loss functions are not restricted to excavator pose estimation, as most machinery exhibits similar structures—multi-segment articulated links and cuboid components rotating around axes. With proper modification on the geometric representation and loss functions, this idea can be extended to the pose estimation of various articulated machinery, including robotic arms, dump trucks, and cranes.

7 | CONCLUSION

This study presents a method for estimating the full-body 3D pose of excavators from point clouds without requiring manual 3D annotations. Built upon the kinematic constraints and geometric characteristics of excavators,

the approach represents the five excavator components through the geometric primitives, including their coordinates, rotations, and sizes. A unified point-based network is proposed to regress these pose parameters at multiple levels and is trained in a two-stage framework: The first stage uses synthetic data to initialize parameters; in the core second stage, the network is fine-tuned to align real-world data through the proposed self-supervised loss. An evaluation of the proposed method on real construction sites is also presented. The method achieves a joint localization error of 0.26 m, with constant performance at various distances and view angles. Moreover, the method shows strong performance in diverse and challenging operation scenarios, such as interacting with the environment and incomplete point clouds. These findings demonstrate the method's robustness and potential for practical application. Future research will focus on mitigating the impact of occlusion on predictions and expanding the application of this method to other mechanical structures.

ACKNOWLEDGMENTS

This research was funded by the National Natural Science Foundation of China [No. 42302322], Internal Research Fund of PolyU (UGC) [No. P0047899], and Collaborative Research Fund (CRF) from the Research Grants Council (Hong Kong) [No. C6044-23GF].

REFERENCES

Assadzadeh, A., Arashpour, M., Li, H., Hosseini, R., Elghaish, F., & Baduge, S. (2023). Excavator 3D pose estimation using deep learning and hybrid datasets. *Advanced Engineering Informatics*, *55*, 101875. https://doi.org/10.1016/j.aei.2023.101875

Bang, S., Hong, Y., & Kim, H. (2021). Proactive proximity monitoring with instance segmentation and unmanned aerial vehicle-acquired video-frame prediction. *Computer-Aided Civil and Infrastructure Engineering*, *36*(6), 800–816. https://doi.org/10.1111/mice.12672

Cheng, W., Park, J. H., & Ko, J. H. (2021). HandFoldingNet: A 3D hand pose estimation network using multiscale-feature guided folding of a 2D hand skeleton. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada (pp. 11240–11249). https://doi.org/10.1109/ICCV48922.2021.01107

Cho, Y. K., & Gai, M. (2014). Projection-recognition-projection method for automatic object recognition and registration for dynamic heavy equipment operations. *Journal of Computing in Civil Engineering*, *28*(5), A4014002. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000332

Cui, Y., An, Y., Sun, W., Hu, H., & Song, X. (2022). Memory-augmented point cloud registration network for bucket pose estimation of the intelligent mining excavator. *IEEE Transactions on Instrumentation and Measurement*, *71*, 1–12. https://doi.org/10.1109/TIM.2022.3149331

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

Ge, L., Ren, Z., & Yuan, J. (2018). Point-to-point regression PointNet for 3D hand pose estimation. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Lecture notes in computer science: Vol. 11217. Computer vision – ECCV 2018* (pp. 489–505). Springer International Publishing. https://link.springer.com/10.1007/978-3-030-01261-8_29

Ionescu, C., Papava, D., Olaru, V., & Sminchisescu, C. (2014). Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(7), 1325–1339. https://doi.org/10.1109/TPAMI.2013.248

Kashani, A. H., Owen, W. S., Himmelman, N., Lawrence, P. D., & Hall, R. A. (2010). Laser scanner-based end-effector tracking and joint variable extraction for heavy machinery. *The International Journal of Robotics Research*, *29*(10), 1338–1352. https://doi.org/10.1177/0278364909359316

Kim, J., Chi, S., & Kim, J. (2023). 3D pose estimation and localization of construction equipment from single camera images by virtual model integration. *Advanced Engineering Informatics*, *57*, 102092. https://doi.org/10.1016/j.aei.2023.102092

Kim, K., Jeong, I., & Cho, Y. K. (2023). Signal processing and alert logic evaluation for IoT–based work zone proximity safety system. *Journal of Construction Engineering and Management*, *149*(2), 05022018. https://doi.org/10.1061/JCEMD4.COENG-12417

Kluger, F., Brachmann, E., Yang, M. Y., & Rosenhahn, B. (2024). Robust shape fitting for 3D scene abstraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*(9), 6306–6325. https://doi.org/10.1109/TPAMI.2024.3379014

