

# MobileOcc: A Human-Aware Semantic Occupancy Dataset for Mobile Robots

Junseo Kim\* Guido Dumont\* Xinyu Gao\* Gang Chen\* Holger Caesar Javier Alonso-Mora  
 Delft University of Technology  
 Mekelweg 5, 2628 CD Delft, Netherlands

{J.Kim-18, G.Dumont, X.Gao-14}@student.tudelft.nl, {G.Chen-5, H.Caesar, J.AlonsoMora}@tudelft.nl

## Abstract

Dense 3D semantic occupancy perception is critical for mobile robots operating in pedestrian-rich environments, yet it remains underexplored compared to its application in autonomous driving. To address this gap, we present MobileOcc, a semantic occupancy dataset for mobile robots operating in crowded human environments. Our dataset is built using an annotation pipeline that incorporates static object occupancy annotations and a novel mesh optimization framework explicitly designed for human occupancy modeling. It reconstructs deformable human geometry from 2D images and subsequently refines and optimizes it using associated LiDAR point data. Using MobileOcc, we establish benchmarks for two tasks, i) Occupancy prediction and ii) Pedestrian velocity prediction, using different methods including monocular, stereo, and panoptic occupancy, with metrics and baseline implementations for reproducible comparison. Beyond occupancy prediction, we further assess our annotation method on 3D human pose estimation datasets. Results demonstrate that our method exhibits robust performance across different datasets.

## 1. Introduction

Semantic occupancy prediction [20, 28, 36, 47, 50] has emerged as a key perception component in modern navigation systems. It predicts a 3D field representing free, occupied, and unknown space, along with semantic labels, directly from image inputs. This enables the construction of a differentiable perception model that can be seamlessly integrated into downstream planning tasks. However, existing datasets [29, 46, 49] for training semantic occupancy prediction models are primarily developed for autonomous driving scenarios and focus on rigid objects such as roads, buildings, and vehicles. In contrast, non-rigid objects, such as humans, appear less frequently and are typically modeled by aggregating multi-view points into rigid bodies, neglect-

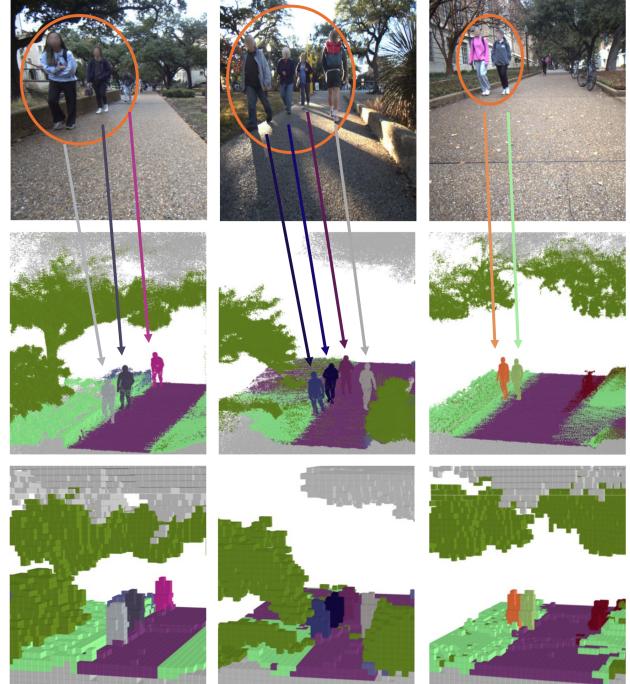


Figure 1. Qualitative results across occupancy resolutions. Top: input image from UT Campus Object Dataset (CODa) [57]. Middle: our semantic occupancy label at fine resolution (0.02 m). Bottom: semantic occupancy at coarse resolution (0.2 m). Gray voxels represent unknown regions, while free space is not visualized for clarity.

ing their deformable nature. For mobile robots, however, humans are among the most frequently encountered dynamic entities during navigation. Therefore, a semantic occupancy prediction dataset designed for human-populated environments is still missing.

In this paper, we present MobileOcc, a new semantic occupancy dataset designed for human-populated environments. The dataset is annotated using a proposed pipeline that fuses synchronized image and LiDAR data to generate occupancy labels, with a particular focus on accurate

\*Equal contribution.

human representation. The pipeline produces occupancy annotations for static objects, as well as for free and unknown space, by fusing accumulated and filtered semantic LiDAR points within an OctoMap framework [18]. For human occupancy modeling, we introduce a mesh optimization method that first detects humans in images to estimate their initial pose and mesh, then optimizes by fusing corresponding LiDAR points. Compared with image-only human mesh recovery (HMR) methods, which suffer from depth and scale ambiguities, and LiDAR-only approaches, which are limited by sparse points, missed detections, and pose uncertainty in cluttered or distant scenes [14, 27, 53], our fusion-based method produces robust and accurate human meshes by leveraging the strengths of both modalities.

The contributions of this paper are as follows:

- We introduce MobileOcc, a 3D semantic occupancy prediction dataset built on raw sensor data from UT Campus Object Dataset (CODa) [57] for mobile robots navigating among near-field pedestrians.
- We develop an annotation pipeline that generates occupancy annotations from synchronized RGB images and dense LiDAR data. In particular, the pipeline leverages both image information and LiDAR point clouds to produce accurate human occupancy representations.
- We establish MobileOcc as a benchmark suite by evaluating existing occupancy frameworks on two tasks: i) semantic occupancy prediction and ii) pedestrian velocity prediction, with baselines across monocular, stereo, and panoptic occupancy methods.

## 2. Related Work

### 2.1. 3D Occupancy Prediction Datasets

3D occupancy prediction plays an important role in navigation tasks and is widely used in autonomous driving. SemanticKITTI [1] first introduced an occupancy perception dataset based on the KITTI dataset, but it does not account for moving objects and has limited scale and diversity. Subsequent datasets such as Occ3D [46], OpenOccupancy [49], and SSCBench [29] provide larger-scale, densely annotated voxel labels on nuScenes [4], Waymo [43], and KITTI-360 [1]. In off-road settings, WildOcc [56] extends RELLIS-3D [22] with dense semantic occupancy for unstructured environments. These newer datasets typically model moving vehicles as rigid bodies, accumulating points for each vehicle using pre-labeled 3D bounding box poses. However, humans are non-rigid and are not well represented in these datasets (as shown in Figure 1). For mobile robots, humans are among the most common dynamic objects encountered in navigation tasks, and a dataset capable of accurately modeling human motion is still needed.

### 2.2. 3D Occupancy Prediction Methods

Unlike traditional filtering-based methods that fuse 3D points to generate an occupancy map [7], occupancy prediction typically learns to map 2D images from a single or multiple cameras to voxel-level semantic occupancy [47]. The image itself contains only 2D information; therefore, lifting 2D features to 3D is crucial for occupancy prediction methods. Camera-only methods such as VoxFormer [28] and OcDepth [36] first infer depth or geometry cues and then decode a dense 3D volume. With surround-view input, TPV-Former [20], SurroundOcc [50], and OccNet/OpenOcc [47] aggregate multi-camera features before or during lifting to improve coverage in long-range and occluded regions. Building on these ideas, FB-OCC [31] and BEVDet4D [19] combine voxel and BEV representations and exploit temporal cues to obtain strong camera-only baselines for dynamic scenes. FlashOcc [54] and Panoptic-FlashOcc [55] further emphasize efficiency and panoptic consistency by keeping features in BEV and lifting logits to 3D. Our setting is complementary: we focus on mobile robots in human-rich environments and evaluate such occupancy frameworks under non-rigid human motion.

### 2.3. Image-Based Human Mesh Recovery

Estimating a full 3D human body mesh from a single RGB image has been widely studied using parametric models. Early regression-based methods like HMR [23] directly predict SMPL [35] shape and pose parameters from an image, regularized by learned priors. Subsequent works improve accuracy with stronger priors and training schemes, as in SPIN [25] and CLIFF [30], but still suffer from depth ambiguity and limited geometric detail. Alternative designs lift skeletons or lixel-based heatmaps to meshes [9, 37], or use transformer regressors [32, 33]. Implicit-surface approaches [40, 51] learn continuous occupancy or signed-distance fields to represent detailed shapes, yet purely image-based pipelines remain sensitive to missing depth. To reduce this depth uncertainty, several methods incorporate LiDAR as input. LiDARCap [27, 58] and LiDAR-HMR [14] fit SMPL-based meshes to long-range LiDAR, while S3 [53] models human shape, pose, and skinning via implicit fields. However, LiDAR-only methods can be brittle when human points overlap with nearby objects or become sparse at long distances, leading to missed detections and ambiguous poses.

We present a method that first obtains an initial 3D human pose and mesh from an image, then refines it by fusing LiDAR data via a novel optimization framework. This training-free approach leverages both modalities and generalizes directly across datasets.

