

VoxelKP: A Voxel-based Network Architecture for Human Keypoint Estimation in LiDAR Data

Jian Shi
KAUST

jian.shi@kaust.edu.sa

Peter Wonka
KAUST

peter.wonka@kaust.edu.sa

Abstract

We present *VoxelKP*, a novel fully sparse network architecture tailored for human keypoint estimation in LiDAR data. The key challenge is that objects are distributed sparsely in 3D space, while human keypoint detection requires detailed local information wherever humans are present. First, we introduce a dual-branch fully sparse spatial-context block where the spatial branch focuses on learning the local spatial correlations between keypoints within each human instance, while the context branch aims to retain the global spatial information. Second, we use a spatially aware multi-scale BEV fusion technique to leverage absolute 3D coordinates when projecting 3D voxels to a 2D grid encoding a bird’s eye view for better preservation of the global context of each human instance. We evaluate our method on the Waymo dataset and achieve an improvement of 27% on the MPJPE metric compared to the state-of-the-art, HUM3DIL, trained on the same data, and 12% against the state-of-the-art, GC-KPL, pretrained on a 25× larger dataset. To the best of our knowledge, *VoxelKP* is the first single-staged, fully sparse network that is specifically designed for addressing the challenging task of 3D keypoint estimation from LiDAR data, achieving state-of-the-art performance. Our code is available at <https://github.com/shijianjian/VoxelKP>.

1. Introduction

Human pose estimation (HPE) is a critical area of research with applications spanning computer vision, robotics, human-computer interaction, and augmented/virtual reality. Previous works [23, 32, 36] are mostly based on 2D images and videos. Compared to regular RGB input, LiDAR sensors provide detailed 3D structural information by measuring the distance to objects using laser light. Apart from its robustness under occlusion and illumination changes, LiDAR also offers privacy protection as it can not retain facial details. Existing 3D HPE approaches often rely on multi-stage pipelines [40], or require additional supervision, either from synthetic LiDAR data [37] or RGB im-

ages [42, 44]. Despite significant progress in 3D object detection from LiDAR point clouds [26–28], our experiments (in supplementary material) show that simply introducing a keypoint estimation head to these models often leads to suboptimal performance in 3D HPE tasks. Based on this observation, we suspect the object detection methods focus on capturing objects scattered sparsely across the 3D space, while the keypoints tend to be distributed densely within localized regions around the human body. This fundamental discrepancy in the context captured by existing detectors limits their suitability for precise 3D keypoint prediction due to the lack of fine-grained spatial information. In addition, most 3D object detectors use projected bird’s-eye view (BEV) features to reduce computational costs. These height-compressed features complicate the precise keypoint localization. To address these gaps, we aim to extend the success of 3D object detection to 3D HPE for LiDAR point cloud data by introducing novel components to preserve fine-grained spatial information while retaining height information. As shown in Figure 1, our method significantly improves the precision of the estimated keypoints.

We introduce *VoxelKP*, a novel fully sparse neural network tailored specifically for human pose estimation within LiDAR point clouds. We introduce a dual-branch *fully sparse spatial-context block* that integrates local dense features with the global spatial context from the sparse representations of LiDAR scans. More precisely, the spatial branch is used to extract local spatial correlation between keypoints, whilst the context branch is used to preserve the global spatial details. In this work, we refer to local features as fine-grained keypoint details within a detected human, and global features as larger-scale scene context. Second, we propose a *spatially aware multi-scale BEV fusion* module that aims to effectively encode absolute 3D coordinates to light-weight BEV representations. To the best of our knowledge, *VoxelKP* is the first single-staged, fully sparse network that is specifically designed for addressing the challenging task of 3D HPE from LiDAR data, achieving 27% on the MPJPE metric compared to the current state-of-the-art trained on the same data.

