LPFormer: LiDAR Pose Estimation Transformer with Multi-Task Network

Dongqiangzi Ye*   Yufei Xie*   Weijia Chen*   Zixiang Zhou[1]*   Hassan Foroosh[1]
[1]University of Central Florida

{eowinye, univerflyxie, weijia.chen619}@gmail.com
{zixiang.zhou, hassan.foroosh}@ucf.edu

Abstract

In this technical report, we present the 1st place solution for the 2023 Waymo Open Dataset Pose Estimation challenge. Due to the difficulty of acquiring large-scale 3D human keypoint annotation, previous methods have commonly relied on 2D image features and 2D sequential annotations for 3D human pose estimation. In contrast, our proposed method, named LPFormer, uses only LiDAR as its input along with its corresponding 3D annotations. LPFormer consists of two stages: the first stage detects the human bounding box and extracts multi-level feature representations, while the second stage employs a transformer-based network to regress the human keypoints using these features. Experimental results on the Waymo Open Dataset demonstrate the top performance, and improvements even compared to previous multi-modal solutions.

1. Introduction

Human pose estimation has gained significant popularity in the image and video domain due to its wide range of applications. However, pose estimation using 3D inputs, such as LiDAR point cloud, has received less attention due to the difficulty associated with acquiring accurate 3D annotations. As a result, previous methods [3, 16, 18] on LiDAR-based human pose estimation commonly rely on weakly-supervised approaches that utilize 2D annotations. These approaches often assume precise calibration between camera and LiDAR inputs. However, in real-world scenarios, small errors in annotations or calibration can propagate into significant errors in 3D space, thereby affecting the training of the network. Additionally, due to the differences in perspective, it is difficult to accurately recover important visibility information by simply lifting 2D annotations to the 3D space.

In image-based human pose estimation, the dominant approach is the top-down method [9], which involves first
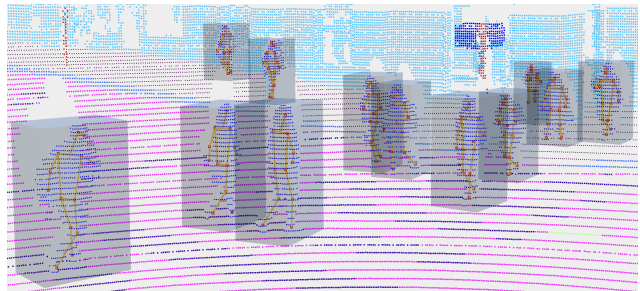

Figure 1. Our method can predict 3D keypoints (red points with yellow wireframes), 3D bounding boxes, and 3D semantic segmentation in a single framework.

detecting the human bounding box and then predicting the single-person pose based on the cropped features. However, a significant gap exists in the backbone network between the 2D detector and the 3D detector. Most LiDAR object detectors utilize projected bird's-eye view (BEV) features to detect objects, which helps reduce computational costs. This procedure leads to the loss of separable features in the height dimension that are crucial for human pose estimation. An effective use of learned object features for human pose estimation is still unexplored.