### 3. MobileOcc Dataset Building

Our dataset is constructed by annotating the image and LiDAR data from the CODa dataset [57] using the proposed annotation pipeline. This section introduces the pipeline by first providing a general overview of the workflow and then detailing its key components.

#### 3.1. Annotation Pipeline

Our general annotation pipeline is shown in Figure 2. Given synchronized image streams and LiDAR along a robot trajectory, we apply a comprehensive annotation pipeline to capture both dynamic and static scene elements. Several data preprocessing steps are used before getting the occupancy annotations. We first run OC-SORT [5] with YOLOX [15] detection on monocular camera footage to obtain tracked 2D bounding boxes with IDs for all pedestrians across frames. For each tracked person, we estimate 2D pose with keypoint confidence using VITPose [52]. We then obtain fine instance masks via Mask2Former [8] and associate each mask with the LiDAR point cloud to get points for each person. Finally, we apply semantic segmentation to assign Cityscapes [10] class labels, providing category priors for static background elements (road, vegetation, sky, etc.) and distinguishing dynamic objects. All preprocessed data are time-stamped and co-referenced across modalities to enable proper separation of dynamic pedestrians from static background before voxelization. With these preprocessing steps, we perform human mesh optimization (Section 3.2), and static mapping (Section 3.3), then fuse them into a 3D occupancy representation (Section 3.4).

#### 3.2. Human Mesh Optimization

We propose an optimization-based framework that fuses LiDAR points with monocular images to improve 3D human mesh estimation. Our approach first obtains an initial parametric mesh (SMPL) from the image [35], then refines it in three stages: i) filtering out occluded body parts using 2D keypoint cues, ii) performing a coarse mesh alignment to LiDAR via iterative closest point (ICP), and iii) optimizing the SMPL pose/shape parameters to fit the image and LiDAR observations jointly.

We use the following notation throughout the paper:

- **SMPL parameters:** pose  $\theta = \{\theta_{\text{glob}}, \theta_{\text{body}}\}$ , shape  $\beta$ , and camera translation  $\mathbf{t}_{\text{cam}} \in \mathbb{R}^3$ .
- **Camera:** intrinsics  $\mathbf{K}$ .
- **Mesh:**  $\mathcal{M}$  with vertex matrix  $\mathbf{M}(\beta, \theta) \in \mathbb{R}^{6890 \times 3}$ ; visible subset  $\mathcal{V} \subset \mathcal{M}$  with  $n = |\mathcal{V}|$ .
- **LiDAR:** point cloud  $\mathcal{P} = \{\mathbf{p}_j\}_{j=1}^m$  with  $m = |\mathcal{P}|$ .
- **Operators:** projection  $\Pi_{\mathbf{K}}(\cdot)$ , global rotation  $\mathbf{R}(\theta)$ , and SMPL joints  $\mathbf{J}_i(\beta)$ .

##### 3.2.1. Initial SMPL Prediction

We first obtain an initial estimate of the SMPL parameters with CLIFF [30], a top-down HMR regressor (detects a person, then regresses SMPL parameters from the person crop while conditioning on the box’s full image location). CLIFF produces an initial mesh  $\mathcal{M}_0$  with shape parameters  $\beta_0$ , pose parameters  $\theta_0$ , and a global translation  $\mathbf{t}_0$  by extending a ResNet-based HMR network [23] to incorporate the person’s bounding box location in the image and to compute loss in uncropped image coordinates. This provides a reasonable starting point for the person’s pose and location in the camera frame. However, like all monocular methods, it still suffers from pose and shape errors due to depth ambiguity, which our subsequent steps correct.

##### 3.2.2. LiDAR-Mesh Alignment

Before optimizing the human mesh provided by CLIFF, we first perform visibility filtering (details in the supplementary material) to identify the visible mesh vertices from LiDAR. We then coarsely register the visible mesh vertices  $\mathcal{V}$  to the LiDAR points  $\mathcal{P}$  via ICP [2], estimating a rigid transform  $\mathbf{T} \in SE(3)$  initialized from  $(\theta_{\text{glob}}, \mathbf{t}_{\text{cam}})$ . Applying  $\mathbf{T}$  yields an updated global pose  $(\theta'_{\text{glob}}, \mathbf{t}'_{\text{cam}})$  that places the mesh correctly in 3D space. However, since ICP is rigid and does not alter internal joint angles, the pose can become inconsistent with the image. Our final optimization step resolves these inconsistencies by adjusting pose and shape using both image and LiDAR cues.

##### 3.2.3. Mesh Optimization

To resolve the remaining pose inconsistencies highlighted in Figure 2 (red ellipse), we refine all SMPL parameters  $(\beta, \theta, \mathbf{t}_{\text{cam}})$  to better fit both the 2D and 3D data. Inspired by SMPLify [3], we formulate this as minimizing an objective function

$$\mathcal{L}_{\text{total}}(\beta, \theta, \mathbf{t}_{\text{cam}}) = \mathcal{L}_J + \lambda_{3D} \mathcal{L}_{3D} + \lambda_{\theta} \mathcal{L}_{\theta} + \lambda_a \mathcal{L}_a + \lambda_{\beta} \mathcal{L}_{\beta} + \lambda_{\text{occ}} \mathcal{L}_{\theta, \text{occ}} \quad (1)$$

that contains terms for image alignment, 3D alignment, and regularization priors with scalar weights  $\lambda$  (details of scalar weights and optimization provided in the supplementary material):

- **2D joint reprojection** ( $\mathcal{L}_J$ ): Penalizes the distance between the projected 3D joints of the mesh and the detected 2D keypoints in the image,

$$\mathcal{L}_J = \sum_i w_i \rho \left( \Pi_{\mathbf{K}}(\mathbf{R}(\theta) \mathbf{J}_i(\beta)) + \mathbf{t}_{\text{cam}} - \mathbf{J}_i^{\text{est}} \right), \quad (2)$$

where  $w_i$  is the keypoint confidence and  $\rho(\cdot)$  is the German–McClure robust penalty [16].

- **3D LiDAR alignment** ( $\mathcal{L}_{3D}$ ): Minimizes the distance between visible mesh vertices  $\mathcal{V}$  and the LiDAR point cloud

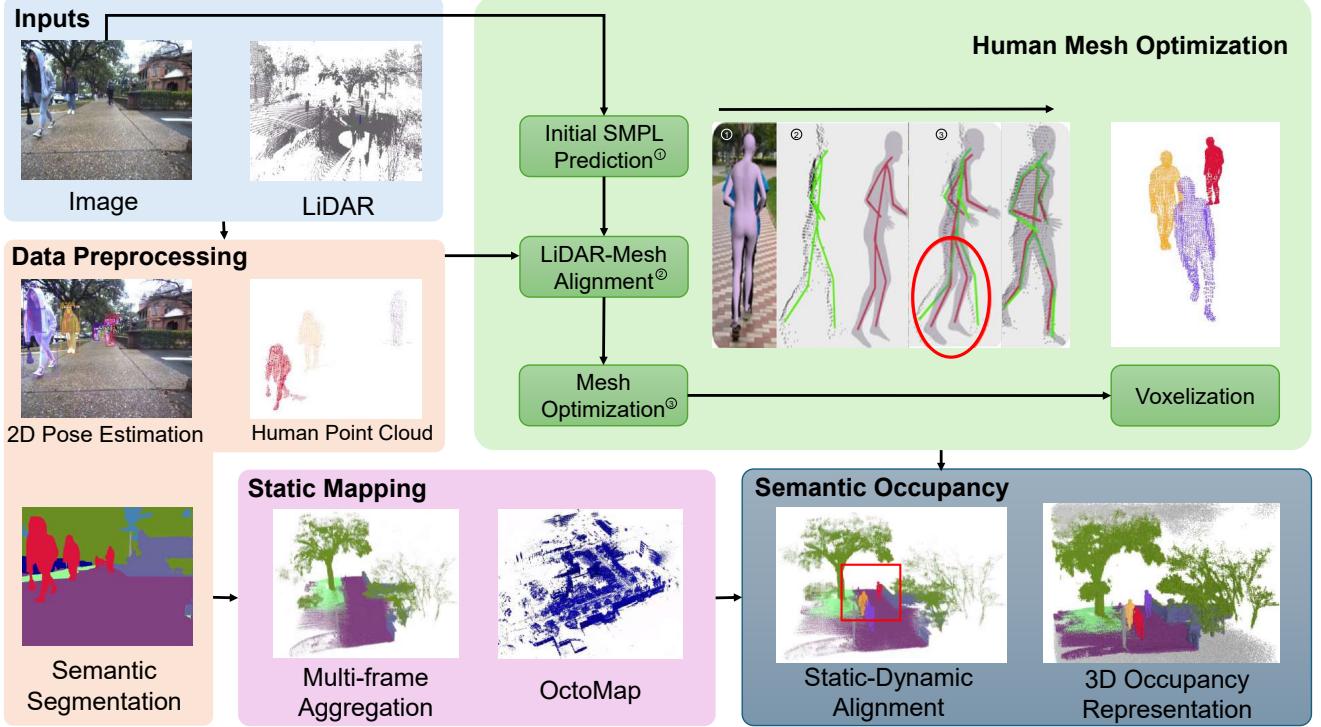


Figure 2. Overview of MobileOcc pipeline. The pipeline consists of: Data preprocessing, Static mapping, Human mesh optimization, and 3D occupancy representation generation. The colors of the voxels follow the labeling scheme used in the Cityscapes dataset [10]. Human instances are assigned colors, and unknown regions are shown in gray. In human mesh optimization, the green and red skeletons represent the ground truth and the predicted 3D pose of the mesh, respectively.