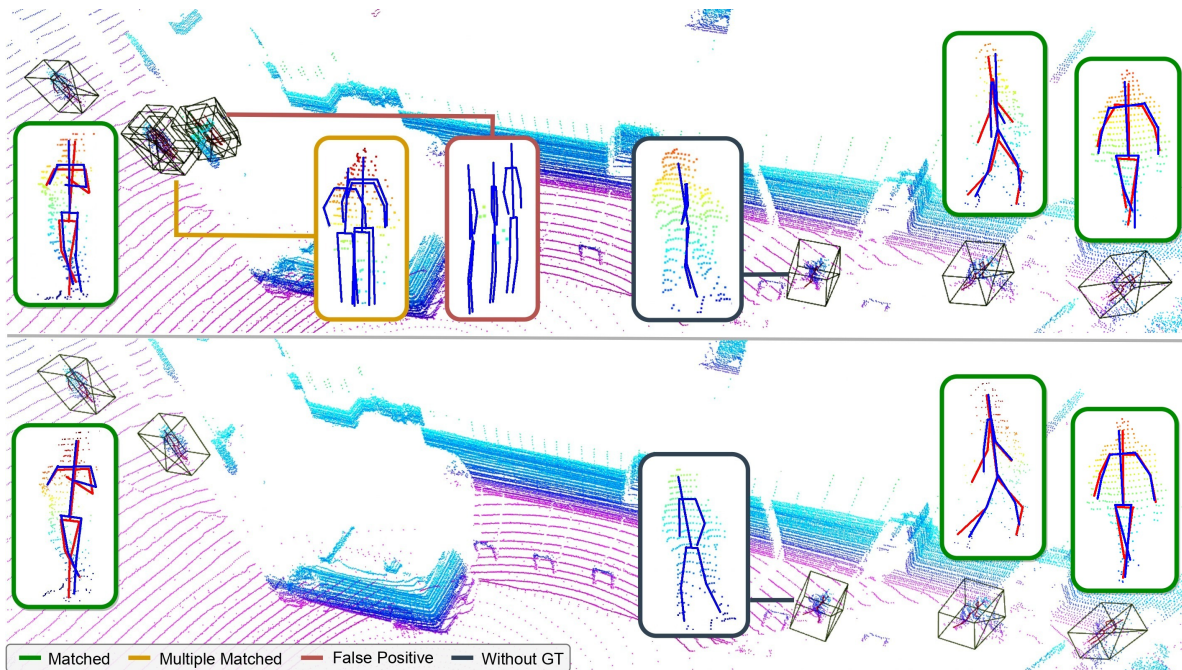


Figure 1. A visual demonstration of our baseline model (top) and the proposed *VoxelKP* (bottom). Our *VoxelKP* offers improved estimated human poses with precise locations and fewer false positives. The insets are color-coded according to the legend in the figure. In the green-colored insets, a comparison with the ground truth is shown, with ground truth in red and predictions in blue. Our baseline model is *VoxelNeXt* with additional human pose outputs.

2. Related Work

2.1. Deep Learning on Point Clouds

Many neural network architectures have been proposed for processing smaller-scale point clouds [16, 22, 24, 25, 35]. However, typical LiDAR-generated point clouds contain more than 100,000 points, making point-by-point computations overwhelming due to the massive data scale. VoxelNet [45] proposed a voxel feature encoding (VFE) layer as a workaround for the high computational and memory issues brought by point-by-point computations. Meanwhile, sparse and submanifold sparse convolution operations [6] exploit sparsity in the voxel grid to reduce computations. SECOND [38] introduced an efficient sparse convolutional approach that benefits from the sparse operations. Following SECOND, subsequent works like PointPillars [13], 3DSSD [39], PV-RCNN [28], CenterPoint [41] further advanced sparse convolutional detection on point clouds. VoxelNeXt [2] further demonstrates a fully sparse voxel-based method without sparse-to-dense conversion or NMS post-processing. However, these approaches are targeted at improving bounding box localization accuracy, which does not require fine-grained spatial features for precise pose estimation tasks. Instead, we propose *VoxelKP*, a novel sparse convolutional architecture tailored for learning discriminative local features from sparse LiDAR data for accurate human pose estimation.

2.2. Human Pose Estimation on Point Clouds

Human pose estimation has been extensively studied in images, with methods like DeepPose [36], Stacked Hourglass [23], and HRNet [32] achieving high accuracy on benchmarks like COCO-wholebody [10]. However, compared to RGB images, point clouds provide explicit 3D structural information about the shape and depth of objects. [29] pioneered point cloud human pose estimation from a single depth image. Recent works such as [21, 46] proposed a deep learning-based 3D human pose estimation from depth images. Waymo [33] has released keypoint annotations for LiDAR-collected point cloud scenes, while only 3% of the frames are annotated with keypoint human poses. Due to the scarcity of the keypoint annotations within LiDAR point cloud data, many works have taken semi-supervised or weak-supervised approaches to compensate for the limited availability of labeled 3D pose data. For example, [42, 44] took a multi-modal approach to utilize the enriched image annotations to assist the recognition from point clouds. [37] proposed an unsupervised approach that generates pseudo ground truth without using annotated keypoint data, along with a fine-tuning approach that pretrains the model with synthetic data and then fine-tunes on the training set. A concurrent work [40] adopted a fine-tuning strategy that used a frozen backbone pretrained on a large-scale dataset as a feature extractor, achieving

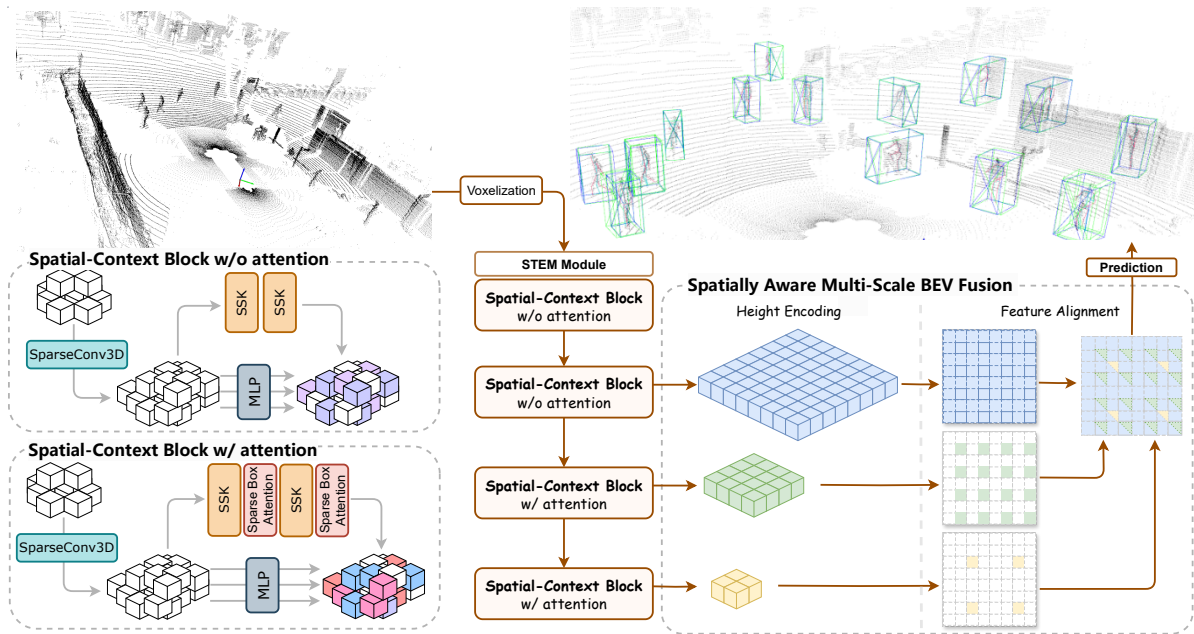


Figure 2. The overall architecture of *VoxelKP*. The model begins with voxelizing a point cloud scene, followed by feature extraction using a stem module (Appendix A.1). Subsequently, the extracted sparse voxel features are processed through four fully sparse spatial-context blocks (Sec. 3.2) for capturing local and global spatial information. Lastly, we utilize a spatially-aware multi-scale BEV representation (Sec. 3.3) for accurate human pose estimation.

plausible performance. In general, multi-person pose estimation from solely point clouds remains relatively unexplored due to the lack of ground-truth 3D human pose annotations. This work proposes a single-staged 3D HPE method with only LiDAR point clouds, achieving comparable performance without extra training data.