In this technical report, we present LPFormer, a complete two-stage top-down 3D human pose estimation framework that uses only LiDAR point cloud as input and is trained solely on 3D annotations. In the first stage, we adopt the design of LidarMultiNet [14], our previous state-of-the-art LiDAR multitask network that can accurately predict human object bounding boxes while generating fine-grained voxel features at a smaller scale. The second stage extracts point-level, voxel-level and object-level features of the points clouds inside each predicted bounding box and regresses the keypoints in a light-weighted transformer-based network. Our approach demonstrates that complex human pose estimation tasks can be seamlessly integrated into the LiDAR multi-task learning framework (as shown in Figure 1), achieving state-of-the-art performance without
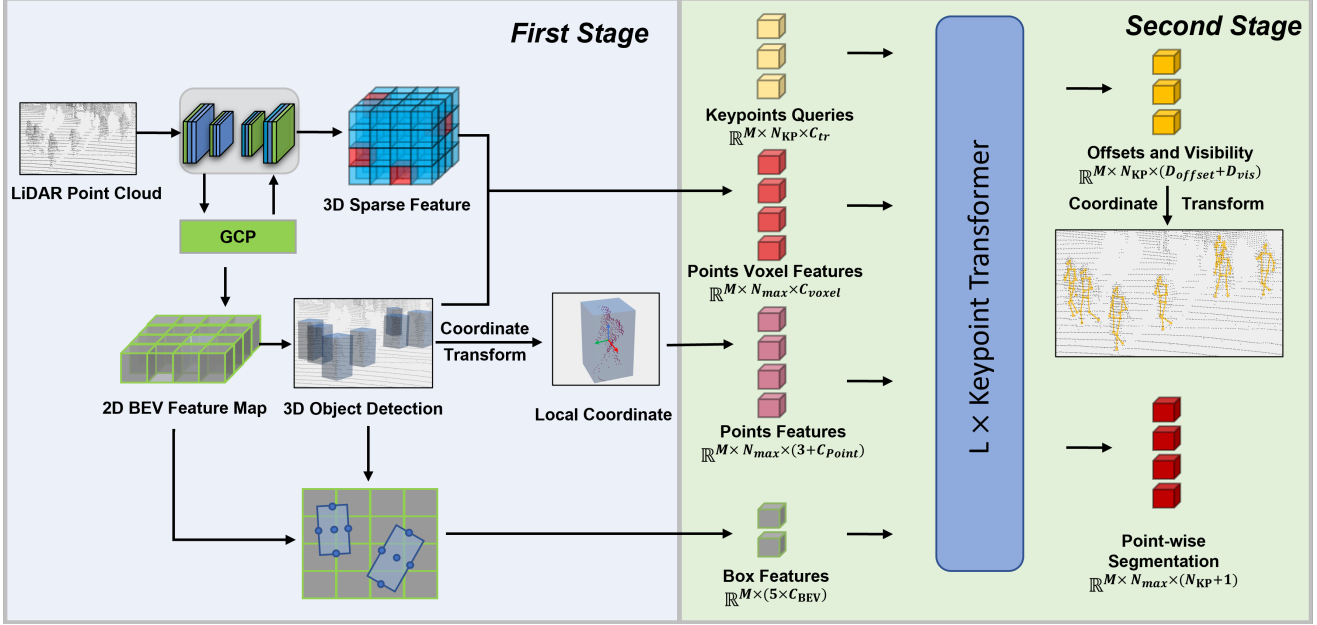
---

*Equal contribution

Figure 2. **Main Architecture of LPFormer.** Our network aims to estimate the 3D human pose for the entire frame based on the LiDAR-only input. It is comprised of two main components. The left part (blue) represents our powerful multi-task network, LidarMultiNet [14], which generates accurate 3D object detection and provides rich voxel and bird's-eye-view (BEV) features. The right part (green) corresponds to our Keypoint Transformer (KPTR), predicting the 3D keypoints of each human box using various inputs from our first-stage network.

the need for image features or annotations.

2. Related Work

Image-based 3D human pose estimation 3D human pose estimation (HPE) has been extensively studied based solely on camera images, where the human pose is represented as a parameter mesh model such as SMPL [5] or skeleton-based keypoints. Previous works in this field can be generally categorized into two main approaches: top-down [4, 9] or bottom-up methods [2, 11]. Top-down methods decouple the pose estimation problem to individual human detection using the off-the-shelf object detection network and single-person pose estimation on the cropped object region. On the contrary, bottom-up methods first estimate the instance-agnostic keypoints and then group them together [2] or directly regress the joint parameter using center-based feature representation [11]. Some recent works [8, 13] explored using the transformer decoder to estimate human pose in an end-to-end fashion following the set matching design in DETR [1]. However, image-based 3D HPE suffers from inaccuracies and is considered not applicable to larger-scale outdoor scenes due to frequent occlusion and the difficulty of depth estimation.

LiDAR-based 3D human pose estimation To solve the depth ambiguity problem, some researchers [12, 19] ex-

plored using depth images for the 3D HPE. Compared to the depth image, LiDAR point cloud has a larger range and is particularly applicable to outdoor scenes, such as in autonomous driving applications. Waymo recently released the human joint keypoint annotations on both associated 2D images and 3D LiDAR point cloud on Waymo Open Dataset [10]. However, due to the lack of enough 3D annotation, previous works [16, 18] have focused on semi-supervised learning approaches. These approaches lift 2D annotation to the 3D space and rely on the fusion of image and LiDAR features for the HPE task.

