# Occupancy Planes for Single-view RGB-D Human Reconstruction

**Xiaoming Zhao, Yuan-Ting Hu, Zhongzheng Ren, Alexander G. Schwing**

University of Illinois Urbana-Champaign
https://github.com/Xiaoming-Zhao/oplanes

## Abstract

Single-view RGB-D human reconstruction with implicit functions is often formulated as per-point classification. Specifically, a set of 3D locations within the view-frustum of the camera are first projected independently onto the image and a corresponding feature is subsequently extracted for each 3D location. The feature of each 3D location is then used to classify independently whether the corresponding 3D point is inside or outside the observed object. This procedure leads to sub-optimal results because correlations between predictions for neighboring locations are only taken into account implicitly via the extracted features. For more accurate results we propose the *occupancy planes* (OPlanes) representation, which enables to formulate single-view RGB-D human reconstruction as occupancy prediction on planes which slice through the camera's view frustum. Such a representation provides more flexibility than voxel grids and enables to better leverage correlations than per-point classification. On the challenging S3D data we observe a simple classifier based on the OPlanes representation to yield compelling results, especially in difficult situations with partial occlusions due to other objects and partial visibility, which haven't been addressed by prior work.

## 1 Introduction

Reconstructing the 3D shape of humans (Guan et al. 2009; Tong et al. 2012; Bogo et al. 2016; Lassner et al. 2017; Guler and Kokkinos 2019; Xiang, Joo, and Sheikh 2019; Yu et al. 2017; Varol et al. 2018; Zheng et al. 2019; Saito et al. 2019; Ren, Zhao, and Schwing 2021) has attracted extensive attention. It enables numerous applications such as AR/VR content creation, virtual try-on in the fashion industry, and image/video editing. In this paper, we focus on the task of human reconstruction from a single RGB-D image and its camera information. A single-view setup is simple and alleviates the tedious capturing of sequences from multiple locations. However, while capturing of data is simplified, the task is more challenging because of the ambiguity in inferring the invisible part of the 3D human shape given only the camera facing part.

A promising direction for 3D reconstruction of a human from a single image is the use of implicit functions. Prior works which use implicit functions (Saito et al. 2019, 2020) have shown compelling results on single view human reconstruction. In common to all these prior works is the use of per-point classification. Specifically, prior works formulate single-view human reconstruction by first projecting a point onto the image. A pixel aligned feature is subsequently extracted and used to classify whether the 3D point is inside or outside the observed human.

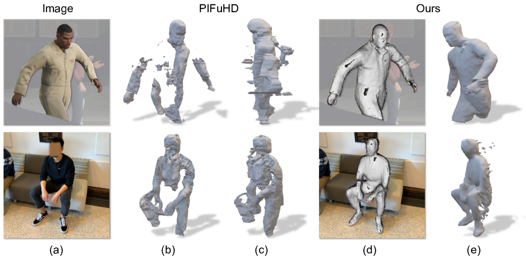

Figure 1: Reconstruction results compared to PIFuHD (Saito et al. 2020) on S3D (Hu et al. 2021) data (1$^{\text{st}}$ row) and real-world data (2$^{\text{nd}}$ row). **(a)**: input image; **(b)** and **(c)**: results from PIFuHD. It struggles with partial visibility and non-standing poses; **(d)**: our reconstruction overlaying the input image with perspective camera projection. **(e)**: another view of our reconstruction.

While such a formulation is compelling, it is challenged by the fact that occupancy for every point is essentially predicted independently (Saito et al. 2019, 2020; Mescheder et al. 2019). While image features permit to take context into account implicitly, the point-wise prediction often remains noisy. This is particularly true if the data is more challenging than the one used in prior works. For instance, we observe prior works to struggle if humans are only partially visible, *i.e.*, if we are interested in only reconstructing the observed part of a partially occluded person.

To address this concern, we propose to formulate single-view RGB-D human reconstruction as an occupancy-plane-prediction task. For this, we introduce the novel *occupancy planes* (OPlanes) representation. The OPlanes representation consists of multiple image-like planes which slice in a fronto-parallel manner through the camera's view frustum and indicate at every pixel location the occupancy for the corresponding 3D point. Importantly, the OPlanes representation permits to adaptively adjust the number and location of the occupancy planes during inference and training. Therefore, its resolution is more flexible than that of a classical voxel grid representation (Varol et al. 2018; Maturana and Scherer 2015). Moreover, the plane structure naturally enables the model to benefit more directly from correlations between neighboring locations within a plane than unstructured representations like point clouds (Fan, Su, and Guibas 2017; Qi et al. 2017) and implicit representations with per-point queries (Saito et al. 2019, 2020; Mescheder et al. 2019). To summarize, our contributions are two-fold: 1) we propose the OPlanes representation for single view RGB-D human reconstruction; 2) we verify that exploiting correlations within planes is beneficial for 3D human shape reconstruction as illustrated in Fig. 1.