3. Method

We propose a single-stage, fully sparse neural network, designed for 3D HPE within LiDAR point clouds. Unlike object detection, 3D HPE not only requires awareness of sparsely distributed human locations within the point cloud but also demands the ability to capture intricate spatial relationships between human keypoints. As illustrated in Figure 2, our *VoxelKP* framework contains two key components: 1) *fully sparse spatial-context blocks* to enhance the capturing of local sparse voxel features, and 2) *spatially aware multi-scale BEV fusion* to encode height information from the 3D voxel grid into 2D BEV maps. Our components are based on sparse operations that improve computational efficiency by skipping the computation for empty voxels. In this section, we first present the formulation of the task, then introduce the key components proposed in our network, and finally elaborate on the details of the network architecture.

3.1. Problem Formulation

Given a 3D point cloud scanned by LiDAR sensors, our goal is to estimate the 3D locations of K keypoints that represent the human pose. Let the input point cloud P be

$\mathbb{R}^{N \times C}$ where N is the number of points and C is the number of features (e.g. x, y, z, intensity, elongation). We use a sparse voxel representation to represent point clouds, which consists of two separate tensors: one feature tensor $\mathbb{R}^{V \times C}$ and one index tensor $\mathbb{R}^{V \times 4}$ where V is the number of non-empty voxels and 4 dimensions are used for batch sample index and the three coordinates of each voxel. We define the ground truth pose for the i^{th} human as a set of 3D keypoint locations $G_i = \{g_i^1, g_i^2, \dots, g_i^K\}$ where $g_i^k \in \mathbb{R}^3$ is the location of the k^{th} keypoint in the global coordinate frame. The set of K keypoints corresponds to anatomical joints of interest such as shoulders, elbows, wrists, hips, knees, and ankles. Our objective is to predict the 3D keypoint locations from the input point cloud, i.e. to learn a function F such that $\hat{G} = F(P)$, where $\hat{G} \in \mathbb{R}^{M \times K \times 3}$ is the tensor of predicted 3D keypoint locations of M humans.

3.2. Fully Sparse Spatial-Context Block

We design specialized building blocks to process spatial information of varying densities effectively. Each building block starts with a basic sparse 3D block, subsequently branching into two distinct pathways for local and global spatial feature learning. Spatial branches are used for learning local spatial correlations between keypoints, incorporating *sparse selective kernel modules* (Sec. 3.2.1) and *sparse box-attention modules* (Sec. 3.2.2). Context branches use a straightforward MLP to maintain per-voxel information, ensuring the retention of global spatial details. The *hybrid feature learning* (Sec. 3.2.3) strategy captures the nuanced

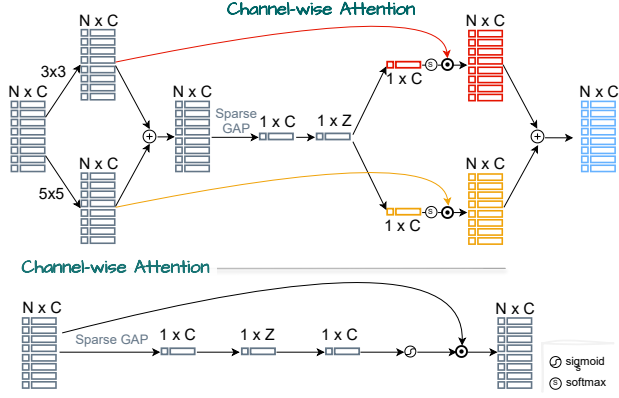


Figure 3. **Sparse selective kernel** module with one sample input. The SSK module selects the best kernels from different receptive fields with a softmax-based channel-wise attention mechanism.

local details with an understanding of the global context. Unlike *PVCNN* [19] and *PVT* [43], our approach does not require voxelization and devoxelization operations between modules, where the input is voxelized once, and all operations remain sparse throughout.

3.2.1. Sparse Selective Kernel Module

The sparse nature of voxel data means that only a limited set of non-empty voxels contribute to feature extraction, making it challenging to capture all relevant spatial details. Inspired by [15], we propose the sparse selective kernel (SSK) module that selectively aggregates multi-scale sparse features to improve spatial context. As demonstrated in Fig. 3, we first use two sparse 3D submanifold convolution branches with varied receptive field sizes of $3 \times 3 \times 3$ and $5 \times 5 \times 5$. A submanifold convolution computes output values only if the convolution kernel is centered on a non-empty voxel, i.e., the number of non-empty voxels remains the same. Next, the sparse features from each branch are summed up and then fed into a selection module that compresses the spatial dimension by a sparse global average pooling (sGAP) to compute the average feature of all non-empty voxels, then a feature squeeze and expansion are applied [7]. The squeeze and expansion process compresses a feature map from C channels to Z channels, then expands it back to C channels, where Z is 25% of C in our implementation. We denote a voxel position as p , the set of voxel positions within a voxel grid as P_s , and the feature corresponding to voxel position p as f_p . The sGAP is achieved by: $\bar{F}_s = \{\frac{1}{|S_{\bar{p}}|} \sum_{p \in S_{\bar{p}}} f_p | p \in P_s\}$, where $S_{\bar{p}}$ and $|S_{\bar{p}}|$ are the set of valid voxels and the number of valid voxels, respectively. This produces channel-wise attention weights after a softmax activation, allowing the network to emphasize or suppress the features from each branch selectively.