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, X., Wang, H., Yi, L., Guibas, L. J., Abbott, A. L., & Song, S. (2020). Category-Level Articulated Object Pose Estimation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3703–3712. https://doi.org/10.1109/CVPR42600.2020.00376

Li, Y., & Ibanez-Guzman, J. (2020). Lidar for autonomous driving: The principles, challenges, and trends for automotive Lidar and perception systems. *IEEE Signal Processing Magazine*, *37*(4), 50–61. https://doi.org/10.1109/MSP.2020.2973615

Liang, C.-J., Lundeen, K. M., McGee, W., Menassa, C. C., Lee, S., & Kamat, V. R. (2019). A vision-based marker-less pose estimation system for articulated construction robots. *Automation in Construction*, *104*, 80–94. https://doi.org/10.1016/j.autcon.2019.04.004

Liu, D., Sun, C., Chen, J., & Liu, L. (2023). Multisensory and BIM-integrated digital twin to improve urban excavation safety. *Journal of Computing in Civil Engineering*, *37*(5), 04023025. https://doi.org/10.1061/JCCEE5.CPENG-5354

Liu, Y., & Jebelli, H. (2024). Intention-aware robot motion planning for safe worker–robot collaboration. *Computer-Aided Civil and Infrastructure Engineering*, *39*(15), 2242–2269. https://doi.org/10.1111/mice.13129

Mahmood, B., Han, S., & Seo, J. (2022). Implementation experiments on convolutional neural network training using synthetic images for 3D pose estimation of an excavator on real images. *Automation in Construction*, *133*, 103996. https://doi.org/10.1016/j.autcon.2021.103996

Open3D. (2024). *Open3D–A modern library for 3D data processing*. https://www.open3d.org/

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, J., Chen, J., & Cho, Y. K. (2017). Self-corrective knowledge-based hybrid tracking system using BIM and multimodal sensors. *Advanced Engineering Informatics*, *32*, 126–138. https://doi.org/10.1016/j.aei.2017.02.001

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. (Original work published 2016.) In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), *Advances in neural information processing systems* 32 (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Phillips, T. G., & McAree, P. R. (2018). An evidence-based approach to object pose estimation from LiDAR measurements in challenging environments. *Journal of Field Robotics*, *35*(6), 921–936. https://doi.org/10.1002/rob.21788

Rafiei, M. H., Gauthier, L. V., Adeli, H., & Takabi, D. (2024). Self-supervised learning for electroencephalography. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(2), 1457–1471. https://doi.org/10.1109/TNNLS.2022.3190448

Robosense. (2023). *Automotive grade LiDAR - RS-LiDAR-M1—RoboSense*. https://www.robosense.cn/en/rslidar/RS-LiDAR-M1

Sharma, G., Dash, B., RoyChowdhury, A., Gadelha, M., Loizou, M., Cao, L., Wang, R., Learned-Miller, E. G., Maji, S., & Kalogerakis, E. (2022). PriFit: Learning to fit primitives improves few shot point cloud segmentation. *Computer Graphics Forum*, *41*(5), 39–50. https://doi.org/10.1111/cgf.14601

Shen, J., Jiao, L., Zhang, C., & Peng, K. (2024). Monocular 3D object detection for construction scene analysis. *Computer-Aided Civil and Infrastructure Engineering*, *39*(9), 1370–1389. https://doi.org/10.1111/mice.13143

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., & Topin, N. (2018). *Super-convergence: Very fast training of neural networks using large learning rates*. arXiv. https://doi.org/10.48550/arXiv.1708.07120

Soltani, M. M., Zhu, Z., & Hammad, A. (2018). Framework for location data fusion and pose estimation of excavators using stereo vision. *Journal of Computing in Civil Engineering*, *32*(6), 04018045. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000783

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., … Anguelov, D. (2020). *Scalability in perception for autonomous driving: Waymo open dataset*. arXiv. http://arxiv.org/abs/1912.04838

Sun, X., Xiao, B., Wei, F., Liang, S., & Wei, Y. (2018). Integral human pose regression. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Lecture notes in computer science: Vol. 11210. Computer vision–ECCV 2018* (Vol. 11210, pp. 536–553). Springer International Publishing. https://link.springer.com/10.1007/978-3-030-01231-1_33

Torres Calderon, W., Roberts, D., & Golparvar-Fard, M. (2021). Synthesizing pose sequences from 3D assets for vision-based activity analysis. *Journal of Computing in Civil Engineering*, *35*(1), 04020052. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000937