$\mathcal{P}$  using a symmetric, size-normalized Chamfer distance:

$$\mathcal{L}_{3D} = \frac{1}{|\mathcal{V}|} \sum_{\mathbf{v} \in \mathcal{V}} \min_{\mathbf{p} \in \mathcal{P}} \|\mathbf{v} - \mathbf{p}\|_2^2 + \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \min_{\mathbf{v} \in \mathcal{V}} \|\mathbf{p} - \mathbf{v}\|_2^2. \quad (3)$$

Optimizing over  $\theta$  and  $\beta$  lets this term correct local pose/shape errors that pure 2D fitting cannot.

- **Pose prior ( $\mathcal{L}_\theta$ ):** The SMPLify MoG pose prior [3] discourages implausible joint configurations,

$$\mathcal{L}_\theta = -\log \sum_{j=1}^N g_j \mathcal{N}(\theta; \mu_{\theta,j}, \Sigma_{\theta,j}). \quad (4)$$

- **Anti-hyperextension prior ( $\mathcal{L}_a$ ):** Discourages unnatural positive bending at elbows/knees (as in [3]),

$$\mathcal{L}_a = \sum_{k \in \{\text{knees, elbows}\}} \exp(\theta_k). \quad (5)$$

- **Shape prior ( $\mathcal{L}_\beta$ ):** A quadratic prior on shape parameters keeps  $\beta$  within the SMPL space [35],

$$\mathcal{L}_\beta = \beta^\top \Sigma_\beta^{-1} \beta. \quad (6)$$

- **Occlusion-aware pose consistency prior ( $\mathcal{L}_{\theta,\text{occ}}$ ):** For joints flagged invisible by the visibility filter, we regularize deviation from the initialization to prevent unrealistic

drift. Let  $\mathcal{I}$  be the set of occluded joints with initial and current unit quaternions  $\{\mathbf{q}_i^{(0)}, \mathbf{q}_i\}$ ; we use

$$\mathcal{L}_{\theta,\text{occ}} = \sum_{i \in \mathcal{I}} (1 - \langle \mathbf{q}_i^{(0)}, \mathbf{q}_i \rangle)^2. \quad (7)$$

We optimize  $\mathcal{L}_{\text{total}}$  with respect to  $(\beta, \theta, t_{\text{cam}})$  using gradient descent in PyTorch. The result is a refined set  $(\hat{\beta}, \hat{\theta}, t_{\text{cam}}^*)$  that balances fitting the image (2D joints) and fitting the LiDAR (3D points), while respecting human shape and pose priors. In contrast to SMPLify [3], we omit the interpenetration term and instead add  $\mathcal{L}_{3D}$  (LiDAR-mesh Chamfer) and  $\mathcal{L}_{\theta,\text{occ}}$  (occlusion-aware pose consistency), while keeping  $\mathcal{L}_J$ ,  $\mathcal{L}_\theta$ ,  $\mathcal{L}_a$ ,  $\mathcal{L}_\beta$ . We drop interpenetration due to its high cost and limited impact on final accuracy [25]. Our occlusion-aware visibility and pose consistency terms help keep occluded limbs in reasonable positions without explicit interpenetration handling.

### 3.3. Static Map Generation

We build a high-resolution static semantic map by fusing multi-frame LiDAR geometry with camera-derived semantics, while explicitly filtering dynamic pedestrians, including those outside the camera frustum.

### 3.3.1. LiDAR-based Detection

To avoid moving pedestrians being imprinted into the background, we run a LiDAR 3D detector per sweep and use its pedestrian outputs, together with the image-based pedestrian detection result, as a mask during static mapping. Points falling inside these masks are excluded from the static map generation. This stage does not change semantic labels nor train any model, and its sole purpose is to suppress pedestrian evidence before building the static map.

### 3.3.2. Multi-frame Aggregation

We fuse LiDAR geometry and camera semantics over time in a global frame. Each sweep is pose-compensated, and image semantics are lifted to the corresponding 3D points, which are then inserted into a voxel map that maintains compact per-voxel label counts. As frames accumulate, each voxel label is updated through max-voting, and geometric sanity checks (e.g., ground consistency) reject outliers. After processing the sequence, the final static map is obtained by assigning each voxel a semantic label, yielding a human-free background that we later combine with human meshes to obtain human-aware occupancy.

## 3.4. 3D Occupancy Representation

### 3.4.1. Static-dynamic Alignment

We merge the human-free static map (Section 3.3) with pedestrian evidence (Section 3.2) in a shared robot-local frame as shown in Figure 2. For each frame, we transform the current static points and rasterized meshes into the same local coordinates using the synchronized robot pose, then voxelize them. Static voxels inherit Cityscapes [10] semantics from RGB points. Pedestrian voxels carry per-instance identifiers, as we assign each tracked pedestrian a unique ID label.

### 3.4.2. Label Assignment

After alignment, we instantiate voxel grids in the robot frame with a user-specified resolution (up to 0.02 m) and assign labels in a priority order: dynamic humans take precedence over static background, so pedestrians do not get imprinted into the background during fusion. If no pedestrian points are present, we assign static semantics using a majority-label-wins rule to avoid label oscillation when many points land in the same voxel. Remaining cells are completed using an OctoMap query [18], which contributes free/unknown states so that occupied space is supported by explicit geometry. This produces an aligned voxel grid where static semantics and dynamic human occupancy co-exist, and all remaining space is consistently categorized as free or unknown.

We output our final data in the NuScenes [4] dataset format to ensure easier integration and consistency with existing tools.

## 4. Results

In this section, we present a comprehensive evaluation of our approach. First, we assess human mesh optimization using three different datasets. Second, we train representative baselines on MobileOcc and report performance on two tasks: i) occupancy prediction and ii) pedestrian velocity prediction.

### 4.1. Human Mesh Evaluation

We evaluate our human mesh optimization method in 3DPW [48], SLOPER4D [11], and HumanM3 [13] datasets. Four metrics are used in total:

- PVE (Per-Vertex Error, mm): average Euclidean distance between each predicted mesh vertex and its ground-truth counterpart. We evaluate on the canonical SMPL topology (6890 vertices) when vertex supervision is available.
- MPJPE (Mean Per-Joint Position Error, mm): average Euclidean distance over the dataset’s standard kinematic joint set. Joint locations are obtained from the predicted SMPL mesh via the dataset’s regressor, matching the ground-truth protocol.
- PA-MPJPE (Procrustes-Aligned MPJPE, mm): MPJPE after a rigid Procrustes alignment (rotation, uniform scale, translation) between predicted and ground-truth joints, isolating articulated pose quality from global placement.
- MPERE [14] (Mean Per-Edge Relative Error, unitless): average relative L1 error of mesh edge lengths between prediction and ground truth.

#### 4.1.1. Results on the 3DPW Dataset

3DPW [48] dataset is a popular in-the-wild dataset with 3D pose and shape ground truth. Since no LiDAR points are available in it, we add synthetic LiDAR sweeps with different densities in the camera frame. Specifically, we mimic Ouster-style sensors with vertical beams = 32, 64, 128 (higher → denser body coverage) and added Gaussian range perturbations and probabilistic point dropouts following the CARLA LiDAR noise model [12] (details of how the synthetic LiDAR sweeps can be found in the supplementary material). This allows us to i) isolate the effect of adding LiDAR points to image/video HMR; and ii) study how performance scales with sensor density.

Table 1 presents the results of our method compared with other image- and video-based HMR approaches evaluated on the 3DPW dataset. Since image- and video-based methods inherently lack depth, they are evaluated with root alignment [48], which translates the predicted pelvis to the ground truth position. In contrast, our method instead uses simulated LiDAR on the human body to optimize position, pose, and shape directly in the sensor frame.

The results show that our method improves in all metrics without using root alignment, indicating better absolute placement (PVE/MPJPE) and pose accuracy (PA-MPJPE).