3.2.2. Sparse Box-Attention Module

Intuitively, the keypoint location is only determined by its local surrounding region, and the global context can barely

help. In fact, as demonstrated in Sec. 5.1, the integration of global features can even harm the estimation accuracy. Thus, unlike the previous works that tried to capture a wider range of global features with self-attention methods for segmentation tasks [11, 12], we focus on localized feature attention to resolve the densely distributed keypoints in local regions. The key idea is to partition the sparse 3D voxel space into non-overlapping boxes by aggregating voxels by their spatial coordinates. Within each local box, we apply self-attention to capture dependencies between the voxels inside the box. The sparse voxel features in each box go through a linear layer for the queries Q , keys K , and values V , where $Q, K, V \in \mathbb{R}^{n_b \times h \times d}$ and n_b, h, d are the number of valid voxels in the b -th box, attention heads, and feature dimensions. Since we are using sparse tensor representations, each box partition may contain a varying number of voxels. Referring to [11, 43], we then compute the attention map by the following equation:

$$\begin{aligned} \text{attn}_{i,j,h} &= Q_{i,h} \cdot K_{j,h}, & \hat{\text{atn}}_{i,,h} &= \text{softmax}(\text{atn}_{i,,h}) \\ y_{i,h} &= \sum_{j=1}^{n_b} \hat{\text{atn}}_{i,j,h} \times V_{j,h}. \end{aligned} \quad (1)$$

The output is then obtained by applying a projection layer and a residual connection, as shown in Fig. 4.

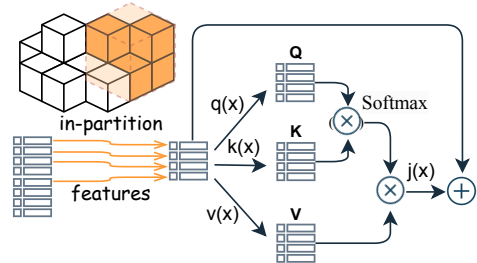


Figure 4. **Sparse box-attention**. This attention mechanism selects the voxel features that correspond to one box partition referring to the index tensor and then performs self-attention on the selected voxels. The functions q , k , v , and j are linear layers.

3.2.3. Hybrid Feature Learning

The sparse convolutional operations aggregate neighborhood information via spatial coordinates, which can overlook the fine-grained per-voxel details essential for keypoint localization. Concurrently, inspired by the previous point-voxel networks [19, 28, 43], we include an MLP branch for each stage. Differently, as a benefit of the fully sparse design, our approach avoids repeated voxelization and devoxelization between modules. The integration of an MLP branch alongside a convolutional branch is a strategic approach to capture both fine-grained per-voxel details and relatively coarse-grained local neighborhood information. Each MLP branch is composed of three sequential blocks, each consisting of a linear layer, batch normalization, and

a ReLU activation function. The number of channels in each linear layer is set to match the channels of the incoming tensor. We then merge the output sparse voxel features from the MLP and sparse convolutional branches through element-wise summation to create hybrid features of the per-voxel and per-neighborhood information.

3.3. Spatially Aware Multi-Scale BEV Fusion

Compressing features into bird’s eye view (BEV) maps is a common practice for object detection [1, 38] to collapse the point cloud to 2D for efficiency. However, different from object detection tasks, height information is essential for 3D HPE tasks to precisely locate each keypoint. For a sparse 3D voxel grid of size $C \times X \times Y \times Z$, we use C to denote the number of features per voxel, X and Y as the spatial extent in the ground plane, and Z as the up axis. Ignoring the height information such as [2] is clearly unreasonable for 3D HPE. Another intuitive approach is to directly deploy 3D feature maps. Unfortunately, this direct 3D approach does not lead to a decent performance as training does not converge well due to the large amount of irrelevant features, as shown in Tab. 5. We, therefore, propose a *spatially aware multi-scale BEV fusion* approach for fusing sparse voxel features from multiple encoder layers in a way that retains spatial information, as illustrated in Fig. 5. Specifically, we use height encoding and scale-wise feature alignment to compensate for the loss of spatial information during the 2D projection.

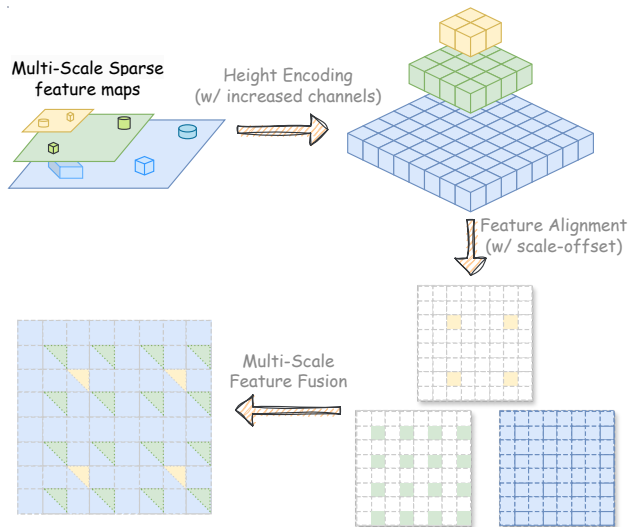


Figure 5. **Spatially aware multi-scale BEV Fusion.** We use a dense representation for a better visual illustration of the method.