3. Method

LPFormer is a two-stage LiDAR-only model designed for 3D pose estimation. Figure 2 provides an overview of our framework. The input to LPFormer only consists of point clouds, represented as a set of LiDAR points $P = \{p_i | p_i \in \mathbb{R}^{3+C_{point}}\}_{i=1}^{N}$, where N denotes the number of points and C_{point} includes additional features such as intensity, elongation, and timestamp for each point. In the first stage, we employ a powerful multi-task network [14] that accurately predicts 3D object detection and 3D semantic segmentation, incorporating meaningful semantic features. Inspired by a recent work [16], our second stage leverages a transformer-based model. This model takes various out-
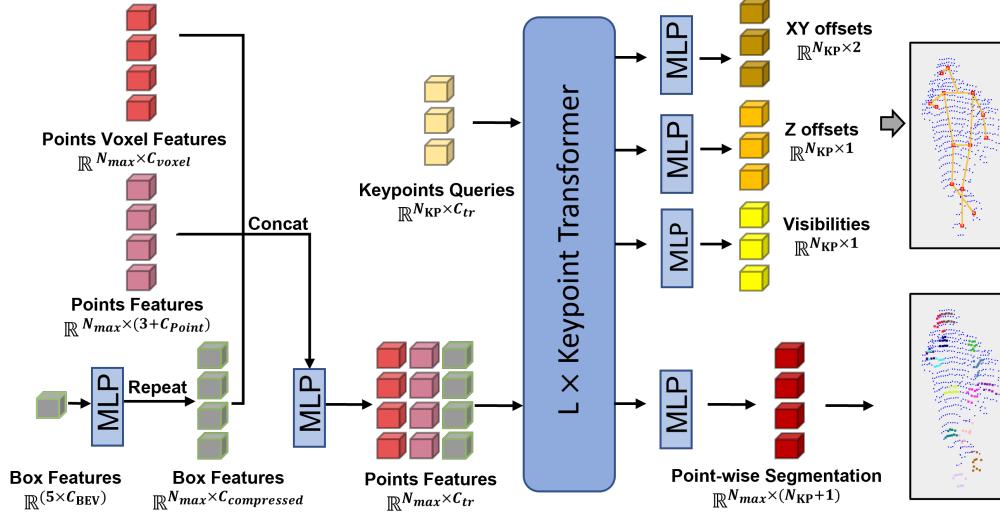
Figure 3. **Illustration of Keypoint Transformer (KPTR).** In the initial stage of our KPTR, we start by compressing the feature dimension of the box features. These compressed box features are then repeated and concatenated with the point features and point voxel features. The keypoint queries are generated from learnable embedding features. Then L sequences of KPTR operations are performed on the keypoint queries and point tokens. Finally, the keypoint queries are passed through three distinct MLPs to learn the XY offsets, Z offsets, and visibilities of the 3D keypoints. Simultaneously, the point tokens are processed by an MLP to learn the point-wise segmentation labels for the 3D keypoints, which serves as an auxiliary task.

puts from the first stage as inputs and generates 3D human keypoints $Y_{kp} \in \mathbb{R}^{N_{kp} \times 3}$ along with their corresponding visibilities $Y_{vis} \in \mathbb{R}^{N_{kp}}$, where N_{kp} is the number of 3D keypoints.

3.1. First Stage Detection

The first stage of our LPFormer adopts the methodology of LidarMultiNet [14] for extracting point clouds features from raw point clouds P. Illustrated in Figure 2, it consists of a 3D encoder-decoder structure with Global Context Pooling (GCP) module in between. The 3D object detection predictions are obtained through the 3D detection head, which is attached to the dense 2D BEV feature map.

Enriching point features with multi-level feature embedding Within each detected bounding box, the points are performed by a local coordinate transformation involving translation and rotation. Subsequently, the transformed points are concatenated with their corresponding original point features, resulting in $P_{point} \in \mathbb{R}^{M \times N_{max} \times (3 + C_{point})}$, where M is the number of bounding boxes and N_{max} represents the maximum number of point clouds within each bounding box. For each box, we randomly shuffle and remove extra points, and pad with zero if the number of points within a box is less than N_{max}. Additionally, we generate point voxel features $P_{voxel} \in \mathbb{R}^{M \times N_{max} \times C_{voxel}}$ by gathering the 3D sparse features from the decoder using their corresponding voxelization index, where C_{voxel} denotes the channel size of the last stage of the decoder. Similar to Cen-

terPoint [15], for each bounding box, we adopt the BEV features at its center as well as the centers of its edges in the 2D BEV feature map as the box features $B \in \mathbb{R}^{M \times (5 \times C_{BEV})}$

3.2. Second Stage Keypoint Transformer

By leveraging the capabilities of the robust first stage model LidarMultiNet [14], our second stage is able to exploit valuable semantic features for capturing intricate object details, including human 3D pose. Different from LidarMultiNet [14], we choose a transformer architecture instead of a PointNet-like [7] structure as our second stage, in order to effectively understand 3D keypoints by leveraging local points information through an attention mechanism. The details of our second stage are shown in Figure 3.