Modality	Method	PVE ↓	MPJPE ↓	PA-MPJPE ↓
Video	VIBE [24]	99.1	82.9	51.9
	DynaBOA [17]	82.0	65.5	40.4
	MotionBERT [59]	79.4	68.8	40.6
	WHAM-B [42]	71.0	59.4	37.2
	SMPLify [3]	106.8	—	—
	TRACE [45]	97.3	79.1	37.8
Image	SPIN [25]	96.9	59.2	—
	ROMP [44]	93.4	76.7	47.3
	HybrIK [26]	82.3	71.6	41.8
	CLIFF [30]	81.2	69.0	43.0
	Cha. [6]	76.3	66.0	39.0
	PLIKS [41]	73.3	60.5	38.5
LiDAR	Ours (Ouster-32)	73.2	57.0	47.7
	Ours (Ouster-64)	57.2	43.9	38.5
	<b>Ours (Ouster-128)</b>	<b>50.5</b>	<b>39.1</b>	<b>35.1</b>

Table 1. Simulated LiDAR comparison on 3DPW [48]. Video/Image methods use no depth and are reported with root alignment. Ours fuses simulated LiDAR and operates without root alignment in the sensor frame.

Performance increases with denser LiDAR, which provides more reliable 3D correspondences.

#### 4.1.2. Results on the SLOPER4D and HumanM3 Datasets

We further evaluate our method on SLOPER4D [11] and HumanM3 [13] datasets, which are targeted for LiDAR-based 3D human pose estimation. SLOPER4D [11] has synchronized RGB image and LiDAR points, and ground-truth mesh vertices. An Ouster-OS1-128 LiDAR is used for data collection, and only one person appears in the dataset at a time. HumanM3 [13] has multiple humans at a distance, and a Livox Mid-100 LiDAR is used. It has ground-truth for 3D joints but not for the mesh.

On SLOPER4D, we report PVE, MPJPE, PA-MPJPE, and MPERE. MPERE [14] is computed for any method that outputs a mesh (SMPL-based or model-free that reconstructs surfaces). On HumanM3, vertex ground truth is unavailable, so we report only joint metrics (MPJPE and PA-MPJPE). At test time, we reuse the 3DPW pipeline: visibility filtering, single-sweep rigid pre-alignment, and joint refinement using RGB and LiDAR, all evaluated in the sensor frame.

We compare against different LiDAR methods that have been benchmarked on these two datasets and add additional results with our base image model, CLIFF [30]. As summarized in Table 2, our method achieves strong accuracy without any LiDAR-specific training. We initialize from an image model (CLIFF) and improve substantially through test-time optimization with LiDAR constraints. Table 3 isolates this effect on PA-MPJPE by directly comparing the initialization (CLIFF) to our optimized result (Ours). The substantial reductions on both SLOPER4D and HumanM3

Method	SLOPER4D [11]			HumanM3 [13]
	PVE ↓	MPJPE ↓	MPERE ↓	MPJPE ↓
<i>LiDAR-based</i>				
PRN [14]	—	57.0	—	82.2
V2V-PoseNet [38]	—	<b>50.7</b>	—	83.0
LiDAR-HMR [14]	<b>51.9</b>	51.0	0.094	<b>77.6</b>
LiDARCap [27]	148.1	158.3	<b>0.050</b>	175.8
SAHSR [21]	81.2	72.6	0.085	105.5
VoteHMR [34]	60.9	54.6	0.079	105.8
<i>RGB-based</i>				
CLIFF [30]	93.7	84.7	0.057	106.3
<i>RGB+LiDAR</i>				
<b>Ours</b>	64.4	55.8	0.060	83.9

Table 2. Comparison on real LiDAR: SLOPER4D and HumanM3. Modality denotes the sensor used at inference (RGB = monocular/stereo image streams, LiDAR = point clouds, RGB+LiDAR = fused).

Method	Modality	PA-MPJPE (mm) ↓	
		SLOPER4D	HumanM3
CLIFF [30]	RGB	69.3	66.5
<b>Ours</b>	<b>RGB+LiDAR</b>	<b>43.5</b>	<b>58.1</b>

Table 3. Comparison in PA-MPJPE (mm) between our base image-model CLIFF and Ours after LiDAR optimization.

confirm that most of the gains arise from the LiDAR-guided optimization. Most competing methods are trained on LiDAR (and often dataset-specific) supervision, whereas our improvements arise purely from the optimization process at inference.

## 4.2. Benchmark Results

### 4.2.1. Benchmark Details

We evaluate all methods using a voxel resolution of 0.2 m, with the occupancy grid spanning  $x \in [0.4, 10.0]$  m,  $y \in [-4.8, 4.8]$  m, and  $z \in [-1.0, 3.8]$  m. The raw sequences are downsampled to 5 Hz by taking every second frame. The training split contains 92,303 samples, and the validation split contains 24,208 samples, corresponding to 79.22% and 20.78% of the total data, respectively. Among these, 30,457 training frames and 7,165 validation frames have pedestrian annotations. We use 10 semantic classes, including one free-space class and nine occupied classes. Rare or ambiguous categories are merged: bicycles, motorcycles, and scooters are combined into a single two-wheeler class, and several static structural categories are grouped into an other-structure class.

### 4.2.2. Baseline Implementations

**Monocular BEV-based baselines.** Our monocular baselines include BEVDet4D [19] as a detection and velocity baseline, and FlashOcc [54] and Panoptic-FlashOcc [55] as

■ pedestrian ■ car ■ other struct. ■ pole ■ road ■ terrain ■ truck ■ two-wheeler ■ vegetation

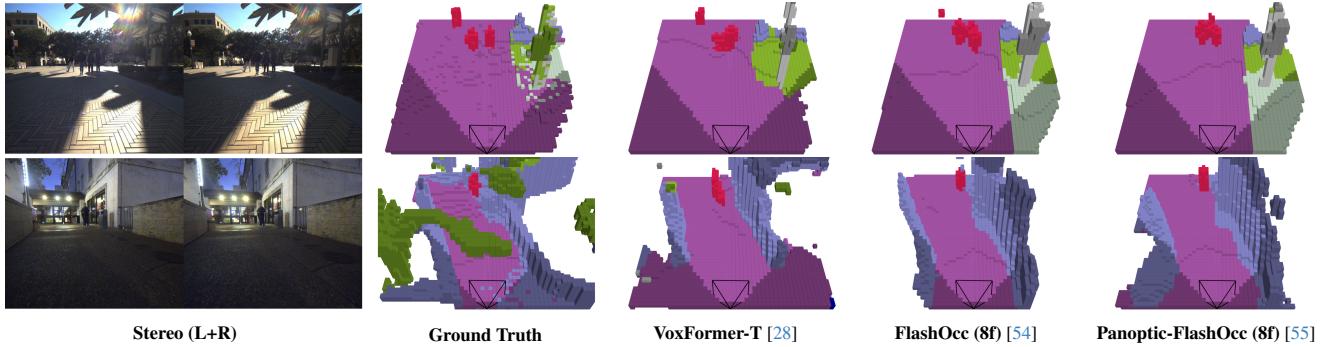


Figure 3. Qualitative comparison of different baselines under various lighting conditions, including sunny, night, and cloudy scenes.

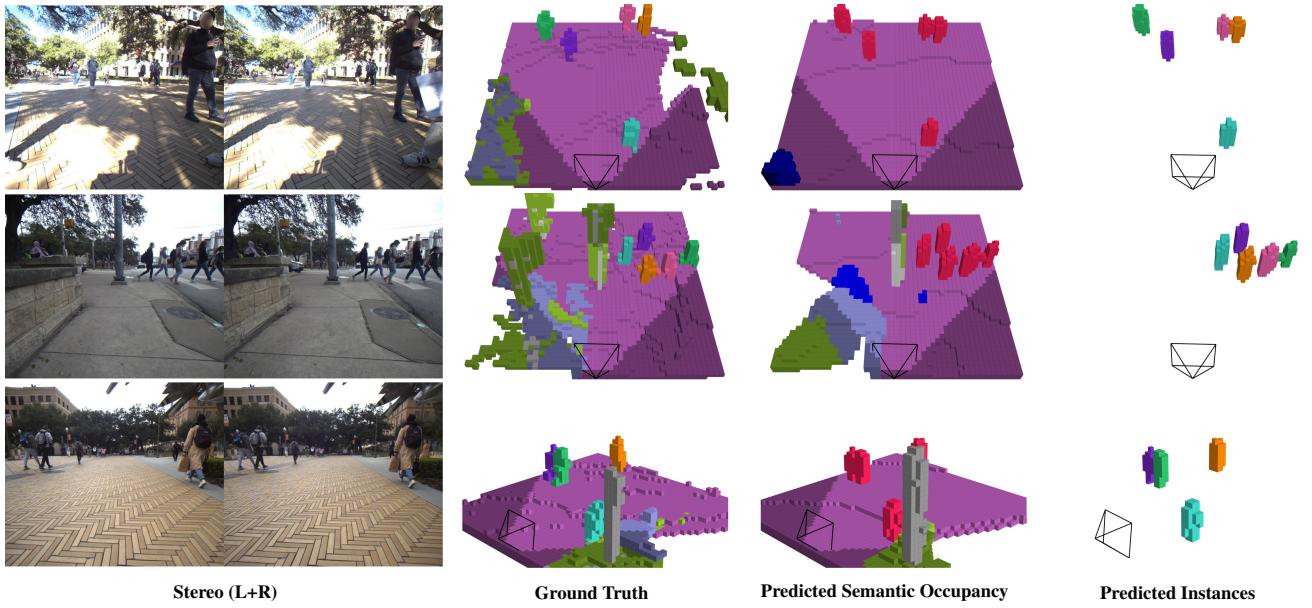


Figure 4. Panoptic occupancy prediction performance using Panoptic-FlashOcc (8f) [55].