Height Encoding Transforming 3D data into BEV is often used in 3D object detection and segmentation tasks, for reducing the dimensionality of point clouds and making them more manageable for processing. An object detection method may project the 3D voxel grid to a 2D BEV repre-

sentation by adding features from voxels that share the same x and y position, losing the information about which height a feature was taken from. Instead, we use a height encoding method that compresses the height dimension to 1 using convolution kernels of size $(1, 1, h)$ where h is the height of each 3D voxel grid. Meanwhile, we increase the number of resulting channels to retain more spatial details and features from the 3D representation. This provides a richer representation for the 2D regression heads to work with.

Multi-scale Feature Alignment After obtaining z multi-scale height-encoded BEV maps from the last few stages of the network, we then fuse those feature maps to create a feature map that contains multi-scale sparse voxel features. Unlike working with dense tensors, the direct interpolation of the feature maps in the sparse case is computationally complex, as it requires specialized algorithms to efficiently navigate through the predominantly empty voxels to find and interpolate the adjacent non-empty voxels. Instead, we directly modify the feature position of the sparse tensor by multiplying the voxel position by its scale r . To avoid overlapping feature positions of $(x_p * 2^r, y_p * 2^r)$ $r \in \{0, 1, 2, \dots\}$ during the scale multiplication, we align the xy -plane positions (x_p, y_p) using scale offsets $(x_p * 2^r + r, y_p * 2^r + r)$, where p is the position of a voxel in a voxel grid.

By stacking the r -scaled feature maps together, we obtain a multi-scale 3D feature map with a height of r . To obtain a BEV feature map, instead of collapsing with $1 \times 1 \times r$ convolutions, we simply apply an intuitive scaling for each scale of the feature map. The scaling factor \hat{r}_p is proportional to the height (scale) of the 3D feature map for each feature f_p at position p , then we obtain scaled feature $\bar{f}_p = f_p \cdot \hat{r}_p$. Given \bar{P} as the set of the 2D xy -plane positions in the voxel grid, the compressed sparse features \bar{F} and their positions \bar{P} are obtained as:

$$\bar{P} = \{(x_p, y_p) | p \in P\}, \quad \bar{F} = \left\{ \sum_{p \in S_{\bar{p}}} \bar{f}_p \mid \bar{p} \in \bar{P} \right\}, \quad (2)$$

where $S_{\bar{p}} = \{p | x_p = x_{\bar{p}}, y_p = y_{\bar{p}}, p \in P\}$ contains voxels that are put onto the same 2D xy -plane position \bar{p} .

3.4. Network Architecture

The input is a point cloud $\mathbb{R}^{N \times C}$ where N is the number of points and C is the number of features (e.g. x, y, z , intensity). We voxelize the point cloud into a sparse voxel representation. We use an input stem network and four stages with gradually decreased feature map size, where each stage reduces the spatial shape of the sparse voxel space by a factor of two. The input stem network is a simple stack of convolution layers, as shown in Appendix A.1, to extract low-level features from the voxelized point cloud. Next, four *fully sparse spatial-context blocks* are used for each subsequent stage for capturing features for accurate keypoint

Method	Dataset	Description	MPJPE cm.
With Extra Training Data			
Zheng <i>et al.</i> [44]	(CVPR 22)	Internal dataset + Waymo v.?	Trained on 155, 182 objects from internal data. Generated pseudo labels from 2D image labels. 10.80 (-18%)
GC-KPL [37]	(CVPR 23)	Waymo v.?	Pre-trained on synthetic data. Fine-tuned on ground truth 11.27 (-21%)
		Waymo v.?	Pre-trained on 200, 000 Waymo objects. Fine-tuned on ground truth 10.10 (-12%)
Without Extra Training Data			
HUM3DIL [42]	(CoRL 22)	Waymo v.1.3.2	Randomly initialized 12.21 (-27%)
VoxelKP		Waymo v.1.4.2	Randomly initialized 8.87

Table 1. Benchmark results. The numbers in the table are taken from their corresponding papers aside from *HUM3DIL*, which is taken from *GC-KPL* paper. It is unclear about the exact training dataset used for *Zheng et al.* and *GC-KPL*. Waymo v1.3.2 and Waymo v1.4.2 share the same data for the human pose estimation task.

localization. The *sparse box-attention modules* are only applied for our last two blocks to emphasize local-region features. Note that we do not increase the number of channels for the last three stages. We then convert the resulting 3D feature maps from the last three blocks to 2D spatial-encoded BEV representations. Note that we increase the number of channels for the BEV representation to compensate for the information loss of the BEV conversion. These 2D features are further refined with 2D convolutions to aggregate spatial context. In the end, we obtain the estimated keypoints $Y_{kp} \in \mathbb{R}^{K \times 3}$ and the corresponding predicted visibilities $Y_{kp} \in \mathbb{R}^K$, where K is the number of keypoints. The details of the used heads are provided in Appendix A.1.

4. Experiments

4.1. Implementation Details

Dataset We use the Waymo v1.4.2 dataset [33]. During the training, we merged “Pedestrian” and “Cyclist” classes together as a “Human” class. Note that there are only 8, 125 human examples with keypoint annotations whilst over 1 million bounding box annotations. We, therefore, removed the points inside those bounding boxes without keypoint annotations. Each human object is labeled with 14 3D keypoints (nose, left/right shoulders, left/right elbows, left/right wrists, left/right hips, left/right knees, and left/right ankles, head). Additional evaluations on *SLOPER4D* [5] are provided in the supplementary.

Network The architecture of the network is composed of a stem module followed by four stages, with output channels set to 64, 128, 256, 256, and 256, respectively. Given the high resolution (*e.g.* $1504 \times 1504 \times 61$) of the voxelized point cloud input, we employ larger sparse convolution kernels (kernel size $k = 5$) for the downsampling block in both the stem module and the initial stage. For the subsequent three stages, we revert to a smaller kernel size ($k = 3$). To compensate for the information loss in the BEV projection, we increased the channels from 256 to 384.