Vahdatikhaki, F., Hammad, A., & Siddiqui, H. (2015). Optimization-based excavator pose estimation using real-time location systems. *Automation in Construction*, *56*, 76–92. https://doi.org/10.1016/j.autcon.2015.03.006

Vinutha, H. P., Poornima, B., & Sagar, B. M. (2018). Detection of outliers using interquartile range technique from intrusion dataset. In S. C. Satapathy, J. M. R. S. Tavares, V. Bhateja, & J. R. Mohanty (Eds.), *Information and decision sciences* (pp. 511–518). Springer. https://doi.org/10.1007/978-981-10-7563-6_53

Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., & Guibas, L. J. (2019). Normalized object coordinate space for category-level 6D object pose and size estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019*, Long Beach, CA (pp. 2642–2651). https://openaccess.thecvf.com/content_CVPR_2019/html/Wang_Normalized_Object_Coordinate_Space_for_Category-Level_6D_Object_Pose_and_CVPR_2019_paper.html

Weng, Y., Wang, H., Zhou, Q., Qin, Y., Duan, Y., Fan, Q., Chen, B., Su, H., & Guibas, L. J. (2021). CAPTRA: CAtegory-level Pose Tracking for Rigid and Articulated objects from point clouds. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada (pp. 13189–13198). https://doi.org/10.1109/ICCV48922.2021.01296

Weng, Z., Gorban, A. S., Ji, J., Najibi, M., Zhou, Y., & Anguelov, D. (2023). 3D human keypoints estimation from point clouds in the wild without human labels. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada (pp. 1158–1167). https://doi.org/10.1109/CVPR52729.2023.00118

Wu, X., Lao, Y., Jiang, L., Liu, X., & Zhao, H. (2022). *Point Transformer V2: Grouped Vector attention and partition-based pooling*. arXiv. http://arxiv.org/abs/2210.05666

Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., Yi, L., Chang, A. X., Guibas, L. J., & Su, H. (2020). SAPIEN: A SimulAted Part-Based Interactive ENvironment. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA (pp. 11094–11104). https://doi.org/10.1109/CVPR42600.2020.01111

Xu, Z., Bi, L., & Zhao, Z. (2023). Real-time bucket pose estimation based on deep neural network and registration using onboard 3D sensor. *SENSORS*, *23*(15), 6958. https://doi.org/10.3390/s23156958

Yu, Y., & Da, F. (2024). On boundary discontinuity in angle regression based arbitrary oriented object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*(10), 6494–6508. https://doi.org/10.1109/TPAMI.2024.3378777

Yue, H., Wang, Q., Zhao, H., Zeng, N., & Tan, Y. (2024). Deep learning applications for point clouds in the construction industry. *Automation in Construction*, *168*, 105769. https://doi.org/10.1016/j.autcon.2024.105769

Zanfir, A., Zanfir, M., Gorban, A., Ji, J., Zhou, Y., Anguelov, D., & Sminchisescu, C. (2023). HUM3DIL: Semi-supervised multi-modal 3D humanpose estimation for autonomous driving. *Proceedings of The 6th Conference on Robot Learning*, Auckland, New Zealand (pp. 1114–1124). https://proceedings.mlr.press/v205/zanfir23a.html

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*(19), 2990–3007. https://doi.org/10.1111/mice.13238

Zhao, H., Jiang, L., Jia, J., Torr, P., & Koltun, V. (2021). *Point Transformer*. http://arxiv.org/abs/2012.09164

Zheng, J., Shi, X., Gorban, A., Mao, J., Song, Y., Qi, C. R., Liu, T., Chari, V., Cornman, A., Zhou, Y., Li, C., & Anguelov, D. (2022). Multi-modal 3D human pose estimation with 2D weak supervision in autonomous driving. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, New Orleans, LA (pp. 4477–4486). https://doi.org/10.1109/CVPRW56347.2022.00494

Zhou, Y., Barnes, C., Lu, J., Yang, J., & Li, H. (2019). On the continuity of rotation representations in neural networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA (pp. 5738–5746). https://doi.org/10.1109/CVPR.2019.00589

How to cite this article: Zhang, M., Guo, W., Zhang, J., Han, S., Li, H., & Yue, H. (2025). Excavator 3D pose estimation from point cloud with self-supervised deep learning. *Computer-Aided Civil and Infrastructure Engineering*, 1–19. https://doi.org/10.1111/mice.13500