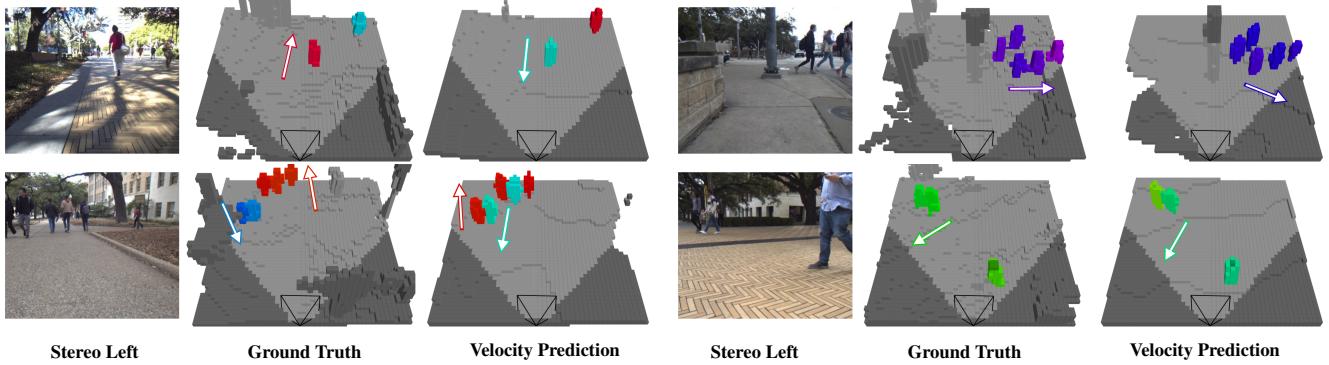


Figure 5. Pedestrian velocity prediction using Panoptic-FlashOcc-vel (8f). The directions are shown in colors and arrows.

semantic and panoptic 3D occupancy baselines. We use 8 historical frames based on their public settings. The original BEVDet4D, FlashOcc, and Panoptic-FlashOcc models assume multi-view surround-camera inputs, whereas our dataset contains only front-view images. Therefore, we adapt them to a monocular setting by using only the left stereo camera for depth estimation. We also introduce Panoptic-FlashOcc-vel, which extends the detection head of Panoptic-FlashOcc to predict velocities ( $v_x, v_y$ ) with an L1 loss for velocity supervision.

**Stereo baseline:** **VoxFormer** [28]. We adapt VoxFormer-T with 4 historical frames to our stereo setup, sampling frames alternately in time to match the temporal coverage of the 8-frame monocular baselines. Stage-1 takes voxelized stereo-matching depth as input and predicts a 3D occupancy volume at 0.4m resolution, which is then upsampled to our 0.2m target grid. We follow the public training settings of each baseline. The detailed hyperparameters are shown in the supplementary material.

#### 4.2.3. Main Results

**3D occupancy prediction.** Table 4 and Figure 3 compare the baselines on the semantic occupancy prediction task. FlashOcc (FO) achieves the highest mean IoU (mIoU) and outperforms the other methods on most classes. Panoptic-FlashOcc (PF) yields a minor mIoU drop relative to FO due to the multi-task trade-off. VoxFormer (VF) remains competitive on pedestrians, possibly because it benefits from stereo depth, whereas FlashOcc may inherit monocular pedestrian detection errors from BEVDet4D pretraining. Cars and trucks show lower IoU than in self-driving benchmarks, primarily because they are less common in our dataset.

**Panoptic occupancy.** We further evaluate the pedestrian detection and panoptic occupancy quality using BEVDet4D and Panoptic-FlashOcc. As shown in Table 5 and Figure 4, BEVDet4D (BD) provides only pedestrian detections, while Panoptic-FlashOcc (PF) predicts voxel-level panoptic labels. Pedestrian Average Precision ( $AP^{Ped}$ ) is evaluated across distance thresholds 0.1 m, 0.2 m, 0.5 m and 1 m on detected centers. Besides standard panoptic metrics, we also report  $PQ^\dagger$ , a relaxed panoptic quality [39] that relaxes the strict IoU requirement for stuff classes and is better suited for vision-only occupancy prediction.

**Pedestrian velocity prediction.** Finally, we compare BEVDet4D and Panoptic-FlashOcc-vel (PF-vel) on the pedestrian velocity prediction task. Table 6 summarizes the absolute velocity error (AVE; m/s) over all ground truth pedestrian voxels (AVE-T), true-positive detections within 1 m (AVE-D), and correctly classified voxels occupied by pedestrians (AVE-O), together with mIoU. PF-vel achieves comparable velocity accuracy while maintaining voxel-level semantic occupancy. Qualitative examples in Figure 5 show that PF-vel produces plausible velocities for

most pedestrians but still exhibits typical failure cases such as confusion between forward and backward walking directions, indicating remaining headroom for improving pedestrian velocity prediction together with panoptic occupancy.

Method	IoU	mIoU									
			pedestrian	car	other struct.	pole	road	terrain	truck	two-wheeler	vegetation
VF-T	57.81	24.89	<b>32.79</b>	1.39	28.47	7.08	<b>70.90</b>	23.57	5.23	28.04	26.52
FO (8f)	57.71	<b>27.18</b>	31.91	<b>5.36</b>	31.00	<b>10.16</b>	70.82	<b>27.29</b>	<b>7.35</b>	30.85	<b>29.88</b>
PF (8f)	<b>57.83</b>	26.64	32.45	3.66	<b>31.79</b>	9.96	70.35	25.90	5.87	<b>30.99</b>	28.83

Table 4. 3D occupancy prediction performance. VF-T denotes VoxFormer-T, and FO and PF denote FlashOcc and Panoptic-FlashOcc. The best results are shown in **bold**. IoU denotes the geometric IoU.

Method	PQ	$PQ^\dagger$	RQ	SQ	$PQ^{Ped}$	$RQ^{Ped}$	$SQ^{Ped}$	$AP^{Ped}$
BD (8f)	—	—	—	—	—	—	—	41.7
PF (8f)	19.9	28.1	65.8	32.6	42.5	60.2	70.7	45.5

Table 5. Baseline performance on 3D panoptic occupancy. BD denotes BEVDet4D and PF denotes Panoptic-FlashOcc.

Method	AVE-T $\downarrow$	AVE-D $\downarrow$	AVE-O $\downarrow$	mIoU $\uparrow$
BEVDet4D (8f)	1.00	0.36	—	—
PF-vel (8f)	0.97	0.39	0.67	26.00

Table 6. Comparison of pedestrian absolute velocity error (AVE).

## 5. Conclusion

This paper presents MobileOcc, a human-aware semantic occupancy dataset for mobile robots operating in pedestrian-dense, near-field environments. We provide baselines for occupancy prediction and pedestrian-velocity estimation across monocular, stereo, and panoptic settings. We further validate the human-mesh optimization component on established datasets, where the addition of LiDAR points for optimization substantially improves image-based methods and remains robust across datasets. Together, the dataset, pipeline, and benchmarks form a practical testbed for perception stacks that must jointly reason about humans, static objects, and free space to enable safe, precise navigation in crowds.

**Limitations and future work.** Currently, MobileOcc focuses on outdoor scenes, which may limit generalization across domains and conditions. Future extensions include broadening geographic and environment diversity (e.g., indoor spaces and adverse weather).