Training We use the point cloud range of $(150.4m, 150.4m, 6m)$ for the Waymo dataset and we transform them into voxel representations by a voxel size of

$(0.1m, 0.1m, 0.1m)$. We directly use the global keypoint locations without any encoding. Due to the limited number of training samples, we first apply a ground truth sampling technique [3, 38] to concatenate target objects from other frames into the sampled frames. Next, we apply global augmentations on the whole point cloud, including random flips on the x and y axes, random scale of the range of $[0.95, 1.05]$, and random rotation ranged from $[-\pi/4, \pi/4]$. Additionally, we apply local augmentations on each annotated object, including the random scale of the range of $[0.95, 1.05]$, random rotation ranged from $[-\pi/20, \pi/20]$, random frustum dropout [8] with an intensity range from $[0., 0.2]$, and random noise around the object. Our model is trained using AdamW [20] optimizer plus OneCycle [31] learning rate scheduler to mitigate overfitting [30]. Specifically, we use a learning rate of 0.003, weight decay of 0.01, and 0.9 momentum. The details of the used loss functions can be found in Appendix A.3.

4.2. Evaluation

Benchmark Methods There is a limited amount of relevant research for this task. Most prior works utilize additional training data beyond the 3D keypoint data within the Waymo dataset. To provide a fair comparison, we consider approaches that use extra data and those that rely solely on Waymo separately. Zheng *et al.* [44] adopted a pseudo-label generation approach to provide stronger supervision. It utilizes an internal dataset for training and uses Waymo for evaluation. *GC-KPL* [37] pre-trains its backbone model with extra synthetic or real-world data, then fine-tunes the model with the full Waymo training set. Given the reliance on extra data in these methods, we consider the LiDAR-only version of *HUM3DIL* [42] as our primary competitor. *HUM3DIL* shares the exact same training data as our approach, allowing a direct comparison of techniques.

Results Previous methods like *GC-KPL* use a subset of the validation data for evaluation, while we evaluate our method with the full validation set for better reproducibility. We report MPJPE on matched keypoints for our benchmark, following prior works. As shown in Tab. 1, we outperform the baseline *HUM3DIL* by approximately 27% in MPJPE.

Components				MPJPE							PEM			
Spatial BEV	SSK	Attention	Hybrid Feat.	head	shoulders	elbows	wrists	hips	knees	ankles	all	all		
				0.0659	0.1127	0.1693	0.2020	0.0961	0.1343	0.1982	0.1394	(0%)	0.1973	(0%)
✓				0.0737	0.1026	0.1457	0.2013	0.0878	0.1285	0.1954	0.1332	(+ 4%)	0.1953	(+ 1%)
✓	✓			0.0603	0.0848	0.1232	0.1715	0.0759	0.1084	0.1608	0.1118	(+20%)	0.1889	(+ 4%)
✓	✓	✓		0.0558	0.0604	0.0903	0.1679	0.0620	0.1091	0.1834	0.1039	(+25%)	0.1791	(+ 9%)
✓	✓	✓	✓	0.0570	0.0669	0.0948	0.1467	0.0670	0.0820	0.1084	0.0887	(+36%)	0.1695	(+14%)

Table 2. Overall ablation for the effectiveness of each component. The first and last row represent the baseline method and our proposed *VoxelKP* architecture, respectively.

Our approach achieves state-of-the-art results among methods trained solely on Waymo ground truth. We also surpass the approaches leveraging extra synthetic data, beating *Zheng et al.* with synthetic pseudo labels by around 18% and *GC-KPL* with synthetic point clouds by about 21%. We achieve better performances as the SOTA *GC-KPL* approach which is pre-trained on 200,000 real-world samples by about 12%. Overall, we demonstrate significant improvements over both the baseline solely using Waymo 3D keypoint data, as well as other techniques relying on extra data. Notably, our method can even achieve better results than previous multi-modal methods, *e.g.* [44]’s multi-modal approach obtained 10.32 MPJPE with both LiDAR and RGB data while we achieved 8.87 with LiDAR only. A visual demonstration is presented in Fig. 1. Please find the accompanying video in the supplementary for a visualization of the results. In addition, we report the full spectrum of the evaluation in Appendix B, including keypoint-wise MPJPE, OKS@AP, and PEM. Our qualitative results in Fig. 6 clearly show that our method offers more reasonable estimations on the human objects that are not annotated by Waymo without pose distortions.

5. Ablations

We demonstrate the effectiveness of each proposed component in Tab. 2. We use the architecture of *VoxelNext* [2] as the baseline model, then gradually update the baseline model with the proposed components. We start with *VoxelNext* for two reasons: 1) it is one of the state-of-the-art point cloud object detection models with a fully sparse architecture design, and 2) it provides a good balance between computational costs and performance. We report the MPJPE for our ablations. The results indicate that all the individual components can contribute to improving 3D HPE. Compared to the baseline architecture, our proposed *VoxelKP* framework improves MPJPE by 36% and PEM by 14%. Next, we further present the ablation studies to show the alternative design choices of the individual component.

5.1. Different Attention Mechanism

Recent advancements, such as the stratified self-attention [11], focus on aggregating long-range contextual information, particularly beneficial for segmentation tasks. However, for human pose estimation tasks, capturing global

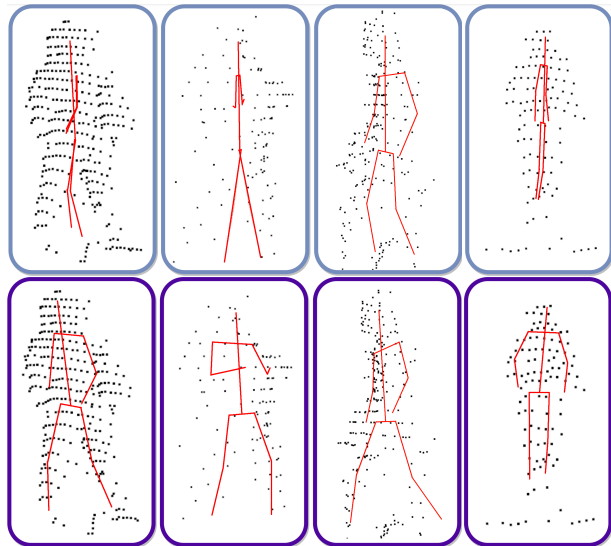


Figure 6. A visual demonstration of the baseline model (top row) and *VoxelKP* (bottom row) on human objects that are not annotated by Waymo. The baseline model tends to produce keypoints that are squeezed together, whereas our approach yields a better result.