Specifically, our second stage takes various features from local points features P_{point}, semantic voxel-wise points features P_{voxel}, and box-wise features B to predict 3D keypoints for each pedestrian or cyclist box. Starting with a box-wise feature B, we first employ a multilayer perceptron (MLP) to compress its dimensions from $\mathbb{R}^{5 \times C_{BEV}}$ to $\mathbb{R}^{C_{compressed}}$. This compressed box-wise feature is then replicated as $P_{box} \in \mathbb{R}^{N_{max} \times C_{compressed}}$ and combined with point-wise features P_{point} and P_{voxel}, resulting in $P_{cat} \in \mathbb{R}^{N_{max} \times (3 + C_{point} + C_{voxel} + C_{compressed})}$. The fused point-wise features are subjected to a simple matrix multiplication, yielding $X_{point} \in \mathbb{R}^{N_{max} \times C_{tr}}$, which serves as one part of the input for Keypoint Transformer (KPTR). The other input for KPTR is a learnable 3D keypoints query

$X_{kp} \in \mathbb{R}^{N_{kp} \times C_{tr}}$. Subsequently, we employ KPTR, which consists of L blocks of a multi-head self-attention and a feed-forward network, to learn internal features X'_{point} and X'_{kp}. Finally, the keypoints' internal features X'_{kp} are fed into three separate MLPs to predict 3D keypoints offsets along the X and Y axes $\hat{Y}_{xy} \in \mathbb{R}^{N_{kp} \times 2}$, 3D keypoints offsets along the Z axis $\hat{Y}_z \in \mathbb{R}^{N_{kp} \times 1}$, and 3D keypoints visibilities $\hat{Y}_{vis} \in \mathbb{R}^{N_{kp}}$. Furthermore, the point-wise internal features X'_{point} are processed by an MLP to estimate point-wise keypoint segmentation $\hat{Y}_{kpseg} \in \mathbb{R}^{N_{max} \times (N_{kp}+1)}$.

For the final predictions, we combine the predicted 3D keypoints offsets $\hat{Y}_{xy}$, $\hat{Y}_z$, and the predicted 3D keypoints visibilities $\hat{Y}_{vis}$ to generate the human pose for each bounding box. Then we apply a reverse coordinate transformation to convert the predicted human pose from the local coordinate system to the global LiDAR coordinate system. Moreover, the predicted point-wise keypoint segmentation $\hat{Y}_{kpseg}$ serves as an auxiliary task, aiding KPTR in learning point-wise local information and enhancing the regression of 3D keypoints through the attention mechanism. In the experiments section, we will demonstrate how this auxiliary task significantly enhances the overall performance of the model.

3.3. Training and Losses

During the training phase, we replace the predicted bounding boxes with ground truth bounding boxes that include 3D keypoints labels. This substitution is necessary since only a limited number of ground truth boxes are annotated with 3D keypoints labels. By employing this approach, we simplify and expedite the training process. Additionally, inspired by [18], we introduce a point-wise segmentation task for keypoints as an auxiliary task to improve the performance of 3D keypoints regression. The pseudo segmentation labels $Y_{kpseg} \in \mathbb{R}^{N_{max} \times (N_{kp}+1)}$ are generated by assigning each 3D keypoint's type to its top K nearest points. This auxiliary task is supervised using cross-entropy loss, expressed as $\mathcal{L}_{kpseg}$.

To facilitate the 3D keypoints regression, we divide it into two branches: one for the regression over the X and Y axes and another for the regression over the Z axis. This division is based on our observation that predicting the offset along the Z axis is comparatively easier than predicting it along the X and Y axes. We employ smooth L1 loss to supervise these regression branches, denoting them as $\mathcal{L}_{xy}$ and $\mathcal{L}_z$. Note that only the visible 3D keypoints contribute to the regression losses. In addition, we treat the visibility of the keypoints as a binary classification problem. We supervise it using binary cross-entropy loss as $\mathcal{L}_{vis}$.

Our first stage LiDARMultiNet is pretrained following instructions in [14] and frozen during the 3D keypoints' training phase. We introduce weight factors for each loss component, and our final loss function is formulated as follows:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{xy} + \lambda_2 \mathcal{L}_z + \lambda_3 \mathcal{L}_{vis} + \lambda_4 \mathcal{L}_{kpseg} \quad (1)$$

where λ_1, λ_2, λ_3, λ_4 are weight factors and fixed at values of 5, 1, 1, and 1, respectively.

4. Experiments

4.1. Dataset

Waymo Open Dataset released the human keypoint annotation on the v1.3.2 dataset that contains LiDAR range images and associated camera images. We use v1.4.2 for training and validation. The 14 classes of keypoints for evaluation are defined as nose, left shoulder, left elbow, left wrist, left hip, left knee, left ankle, right shoulder, right elbow, right wrist, right hip, right knee, right ankle, and head center. There are 144709 objects with 2D keypoints annotations while only 8125 objects with 3D keypoints annotations in the training dataset.

4.2. Metrics

We use mean per-joint position error (MPJPE) and Pose Estimation Metric (PEM) as the metrics to evaluate our method. In MPJPE, the visibility of predicted joint i of one human keypoint set j is represented by $v_i^j \in [0, 1]$, indicating whether there is a ground truth for it. As such, the MPJPE over the whole dataset is:

$$\textbf{MPJPE}(Y, \hat{Y}) = \frac{1}{\sum_{i,j} v_i^j} \sum_{i,j} v_i^j ||Y_i^j - \hat{Y}_i^j||_2, \quad (2)$$

where Y and $\hat{Y}$ are the ground truth and predicted 3D coordinates of keypoints.