## References

- [1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019. [2](#)
- [2] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, pages 586–606. Spie, 1992. [3](#)
- [3] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European conference on computer vision*, pages 561–578. Springer, 2016. [3, 4, 6](#)
- [4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liang, Qiang Xu, Anush Krishnan, Yu Pan, Giacomo Baldan, and Oscar Beijbom. nuScenes: A multi-modal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. [2, 5](#)
- [5] Jinkun Cao, Jiangmiao Pang, Xinshuo Weng, Rawal Khirodkar, and Kris Kitani. Observation-centric sort: Rethinking sort for robust multi-object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9686–9696, 2023. [3](#)
- [6] Junuk Cha, Muhammad Saqlain, GeonU Kim, Mingyu Shin, and Seungryul Baek. Multi-person 3d pose and shape estimation via inverse kinematics and refinement. In *European conference on computer vision*, pages 660–677. Springer, 2022. [6](#)
- [7] Gang Chen, Zhaoying Wang, Wei Dong, and Javier Alonso-Mora. Particle-based instance-aware semantic occupancy mapping in dynamic environments. *Trans. Rob.*, 41:1155–1171, 2025. [2](#)
- [8] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. [3](#)
- [9] Hongsuk Choi, Gyeongsik Moon, and Kyoung Mu Lee. Pose2mesh: Graph convolutional network for 3d human pose and mesh recovery from a 2d human pose. In *European Conference on Computer Vision*, pages 769–787. Springer, 2020. [2](#)
- [10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. [3, 4, 5](#)
- [11] 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*, pages 682–692, 2023. [5, 6](#)
- [12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. [5](#)
- [13] Bohao Fan, Siqi Wang, Wenzuan Guo, Wenzhao Zheng, Jianjiang Feng, and Jie Zhou. Human-m3: A multi-view multi-modal dataset for 3d human pose estimation in outdoor scenes. *arXiv preprint arXiv:2308.00628*, 2023. [5, 6](#)
- [14] Bohao Fan, Wenzhao Zheng, Jianjiang Feng, and Jie Zhou. Lidar-hmr: 3d human mesh recovery from lidar. *IEEE Transactions on Multimedia*, 2025. [2, 5, 6](#)
- [15] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. [3](#)
- [16] Donald Geman and Stuart Geman. Bayesian image analysis. In *Disordered systems and biological organization*, pages 301–319. Springer, 1986. [3](#)
- [17] Shanyan Guan, Jingwei Xu, Michelle Zhang He, Yunbo Wang, Bingbing Ni, and Xiaokang Yang. Out-of-domain human mesh reconstruction via dynamic bilevel online adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):5070–5086, 2022. [6](#)
- [18] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34(3):189–206, 2013. [2, 5](#)
- [19] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint arXiv:2203.17054*, 2022. [2, 6](#)
- [20] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9223–9232, 2023. [1, 2](#)
- [21] Haiyong Jiang, Jianfei Cai, and Jianmin Zheng. Skeleton-aware 3d human shape reconstruction from point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5431–5441, 2019. [6](#)
- [22] Peng Jiang, Philip Osteen, Maggie Wigness, and Srikanth Saripalli. Rellis-3d dataset: Data, benchmarks and analysis. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 1110–1116. IEEE, 2021. [2](#)
- [23] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7122–7131, 2018. [2, 3](#)
- [24] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. Vibe: Video inference for human body pose and shape estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5253–5263, 2020. [6](#)
- [25] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2252–2261, 2019. [2, 4, 6](#)
- [26] Jiefeng Li, Chao Xu, Zhicun Chen, Siyuan Bian, Lixin Yang, and Cewu Lu. Hybrik: A hybrid analytical-neural inverse

- kinematics solution for 3d human pose and shape estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3383–3393, 2021. 6
- [27] Jialian Li, Jingyi Zhang, Zhiyong Wang, Siqi Shen, Chenglu Wen, Yuexin Ma, Lan Xu, Jingyi Yu, and Cheng Wang. Ladarcap: Long-range marker-less 3d human motion capture with lidar point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20502–20512, 2022. 2, 6
- [28] Yiming Li, Zhiding Yu, Christopher Choy, Chaowei Xiao, Jose M Alvarez, Sanja Fidler, Chen Feng, and Anima Anandkumar. Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9087–9098, 2023. 1, 2, 7, 8
- [29] Yiming Li, Sihang Li, Xinhao Liu, Moonjun Gong, Kenan Li, Nuo Chen, Zijun Wang, Zhiheng Li, Tao Jiang, Fisher Yu, et al. Sscbench: A large-scale 3d semantic scene completion benchmark for autonomous driving. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13333–13340. IEEE, 2024. 1, 2
- [30] 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 *European Conference on Computer Vision*, pages 590–606. Springer, 2022. 2, 3, 6
- [31] Zhiqi Li, Zhiding Yu, David Austin, Mingsheng Fang, Shiyi Lan, Jan Kautz, and Jose M Alvarez. Fb-occ: 3d occupancy prediction based on forward-backward view transformation. *arXiv preprint arXiv:2307.01492*, 2023. 2
- [32] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1954–1963, 2021. 2
- [33] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphomer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12939–12948, 2021. 2
- [34] Guanze Liu, Yu Rong, and Lu Sheng. Votehmr: Occlusion-aware voting network for robust 3d human mesh recovery from partial point clouds. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 955–964, 2021. 6
- [35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023. 2, 3, 4
- [36] Ruihang Miao, Weizhou Liu, Mingrui Chen, Zheng Gong, Weixin Xu, Chen Hu, and Shuchang Zhou. Occdepth: A depth-aware method for 3d semantic scene completion. *arXiv preprint arXiv:2302.13540*, 2023. 1, 2
- [37] Gyeongsik Moon and Kyoung Mu Lee. I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single rgb image. In *European Conference on Computer Vision*, pages 752–768. Springer, 2020. 2
- [38] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In *Proceedings of the IEEE conference on computer vision and pattern Recognition*, pages 5079–5088, 2018. 6
- [39] Lorenzo Porzi, Samuel Rota Bulo, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8277–8286, 2019. 8
- [40] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2304–2314, 2019. 2
- [41] Karthik Shetty, Annette Birkhold, Srikrishna Jaganathan, Norbert Strobel, Markus Kowarschik, Andreas Maier, and Bernhard Egger. Pliks: A pseudo-linear inverse kinematic solver for 3d human body estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 574–584, 2023. 6
- [42] Soyong Shin, Juyong Kim, Eni Halilaj, and Michael J Black. Wham: Reconstructing world-grounded humans with accurate 3d motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2070–2080, 2024. 6
- [43] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 2
- [44] Yu Sun, Qian Bao, Wu Liu, Yili Fu, Michael J Black, and Tao Mei. Monocular, one-stage, regression of multiple 3d people. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11179–11188, 2021. 6
- [45] Yu Sun, Qian Bao, Wu Liu, Tao Mei, and Michael J Black. Trace: 5d temporal regression of avatars with dynamic cameras in 3d environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8856–8866, 2023. 6
- [46] Xiaoyu Tian, Tao Jiang, Longfei Yun, Yucheng Mao, Huitong Yang, Yue Wang, Yilun Wang, and Hang Zhao. Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. *Advances in Neural Information Processing Systems*, 36:64318–64330, 2023. 1, 2
- [47] Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, et al. Scene as occupancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8406–8415, 2023. 1, 2
- [48] Timo Von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European conference on computer vision (ECCV)*, pages 601–617, 2018. 5, 6, 2, 3
- [49] Xiaofeng Wang, Zheng Zhu, Wenbo Xu, Yunpeng Zhang, Yi Wei, Xu Chi, Yun Ye, Dalong Du, Jiwen Lu, and Xinggang Wang. Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17850–17859, 2023. 1, 2

- [50] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21729–21740, 2023. [1](#), [2](#)
- [51] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J Black. Icon: Implicit clothed humans obtained from normals. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13286–13296. IEEE, 2022. [2](#)
- [52] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *Advances in neural information processing systems*, 35:38571–38584, 2022. [3](#)
- [53] Ze Yang, Shenlong Wang, Sivabalan Manivasagam, Zeng Huang, Wei-Chiu Ma, Xinchen Yan, Ersin Yumer, and Raquel Urtasun. S3: Neural shape, skeleton, and skinning fields for 3d human modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13284–13293, 2021. [2](#)
- [54] Zichen Yu, Changyong Shu, Jiajun Deng, Kangjie Lu, Zongdai Liu, Jiangyong Yu, Dawei Yang, Hui Li, and Yan Chen. Flashocc: Fast and memory-efficient occupancy prediction via channel-to-height plugin. *arXiv preprint arXiv:2311.12058*, 2023. [2](#), [6](#), [7](#)
- [55] Zichen Yu, Changyong Shu, Qianpu Sun, Yifan Bian, Xiaobao Wei, Jiangyong Yu, Zongdai Liu, Dawei Yang, Hui Li, and Yan Chen. Panoptic-flashocc: An efficient baseline to marry semantic occupancy with panoptic via instance center. *arXiv preprint arXiv:2406.10527*, 2024. [2](#), [6](#), [7](#)
- [56] Heng Zhai, Jilin Mei, Chen Min, Liang Chen, Fangzhou Zhao, and Yu Hu. Wildocc: A benchmark for off-road 3d semantic occupancy prediction. *arXiv preprint arXiv:2410.15792*, 2024. [2](#)
- [57] Arthur Zhang, Chaitanya Eranki, Christina Zhang, Ji-Hwan Park, Raymond Hong, Pranav Kalyani, Lochana Kalyanaraman, Arsh Gamare, Arnav Bagad, Maria Esteva, et al. Toward robust robot 3-d perception in urban environments: The ut campus object dataset. *IEEE Transactions on Robotics*, 40:3322–3340, 2024. [1](#), [2](#), [3](#)
- [58] Jingyi Zhang, Qihong Mao, Siqi Shen, Chenglu Wen, Lan Xu, and Cheng Wang. Lidarcapv2: 3d human pose estimation with human–object interaction from lidar point clouds. *Pattern Recognition*, 156:110848, 2024. [2](#)
- [59] Wentao Zhu, Xiaoxuan Ma, Zhaoyang Liu, Libin Liu, Wayne Wu, and Yizhou Wang. Motionbert: A unified perspective on learning human motion representations. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15085–15099, 2023. [6](#)