dependencies is less crucial. Instead, our approach utilizes local box-attention, which concentrates on adjacent local regions. The results, as presented in Tab. 3, demonstrate that local box-attention outperforms other methods. Interestingly, we found that the stratified attention mechanism could slightly impair performance. We suspect that the box-based approach concentrates on areas most relevant to each keypoint location, whereas long-range attention may cause the network to overlook local, dense details. As a result, the box-based attention mechanism allows efficient modeling of local keypoint distributions, without excessive computation or over-smoothing from global aggregation.

	head	shoulders	elbows	wrists	hips	knees	ankles	all
<i>w/o</i>	0.0659	0.0956	0.1405	0.1855	0.0831	0.1077	0.1515	0.1181
stratified	0.0650	0.0911	0.1347	0.1995	0.0819	0.1245	0.1919	0.1266
box	0.0570	0.0669	0.0948	0.1467	0.0670	0.0820	0.1084	0.0887

Table 3. Different attention methods. *w/o* denotes no attention.

5.2. The MLP Branch For Hybrid Feature Learning

As shown in Fig. 2, the MLP is used for all the blocks, whilst the SSK and attention modules are only used for the

last two blocks. Compared expensive 3D operations SSK and sparse attention, the MLP branch is a lightweight linear module that can be used to ensure global awareness for our model. Intuitively, the sparse box-attention aims to handle high-level feature patterns while the early-stage blocks are low-level feature extractors. If we deploy the sparse attention in the first stage, it will immediately encounter OOM on A100 80G GPUs, even though a CUDA optimized version is already used. Meanwhile, deploying it in the second stage significantly slows down the speed (over $5\times$ slower) without performance boosts. In addition, from Tab. 4, we can see that the SSK and sparse attention create significant computational overhead with very high MACs, especially for the first two stages.

Stage	Speed (ms)				MACs(G)			
	1	2	3	4	1	2	3	4
MLP	1.594	1.591	1.584	1.624	8.1	8.1	2.2	0.5
SSK	42.731	35.88	13.885	6.388	5633.4	2816.7	352.1	44.0
Attention	22.980	10.759	4.965	4.570	136638.0	26671.4	2103.9	468.6

Table 4. MACs for each sparse module are estimated by assuming a dense voxel grid input.

5.3. Spatially Aware BEV

This ablation evaluates the effectiveness of the proposed spatially aware BEV module. We first evaluate the direct use of a naïve 2D and 3D representations, followed by experiments with the spatially aware BEV. The findings, as shown in Tab. 5, indicate that our spatially aware BEV yields superior performance. The direct deployment of the 3D representation results in severe overfitting and, therefore, low performance. In addition, we also show that increasing the number of channels from 256 to 384 during the BEV projection can effectively improve the model performance, by compensating for information loss during projection. Overall, our spatially aware BEV strikes a balance that retains spatial acuity beyond basic BEV for resolving keypoint relationships while avoiding the complexity of full 3D convolutions.

	cp.	head	shoulders	elbows	wrists	hips	knees	ankles	all
3D	-	2.4620	2.4559	2.4492	2.4449	2.4394	2.4264	2.419	2.4422
V. BEV	-	0.0663	0.2810	0.4529	0.3802	0.1939	0.2148	0.2507	0.2614
Ours	✗	0.0688	0.0714	0.0982	0.1657	0.0723	0.1029	0.1595	0.1053
Ours	✓	0.0570	0.0669	0.0948	0.1467	0.0670	0.0820	0.1084	0.0887

Table 5. Ablation study for the spatially aware BEV module. *Cp.* denotes if to expand the number of channels to compensate for the information loss during the 2D projection. *V. BEV* denotes the vanilla BEV method.

5.4. Pose Estimation & Detection Trade-Off

Although the PEM metric accounts for penalties in both box and keypoint mismatches, to provide a clearer understanding of the performance, we report both detection metrics and pose estimation metrics in Table 6 under different NMS

thresholds. We reported the best MPJPE score in Tab. 1 at an NMS threshold of 0.3, while using a threshold of 0.1 improves detection performance with a slight sacrifice in MPJPE. Compared to the *VoxelNeXt* architecture, our method achieves similar detection performance (less than 1% decrease) at an NMS threshold of 0.1, while significantly improving MPJPE performance (approximately 35% increase). At an NMS threshold of 0.3, the MPJPE performance of *VoxelKP* can be further enhanced, although it results in a more substantial loss in detection performance.

Model	NMS Threshold	MPJPE	PEM	AP/L1	AP/L2	Recall@0.3	Recall@0.5
VoxelNeXt	0.1	0.1410	0.2120	0.7147	0.7096	0.9722	0.9375
VoxelKP	0.1	0.0908	0.1900	0.7083	0.7049	0.9658	0.9354
VoxelNeXt	0.3	0.1394	0.1973	0.6665	0.6586	0.8671	0.8404
VoxelKP	0.3	0.0887	0.1694	0.6060	0.5998	0.7816	0.7565

Table 6. Pose estimation and object detection performances under different NMS thresholds.