PEM is a new metric created specifically for the Pose Estimation challenge. Besides keypoint localization error and visibility classification accuracy, it is also sensitive to the rates of false positive and negative object detections, while remaining insensitive to the Intersection over Union (IoU) of object detections. PEM is calculated as a weighted sum of the MPJPE over visible matched keypoints and a penalty for unmatched keypoints, as shown:

$$\textbf{PEM}(Y, \hat{Y}) = \frac{\sum_{i \in M} ||y_i - \hat{y}_i||_2 + C|U|}{|M| + |U|}, \quad (3)$$

where M is the set of indices of matched keypoints, U is the set of indices of unmatched keypoints, and $C = 0.25$ is a constant penalty for unmatched keypoints. The PEM ensures accurate, robust ranking of model performance in a competition setting.

Table 1. PEM and MPJPE results on the `test` split of WOD.

Model	shoulders		elbows		wrists		hips		knees		ankles		head		all	
	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE	PEM	MPJPE
baseline	0.2323	0.1894	0.2354	0.2083	0.2391	0.2240	0.2334	0.1807	0.2327	0.1934	0.2345	0.2250	0.2376	0.1984	0.2349	0.2022
KTD	0.2261	0.1876	0.2301	0.2065	0.2349	0.2227	0.2276	0.1790	0.2267	0.1919	0.2290	0.2237	0.2328	0.1973	0.2295	0.2007
LPFormer	0.1428	0.0462	0.1511	0.0578	0.1771	0.0951	0.1519	0.0562	0.1477	0.0578	0.1479	0.0663	0.1544	0.0443	0.1524	0.0594

4.3. Implementation Details

Throughout all our experiments, we use a pretrained Li-darMultiNet [14] as the first stage of our framework, which remains frozen during the training phase of the second stage. For additional network and training specifics regarding our first stage, please refer to LidarMultiNet [14].

Regarding KPTR, the dimensions of the inputs, namely C_{point}, C_{voxel}, and C_{BEV}, are set to 3, 32, and 512, respectively. The size of the compressed features, denoted as $C_{compressed}$, is 32. We cap the maximum number of points per bounding box at 1024. For the transformer architecture, similar to the recent work [16], we utilize $L = 4$ stages, an embedding size of $C_{tr} = 256$, a feed-forward network with internal channels of 256, and 8 heads for the MultiHeadAttention layer. The total number of 3D keypoints N_{kp} is 14.

During training, we incorporate various data augmentations, including standard random flipping, global scaling, rotation, and translation. It is important to note that flipping the point clouds has an impact on the relationships between the 3D keypoints annotations, similar to the mirror effect. When performing a flip over the X-axis or Y-axis, the left parts of the 3D keypoints should be exchanged with the right parts of the 3D keypoints accordingly.

To train our model, we use the AdamW optimizer along with the one-cycle learning rate scheduler for a total of 20 epochs. The training process utilizes a maximum learning rate of 3e-3, a weight decay of 0.01, and a momentum ranging from 0.85 to 0.95. All experiments are conducted on 8 Nvidia A100 GPUs, with a batch size set to 16.

4.4. Main Pose Estimation Results

In our final submission to the leaderboard, we trained our model using the combined dataset of Waymo's training and validation splits. The results, presented in Table 1, demonstrate the impressive performance of our LPFormer, achieving a PEM of 0.1524, an MPJPE of 0.0594, and ranking 1st on the leaderboard. Notably, our LPFormer outperforms all other methods across all categories in terms of both PEM and MPJPE.

4.5. Ablation Study

To conduct a comprehensive performance analysis of our LPFormer, we compare it with other SOTA methods, as shown in Table 2. It is important to note that all previous methods were evaluated on a subset of the WOD validation split. Additionally, these methods simplify the problem by

Table 2. The comparison on the WOD `val` split. ∗: reported by [16], where the result is tested on randomly selected 50% of subjects from the WOD `val` split. "L", "CL" denote LiDAR-only, camera & LiDAR fusion methods.

Method	Modal	gt box	MPJPE(cm)↓
ContextPose [6] ∗	C	✓	10.82
Multi-modal [18]	CL	✓	10.32
THUNDR [17] ∗	C	✓	9.62
THUNDR [17] w/ depth ∗	CL	✓	9.20
HUM3DIL [16] ∗	CL	✓	6.72
LPFormer	L		**6.16**

Table 3. The ablation of improvement of each component on the WOD `val` split.