# MobileOcc: A Human-Aware Semantic Occupancy Dataset for Mobile Robots

## Supplementary Material

### A. Visibility filtering

We follow the notation introduced in Sec. 3.2 of the main paper: SMPL parameters  $(\beta, \theta, t_{\text{cam}})$ , mesh  $\mathcal{M}$  with visible subset  $\mathcal{V} \subset \mathcal{M}$ , LiDAR point cloud  $\mathcal{P}$ , and camera intrinsics  $\mathbf{K}$ . Our visibility filtering selects a subset of mesh vertices that i) are geometrically visible from the camera and ii) belong to body parts that are likely unoccluded in the image. The resulting visible subset  $\mathcal{V}$  is then used for ICP registration and for the 3D LiDAR alignment loss  $\mathcal{L}_{3D}$ .

#### A.1. Backface culling

The SMPL mesh  $\mathcal{M}$  consists of triangular faces, each with three vertices in 3D. For each face  $f$ , we compute i) a unit face normal  $\mathbf{n}_f$  that indicates which direction the face is pointing, and ii) a representative point  $\mathbf{c}_f$  at the center (centroid) of the triangle. Let  $\mathbf{o}$  denote the 3D position of the camera. The viewing direction from the camera to face  $f$  is then

$$\mathbf{p}_f = \frac{\mathbf{c}_f - \mathbf{o}}{\|\mathbf{c}_f - \mathbf{o}\|_2}. \quad (8)$$

We classify a face as front-facing if the angle between its normal and the viewing direction is sufficiently small. We implement this using the inner product between the normal and the viewing direction:

$$\mathbf{n}_f^\top \mathbf{p}_f < w, \quad (9)$$

where  $w$  is a backface-culling threshold (tuned per dataset, see Table 7). All faces that satisfy this condition are considered front-facing, and all vertices that belong to at least one such face form a candidate visible set  $\mathcal{V}_{\text{front}} \subset \mathcal{M}$ .

This step removes mesh regions oriented away from the camera. It therefore should not be matched to LiDAR points, thereby reducing the influence of faces that are clearly not visible from the current viewpoint.

#### A.2. Body-part filtering from 2D keypoints

Backface culling alone does not handle self-occlusion (e.g., the arm on the far side of the body) or occlusion by other objects. To further restrict the mesh to regions observed, we use the same 2D keypoints and confidence scores as in the 2D joint reprojection loss,  $\mathcal{L}_J$ .

Let  $\mathbf{J}_i^{\text{est}}$  denote the detected 2D joints and  $w_i$  their confidence scores. For each SMPL body part  $b$  (e.g., torso, left upper arm, right lower leg), we define a small set of 2D joints  $\mathcal{J}_b$  that are informative for that part (for example, shoulder and elbow for an upper arm). A body part is marked as occluded if all of its associated joints have low

confidence:

$$\max_{i \in \mathcal{J}_b} w_i < J_{\text{conf}}, \quad (10)$$

where  $J_{\text{conf}}$  is a dataset-specific threshold, listed in Table 7.

For each occluded body part, we remove all mesh faces whose vertices belong to that part from the candidate visible set  $\mathcal{V}_{\text{front}}$  obtained by backface culling. Since the SMPL template provides a fixed correspondence between mesh faces and semantic body parts, this mapping is known in advance and does not depend on pose or shape. The remaining faces define a refined visible mesh that is consistent with both the camera viewpoint and the 2D keypoint evidence. This body-part filtering improves robustness in cases with occluded limbs by preventing LiDAR points from being matched to mesh regions that are not supported by the image.

#### A.3. Final visible vertex set

Combining backface culling and body-part filtering yields our final set of visible mesh vertices  $\mathcal{V} \subset \mathcal{M}$ . By construction,

- $\mathcal{V}$  contains only vertices that belong to front-facing faces, and
- vertices assigned to body parts marked as occluded by the 2D keypoints are removed.

We treat  $\mathcal{V}$  as the subset of the mesh that is both visible in the image and observable by the LiDAR sensor. All subsequent 3D processing is restricted to  $\mathcal{V}$ . Focusing on  $\mathcal{V}$  instead of the full mesh reduces the influence of self-occluded or truncated limbs and leads to more stable LiDAR-mesh alignment in scenes with occlusions.

### B. Scalar Weights and Optimization

The mesh refinement in Section 3.2 of the main paper minimizes the multi-term objective in Eq. 1, combining 2D joint reprojection  $\mathcal{L}_J$ , 3D LiDAR alignment  $\mathcal{L}_{3D}$ , pose and shape priors  $\mathcal{L}_\theta$  and  $\mathcal{L}_\beta$ , an anti-hyperextension prior  $\mathcal{L}_a$ , and an occlusion-aware pose consistency term  $\mathcal{L}_{\theta,\text{occ}}$ . All terms are fully differentiable with respect to  $(\beta, \theta, t_{\text{cam}})$ .

Our optimization scheme contains several scalar hyperparameters:

- Loss weights  $\lambda_\theta, \lambda_a, \lambda_\beta, \lambda_{3D}$ , and  $\lambda_{\text{occ}}$ .
- The Geman–McClure scale  $\rho$  used in  $\mathcal{L}_J$ .
- The backface culling threshold  $w$  used in the visibility filter.
- The joint confidence threshold  $J_{\text{conf}}$  used for part-level occlusion reasoning.

These hyperparameters influence the trade-off between image alignment, LiDAR alignment, and prior regularization.

To obtain the best configurations, we perform Bayesian Optimization on a subset that is held out from the final evaluation for each dataset. For each trial, we run the full optimization pipeline with a candidate parameter configuration and evaluate the resulting meshes using the relevant 3D metrics (e.g., PVE, MPJPE, PA-MPJPE). Bayesian Optimization maintains a surrogate model of the objective over the hyperparameter space and iteratively proposes new configurations that balance exploration and exploitation. In practice, we run 100 iterations per dataset and select the best-performing configuration on the subset.

Table 7 lists the resulting values for all datasets used in our experiments. Note that the UT Campus dataset does not provide ground-truth SMPL annotations; for this dataset, we select hyperparameters based on qualitative inspection of the resulting meshes.

Dataset	$\rho$	$\lambda_\theta$	$\lambda_a$	$\lambda_\beta$	$\lambda_{3D}$	$\lambda_{occ}$	$w$	$J_{conf}$
3DPW	100	2.2	11.0	5.0	800	35	0.2	0.6
SLOPER4D	100	0.75	8.5	1.0	500	55	0.2	0.7
HumanM3	100	1.0	3.0	17.5	600	135	-1.0	0.6
UT Campus	100	1.4	10.0	10.0	300	50	0.2	0.7

Table 7. Sub-optimal scalar weights and thresholds per dataset.

In all experiments, we start from the initial mesh prediction provided by the base HMR model and run gradient descent on  $(\beta, \theta, t_{cam})$  until convergence or a fixed iteration budget is reached. The chosen hyperparameters ensure that the optimization improves both image and LiDAR alignment while keeping poses and shapes within a plausible region of the SMPL space and preventing excessive drift of occluded limbs.

## C. Synthetic LiDAR sweeps

For datasets without real LiDAR (3DPW [48]), we simulate LiDAR sweeps directly on the ground-truth SMPL meshes to study how LiDAR characteristics affect our method. For each camera frame, we place a virtual LiDAR at the camera center and emit rays based on the vertical and horizontal angular resolutions of an Ouster-style spinning sensor. Each ray intersects the SMPL mesh to obtain a 3D return, analogous to a time-of-flight measurement. If a ray does not hit the mesh within the valid range, it produces no return.

We model three virtual sensors with increasing angular resolutions: “Ouster-32”, “Ouster-64”, and “Ouster-128”. For each simulated return, we add range and angular perturbations and randomly drop individual points to mimic missed detections. All rays are restricted to the camera frustum, so the simulated LiDAR only observes the same region as the RGB camera. Since 3DPW does not contain full 3D scene geometry, occlusions from other objects are not simulated. Instead, our visibility filter (Section A) discards mesh

regions that are likely occluded in the image.