6. Conclusion

In this work, we identify the challenge of learning locally dense features within a sparse environment for human pose estimation. We proposed a new 3D fully sparse neural network for estimating dense human poses from point clouds. We present a comprehensive solution to the intricate challenges posed by spatial information of varying densities in the context of human pose estimation. Our method combines several novel components based on sparse operations to accurately and efficiently predict human body keypoints. Experiments on the Waymo dataset demonstrate the advantages of our approach compared to prior art and we demonstrate improved performance compared to other approaches trained on the same data as well as other approaches trained with additional data. While Waymo remains the sole publicly available dataset for these tasks, we eagerly anticipate the development of more datasets in the future. Through the identified challenge and the innovative framework, we pave the way for more nuanced and adaptable systems in this area.

Despite these advancements, we further identify certain areas for future exploration and improvement. As mentioned above, this work used a small volume of training data, but it could benefit from a larger-scale dataset. While we focus on single-frame point clouds, future work could leverage temporal information across sequences of LiDAR point clouds. Additionally, instead of the straightforward estimation of keypoints, future work may adopt inverse kinematics to include physical constraints on human body movement. Aside from refining estimated keypoint locations, this may especially be useful to handle real-world challenges such as occlusion within motion.

Acknowledgments

This work was supported by funding from King Abdullah University of Science and Technology (KAUST) - Center of Excellence for Generative AI, under award number 5940.

References

- [1] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 5
- [2] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2, 5, 7, 1
- [3] Zehui Chen, Zhenyu Li, Shiquan Zhang, Liangji Fang, Qin-hong Jiang, and Feng Zhao. Autoalignv2: Deformable feature aggregation for dynamic multi-modal 3d object detection. *ECCV*, 2022. 6
- [4] Spconv Contributors. Spconv: Spatially sparse convolution library. <https://github.com/traveller59/spconv>, 2022. 1
- [5] Yudi Dai, Yitai Lin, Xiping Lin, Chenglu Wen, Lan Xu, Hongwei Yi, Siqi Shen, Yuexin Ma, and Cheng Wang. Sloper4d: A scene-aware dataset for global 4d human pose estimation in urban environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 682–692, 2023. 6
- [6] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018. 2
- [7] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 4
- [8] Jordan SK Hu and Steven L Waslander. Pattern-aware data augmentation for lidar 3d object detection. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2703–2710. IEEE, 2021. 6
- [9] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. Afdetv2: Rethinking the necessity of the second stage for object detection from point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 969–979, 2022. 1
- [10] Sheng Jin, Lumin Xu, Jin Xu, Can Wang, Wentao Liu, Chen Qian, Wanli Ouyang, and Ping Luo. Whole-body human pose estimation in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2
- [11] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8500–8509, 2022. 4, 7
- [12] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3d recognition. In *CVPR*, 2023. 4
- [13] Alex H Lang, Sourabh Vora, Holger Caesar, Luning Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 2
- [14] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, pages 734–750, 2018. 1, 2
- [15] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 510–519, 2019. 4
- [16] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018. 2
- [17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 2
- [18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1
- [19] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, 2019. 4, 1
- [20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6
- [21] Xiaoxuan Ma, Jiajun Su, Chunyu Wang, Hai Ci, and Yizhou Wang. Context modeling in 3d human pose estimation: A unified perspective. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6238–6247, 2021. 2
- [22] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 922–928. IEEE, 2015. 2
- [23] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII 14*, pages 483–499. Springer, 2016. 1, 2
- [24] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2
- [25] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2

- [26] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 1
- [27] Shaoshuai Shi, Zhe Wang, Xiaogang Wang, and Hongsheng Li. Part-a² net: 3d part-aware and aggregation neural network for object detection from point cloud. *arXiv preprint arXiv:1907.03670*, 2(3), 2019.
- [28] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10529–10538, 2020. 1, 2, 4
- [29] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304. Ieee, 2011. 2
- [30] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018. 6
- [31] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, pages 369–386. SPIE, 2019. 6
- [32] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 1, 2
- [33] Pei Sun, Henrik Kretschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6
- [34] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. <https://github.com/open-mmlab/OpenPCDet>, 2020. 1
- [35] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. 2
- [36] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014. 1, 2
- [37] Zhenzhen Weng, Alexander S Gorban, Jingwei Ji, Mahyar Najibi, Yin Zhou, and Dragomir Anguelov. 3d human keypoints estimation from point clouds in the wild without human labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1158–1167, 2023. 1, 2, 6
- [38] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2, 5, 6
- [39] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. 2
- [40] Dongqiangzi Ye, Yufei Xie, Weijia Chen, Zixiang Zhou, and Hassan Foroosh. Lpformer: Lidar pose estimation transformer with multi-task network. *arXiv preprint arXiv:2306.12525*, 2023. 1, 2
- [41] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021. 2
- [42] Andrei Zanfir, Mihai Zanfir, Alex Gorban, Jingwei Ji, Yin Zhou, Dragomir Anguelov, and Cristian Sminchisescu. Hum3dil: Semi-supervised multi-modal 3d humanpose estimation for autonomous driving. In *Conference on Robot Learning*, pages 1114–1124. PMLR, 2023. 1, 2, 6
- [43] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. Pvt: Point-voxel transformer for point cloud learning. *International Journal of Intelligent Systems*, 37(12):11985–12008, 2022. 4, 1
- [44] Jingxiao Zheng, Xinwei Shi, Alexander Gorban, Junhua Mao, Yang Song, Charles R Qi, Ting Liu, Vishes Chari, Andre Comman, Yin Zhou, et al. Multi-modal 3d human pose estimation with 2d weak supervision in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4478–4487, 2022. 1, 2, 6, 7
- [45] Yin Zhou and Oncel Tuzel. Voxnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2
- [46] Yufan Zhou, Haiwei Dong, and Abdulmotaleb El Saddik. Learning to estimate 3d human pose from point cloud. *IEEE Sensors Journal*, 20(20):12334–12342, 2020. 2