Baseline	2nd	seg aux	transformer	box feat	PEM↓	MPJPE↓
✓					0.1908	0.1801
✓	✓				0.1176	0.0865
✓	✓	✓			0.1149	0.083
✓	✓	✓	✓		0.1044	0.0703
✓	✓	✓	✓	✓	**0.0976**	**0.0616**

providing ground truth 3D bounding boxes along with associated ground truth 2D bounding boxes as inputs. Despite some of these methods incorporating camera and LiDAR fusion or 2D weakly supervision, our LPFormer outperforms them all in terms of MPJPE, achieving an impressive MPJPE of 6.16cm.

Table 3 shows a comparison of the performance between the first stage and LPFormer, as well as the contribution of each component in the second stage to the overall performance. The first stage results are directly output from the Center Head following the BEV feature map. Given the BEV feature map is primarily driven by the detection task and has low resolution, it lacks fine-grained features, resulting in mediocre performance. The second stage which is similar to Second-stage Refinement module in LidaMulti-Net [14], however, significantly improves upon introducing point-wise fine-grained features. Further enhancements are achieved by adding the keypoint segmentation auxiliary task, employing the transformer structure, and incorporating box features, all of which contribute to varying degrees of performance improvement for the model.

4.6. Visualization

Figure 1 shows the output predictions of our model for one frame in the validation set, viewed from a particular
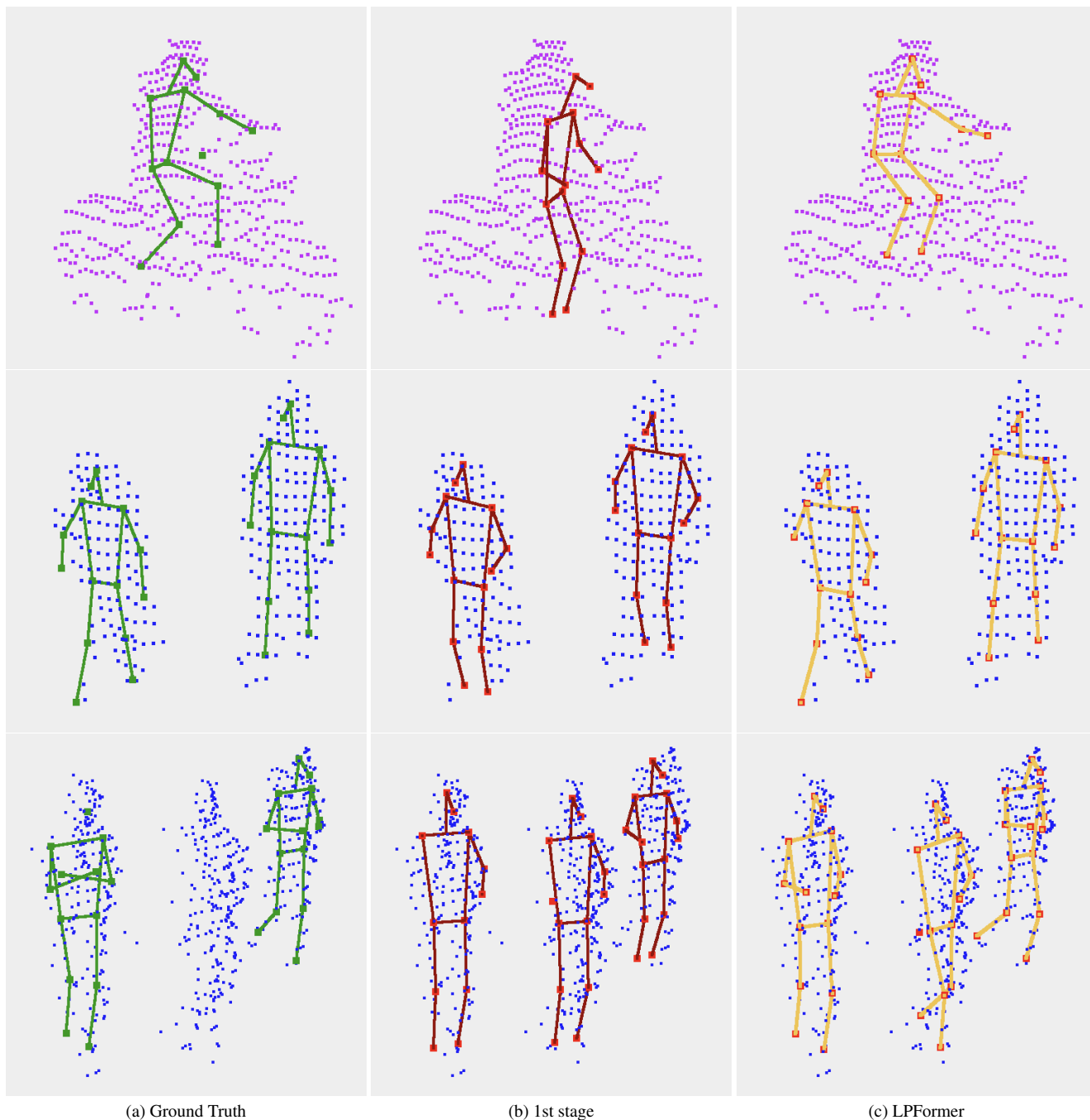
Figure 4. Prediction results compared to the Ground Truth and the 1st stage results.