The sensor configurations are summarized in Table 8. All three sensors share the same noise characteristics and valid distance range [0.5, 90] m, but differ in vertical and horizontal resolution. We use these simulated sweeps as input to our pipeline, keeping all other components identical to the real-LiDAR setting. This design allows us to isolate the impact of LiDAR density and noise on performance.

Nr.	Type	Resolution		Range noise		Angular noise		Range	
		Vert.	Hor.	mean	std	mean	std	min	max
1.	Ouster-32	32	512	$\pm 25.0$	10.0	0.0	0.01	0.5	90.0
2.	Ouster-64	64	1024	$\pm 25.0$	10.0	0.0	0.01	0.5	90.0
3.	Ouster-128	128	2048	$\pm 25.0$	10.0	0.0	0.01	0.5	90.0

Table 8. Specifications of the simulated LiDAR sensors used on 3DPW [48]. Resolution is given in vertical and horizontal channels, range noise mean/std are in millimeters, angular noise mean/std are in degrees, and range min/max are in meters. All sensors use the same noise levels and a fixed dropout probability of 10%.

The simulated LiDAR sweeps approximate realistic Ouster sensors in terms of angular resolution, range noise, and dropout behavior, while being perfectly time-synchronized with the RGB images and SMPL ground truth. This provides a controlled setting to quantify how much additional 3D information is needed for our optimization to improve over purely image-based HMR.

Sensor	Metric	Ours		No visibility filtering			
		mean	std	mean	std	t-value	p-value
Ouster-32	PVE $\downarrow$	82.0	50.9	84.3	60.2	9.512	0.3416
	MPJPE $\downarrow$	62.9	42.6	64.9	52.7	9.623	0.3360
	PA-MPJPE $\downarrow$	50.2	35.0	51.5	42.3	7.720	0.4402
Ouster-64	PVE $\downarrow$	66.9	40.0	69.3	47.6	12.585	0.2082
	MPJPE $\downarrow$	48.2	31.0	50.4	36.4	15.002	0.1337
	PA-MPJPE $\downarrow$	41.1	29.4	42.3	34.8	8.588	0.3905
Ouster-128	PVE $\downarrow$	58.7	35.4	60.2	42.2	8.879	0.3747
	MPJPE $\downarrow$	43.6	26.4	45.3	35.6	12.506	0.2112
	PA-MPJPE $\downarrow$	37.2	27.1	38.7	33.5	11.350	0.2565

Table 9. Ablation study on the visibility filter for partially occluded body parts. “Ours” denotes the full model with the visibility filter, while “No visibility filtering” disables this component.

## C.1. Ablation study

We perform an ablation study on 3DPW [48] to quantify the contribution of each newly introduced component: i) the visibility filter, ii) the 3D LiDAR alignment term  $\mathcal{L}_{3D}$ , and iii) the occlusion-aware pose prior  $\mathcal{L}_{\theta,occ}$ . For each of the three simulated sensors, we run our method with one component removed at a time and compare PVE, MPJPE, and PA-MPJPE to the full model.

Sensor	Metric	Ours				No $\mathcal{L}_{\theta,\text{occ}}$				No $\mathcal{L}_{3D}$			
		mean	std	mean	std	t-value	p-value	mean	std	t-value	p-value		
Ouster-32	PVE $\downarrow$	73.2	50.9	75.9	75.9	-68.66	0.00	171.2	113.1	-191.95	0.00		
	MPJPE $\downarrow$	57.0	42.6	58.2	43.2	-66.92	0.00	159.3	113.2	-194.86	0.00		
	PA-MPJPE $\downarrow$	47.6	35.0	48.9	35.5	-53.33	0.00	49.6	22.1	-78.44	0.00		
Ouster-64	PVE $\downarrow$	57.2	40.0	60.1	41.3	-33.20	$1.09 \times 10^{-23}$	173.2	117.4	-188.63	0.00		
	MPJPE $\downarrow$	43.9	31.0	45.2	31.8	-27.73	0.00	161.7	117.8	-191.47	0.00		
	PA-MPJPE $\downarrow$	38.5	29.4	40.4	31.0	-24.21	0.00	49.2	21.6	-77.00	0.00		
Ouster-128	PVE $\downarrow$	50.5	35.4	52.4	36.8	-7.18	$1.43 \times 10^{-12}$	174.9	121.4	-185.41	0.00		
	MPJPE $\downarrow$	39.1	26.4	40.3	27.6	-5.74	$9.71 \times 10^{-9}$	163.4	121.8	-188.04	0.00		
	PA-MPJPE $\downarrow$	35.1	27.1	36.6	28.9	-7.27	$3.62 \times 10^{-13}$	49.4	23.0	-76.71	0.00		

Table 10. Ablation study on the proposed loss terms in the objective function (Eq. 1), evaluated on the full 3DPW [48] test set (35,515 frames). Removing the occlusion-aware pose prior  $\mathcal{L}_{\theta,\text{occ}}$  slightly degrades performance, while removing the 3D LiDAR alignment term  $\mathcal{L}_{3D}$  leads to a drastic increase in error across all sensors.

Table 9 focuses on the visibility filter. Since this component only affects frames where at least one body part is (partially) occluded, we construct a subset of 1,063 frames from 3DPW [48] that each contain at least one occluded limb (about 2.9% of the dataset). On this subset, we compare our full model with visibility filtering to a variant where the filter is disabled. Across all three simulated sensors, removing the visibility filter consistently increases PVE, MPJPE, and PA-MPJPE, although the numerical differences are modest. This confirms that masking out occluded regions helps in precisely those challenging cases, while having little effect on fully visible poses.

Table 10 investigates the two new loss terms from Section 3.2: the occlusion-aware pose prior  $\mathcal{L}_{\theta,\text{occ}}$  and the 3D LiDAR alignment term  $\mathcal{L}_{3D}$ . Here, we evaluate on the full 3DPW test set (35,515 frames). For each sensor, we report the performance of the full model (“Ours”) and compare it to i) a variant without  $\mathcal{L}_{\theta,\text{occ}}$  and ii) a variant without  $\mathcal{L}_{3D}$ .

The 3D term  $\mathcal{L}_{3D}$  has by far the most significant impact. Removing it roughly doubles the PVE and MPJPE across all three sensors, bringing performance close to that of using only ICP alignment without any LiDAR-aware refinement. This confirms that the LiDAR–mesh Chamfer term is crucial for correcting local pose and shape errors beyond what ICP alone can provide.

The occlusion-aware pose prior  $\mathcal{L}_{\theta,\text{occ}}$  yields a more minor but consistent gain: disabling this term slightly worsens all metrics, indicating that softly constraining occluded joints toward their initial configuration helps prevent unrealistic drift while still allowing visible joints to move freely.

## D. Baseline Details

**BEVDet4D.** The public configuration uses an effective batch size of 64 (8×8), a learning rate of  $2 \times 10^{-4}$ , and 20 epochs. We retain these settings but train on a single A40 using batch 8 with 8-step gradient accumulation, and shorten training to 15 epochs. The BEV and depth ranges

are  $x \in [0, 12.8]$  m,  $y \in [-6.4, 6.4]$  m, and depth [1, 15] m with 0.5 m bins.

**FlashOcc and its variants.** All variants are initialized from our BEVDet4D (1f) checkpoint (12 epochs). The original setup uses LR  $1 \times 10^{-4}$ , effective batch 16 (4×4), and 24 epochs; we match the batch size on a single A40 via batch 4 with 4-step accumulation and train for 15 epochs. The occupancy volume follows our final evaluation grid:  $x \in [0.4, 10.0]$  m,  $y \in [-4.8, 4.8]$  m,  $z \in [-1.0, 3.8]$  m at 0.2 m resolution, with depth bins [1, 12] m at 0.5 m.

**VoxFormer.** For VoxFormer-T we keep the public training settings (20 epochs and learning rate  $2 \times 10^{-4}$ ). The original configuration uses batch size 1 on 8 GPUs (effective batch 8); on our single A40 we use batch size 1 with 8-step gradient accumulation to match this.

**Input processing and augmentation.** All methods use the same image pipeline: input images of  $1024 \times 1224$  (H×W) are vertically cropped from rows 144 to 1008 (yielding  $864 \times 1224$ ) and then resized to  $384 \times 544$ . BEVDet4D and the FlashOcc family use the default image augmentations from their public code (no BEV-space augmentation). For fairness, we also implement a random horizontal flip for VoxFormer in the image space.

**EMA and checkpoints.** The public VoxFormer code does not maintain an exponential moving average (EMA) model, so we report results from the final checkpoint. BEVDet4D and the FlashOcc family use EMA in their official implementations, and we therefore evaluate these baselines using their EMA weights.