angle. The input is solely 3D LiDAR point clouds. Remarkably, our network simultaneously outputs results of 3D semantic segmentation, 3D bounding boxes, as well as their 3D keypoints (red) along with the corresponding wireframes (yellow) for visualization. Our model also predicts visibility, for example, the left knee of the second person from the left is predicted as invisible, while the left foot is visible. Both feet of the third person from the right are predicted as invisible. The right elbow of the sixth person from the right is predicted as invisible, however, the right hand is visible.

Figure 4 presents a selection of predictions made on the validation set. From left to right, the three columns represent ground truths, the predictions of the 1st stage, and

the predictions of LPFormer, respectively. Each row showcases the same group of objects. As can be observed, across all three groups, the performance of LPFormer noticeably surpasses that of the 1st stage output. The first row highlights a cyclist for whom ground truth annotations are extremely limited. Despite the limited amount of annotations, LPFormer still manages to deliver meaningful output. In the second row, LPFormer is strikingly close to the ground truth, with the exception of an FN (False Negative) visibility for the right hand of the pedestrian on the left. The third row demonstrates that even on the pedestrian without ground truth annotations, LPFormer still produces satisfactory results. For the running pedestrian on the right, LPFormer performs pretty well. However, the left pedestrian's head center is an FP (False Positive) case, and the crossed hands pose is a difficult case given the small amount of similar ground truth annotations available.

Figure 5 demonstrates the model's performance in pedestrian-rich scenarios, as the PEM metric is sensitive to both false positive and false negative object detections. In these scenarios, the restriction on a 25m detection range has been eliminated, while the detection score threshold and IoU threshold have been maintained. It is evident that the model can detect more distant pedestrians and provide keypoints predictions. However, it is noted that the visibility for distant pedestrians decreases, which is reasonable as the point clouds in the distance tend to be more sparse and prone to occlusion.

5. Conclusion

In the Waymo Open Dataset pose estimation challenge 2023, our proposed LPFormer secured the 1st place. As for the future work, we plan to further enhance our LPFormer method through broad integration and fusion of LiDAR and camera data, in addition to exploiting 2D weak supervision.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[2] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *CVPR*, 2020. 2

[3] Peishan Cong, Yiteng Xu, Yiming Ren, Juze Zhang, Lan Xu, Jingya Wang, Jingyi Yu, and Yuexin Ma. Weakly supervised 3d multi-person pose estimation for large-scale scenes based on monocular camera and single lidar. *arXiv preprint arXiv:2211.16951*, 2022. 1

[4] Zhihao Li, Jianzhuang Liu, Zhensong Zhang, Songcen Xu, and Youliang Yan. Cliff: Carrying location information in full frames into human pose and shape estimation. In *ECCV*, 2022. 2

[5] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 2015. 2

[6] Xiaoxuan Ma, Jiajun Su, Chunyu Wang, Hai Ci, and Yizhou Wang. Context modeling in 3d human pose estimation: A unified perspective. In *CVPR*, 2021. 5

[7] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 3

[8] Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. In *CVPR*, 2022. 2

[9] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 1, 2

[10] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 2

[11] Yu Sun, Qian Bao, Wu Liu, Yili Fu, Michael J Black, and Tao Mei. Monocular, one-stage, regression of multiple 3d people. In *ICCV*, 2021. 2

[12] Fu Xiong, Boshen Zhang, Yang Xiao, Zhiguo Cao, Taidong Yu, Joey Tianyi Zhou, and Junsong Yuan. A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image. In *ICCV*, 2019. 2

[13] Jie Yang, Ailing Zeng, Shilong Liu, Feng Li, Ruimao Zhang, and Lei Zhang. Explicit box detection unifies end-to-end multi-person pose estimation. In *ICLR*, 2023. 2

[14] Dongqiangzi Ye, Zixiang Zhou, Weijia Chen, Yufei Xie, Yu Wang, Panqu Wang, and Hassan Foroosh. Lidarmultinet: Towards a unified multi-task network for lidar perception. In *AAAI*, 2023. 1, 2, 3, 4, 5

[15] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 3

[16] 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 *CoRL*, 2022. 1, 2, 5

[17] Mihai Zanfir, Andrei Zanfir, Eduard Gabriel Bazavan, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Thundr: Transformer-based 3d human reconstruction with markers. In *ICCV*, 2021. 5

[18] Jingxiao Zheng, Xinwei Shi, Alexander Gorban, Junhua Mao, Yang Song, Charles R Qi, Ting Liu, Visesh Chari, Andre Cornman, Yin Zhou, et al. Multi-modal 3d human pose estimation with 2d weak supervision in autonomous driving. In *CVPRW*, 2022. 1, 2, 4, 5

[19] Christian Zimmermann, Tim Welschehold, Christian Dornhege, Wolfram Burgard, and Thomas Brox. 3d human pose estimation in rgbd images for robotic task learning. In *ICRA*, 2018. 2
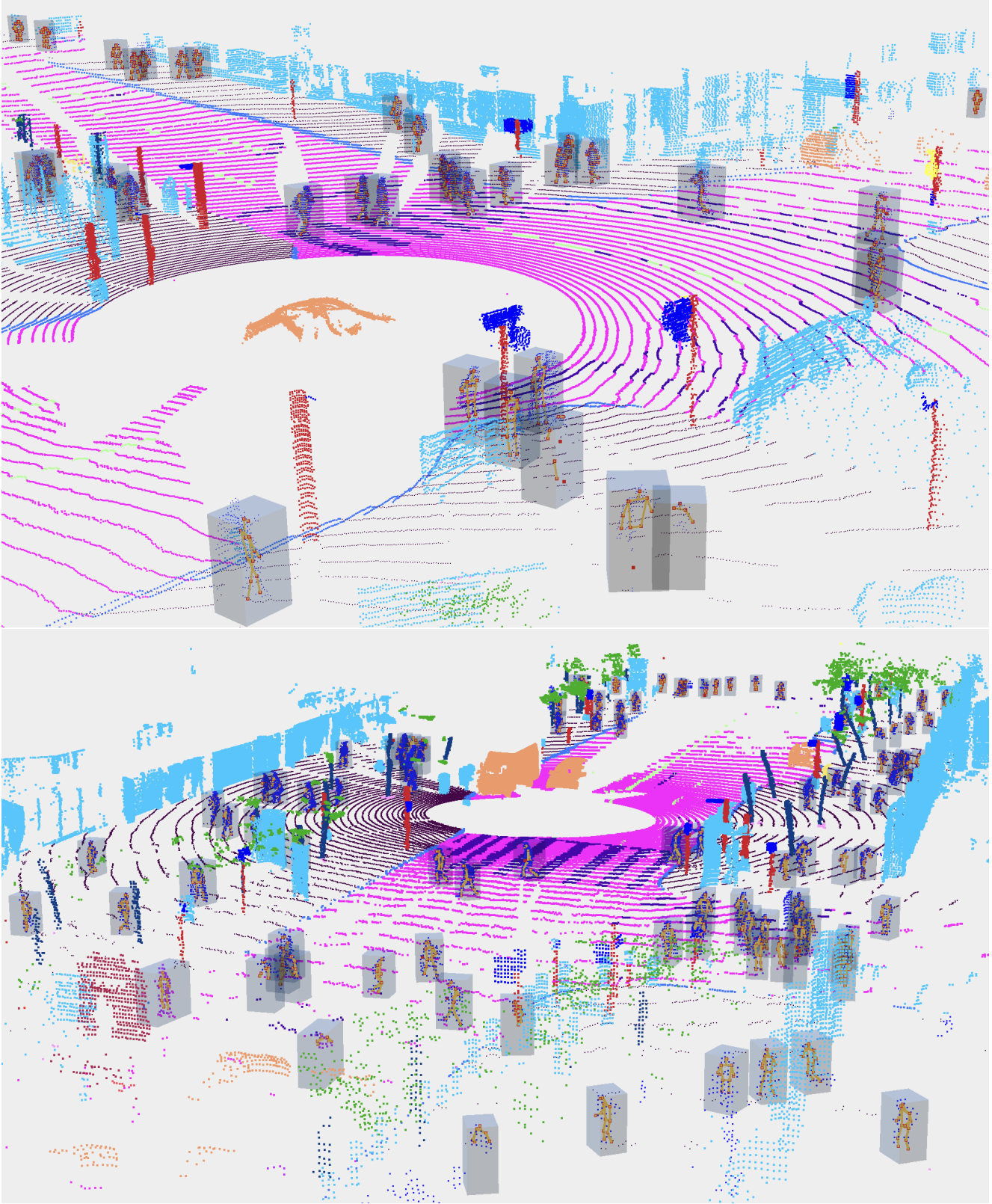
Figure 5. Prediction results on the whole scene with a significant number of pedestrians in the validation set.