We evaluate the proposed approach on the challenging S3D (Hu et al. 2021) data and observe improvements over prior reconstruction work (Saito et al. 2020; Chibane, Alldieck, and Pons-Moll 2020) by a margin, particularly for occluded or partially visible humans. We also provide a comprehensive analysis to validate each of the design choices and results on real-world data.

## 2 Related Work

3D human reconstruction (Guan et al. 2009; Tong et al. 2012; Yang et al. 2016; Zhang et al. 2017; Bogo et al. 2016; Lassner et al. 2017; Guler and Kokkinos 2019; Kolotouros, Pavlakos, and Daniilidis 2019; Xiang, Joo, and Sheikh 2019; Xu, Zhu, and Tung 2019; Yu et al. 2017; Zheng et al. 2019; Varol et al. 2018) has been extensively studied for the last few decades. We first discuss the most relevant works on single-view human reconstruction (Gabeur et al. 2019) and group them into two categories, template-based models and non-parametric models. Then we review the common 3D representations.

**Template-based models for single-view human reconstruction.** Parametric human models such as SCAPE (Anguelov et al. 2005) and SMPL (Bogo et al. 2016) are widely used for human reconstruction. These methods (Kanazawa et al. 2018; Varol et al. 2018; Zheng et al. 2019; Huang et al. 2020) use the human body shape as a prior to regularize the prediction space and predict or fit the low-dimensional parameters of a human body model. Specifically, HMR (Kanazawa et al. 2018) learns to predict the human shape by regressing the parameters of SMPL from a single image. BodyNet (Varol et al. 2018) predicts a 3D voxel grid of the human shape and fits the SMPL body model to the predicted volumetric shape. DeepHuman (Zheng et al. 2019) utilizes the SMPL model as an initialization and further refines it with deep nets. Although parametric human models are deformable and can capture various complex human body poses and different body measurements, these methods generally do not consider surface details such as hair, clothing as well as accessories.

**Non-parametric models for single-view human reconstruction.** Non-parametric methods for human reconstruction (Saito et al. 2019, 2020; He et al. 2020; Gabeur et al. 2019; Wang et al. 2020; Ren, Zhao, and Schwing 2021) gained popularity recently as they are more flexible in recovering surface details compared to template-based methods. Among them, using implicit function (Sclaroff and Pentland 1991) to predict human shape achieves state-of-the-art results (Saito et al. 2020), showing that the expressivity of neural nets enables to memorize the human body shape. To achieve this, the task is usually formulated as a per-point classification, *i.e.*, classifying every point in a 3D space independently into either inside or outside of the observed body. For this, PIFu (Saito et al. 2019) reconstructs the human shape from an image encoded into a feature map, from which it learns an implicit function to predict per-point occupancy. PIFuHD (Saito et al. 2020) employs a two level implicit predictor and incorporates normal information to recover high quality surfaces. GeoPIFu (He et al. 2020) learns additionally latent voxel features to encourage shape regularization. Hybrid methods have also been studied (Huang et al. 2020; Cao et al. 2022), combining human shape templates with a non-parametric framework. These methods usually yield reconstruction results with surface details. However, in common to all the aforementioned methods, the per-point classification formulation doesn't *directly* take correlations between neighboring 3D points into account. Therefore, predictions remain noisy, particularly in challenging situations with occlusions or partial visibility. Because of this, prior works usually consider images where the whole human body is visible and roughly centered. In contrast, for broader applicability and more accurate results in challenging situations, we propose the OPlanes representation.

**3D representations.** Various 3D representation have been developed, such as voxel grids (Varol et al. 2018; Maturana and Scherer 2015; Lombardi et al. 2019; Ren et al. 2022), meshes (Lin, Wang, and Liu 2021; Wang et al. 2018; Wu et al. 2022), point clouds (Qi et al. 2017; Fan, Su, and Guibas 2017; Wu et al. 2020; Aliev et al. 2020), implicit functions (Mescheder et al. 2019; Saito et al. 2019, 2020; He et al. 2020; Hong et al. 2021; Peng et al. 2021), layered representations (Shade et al. 1998; Zhou et al. 2018; Srinivasan et al. 2019; Tucker and Snavely 2020; Zhao et al. 2022) and hierarchical representations (Meagher 1982; Häne, Tulsiani, and Malik 2017; Yu et al. 2021). For human body shape reconstruction, template-based representations (Anguelov et al. 2005; Bogo et al. 2016; Pavlakos et al. 2019; Osman, Bolkart, and Black 2020) are also popular. The proposed OPlanes representation combines the benefits of both layered and implicit representations. Compared to voxel grids, OPlanes is more flexible, enabling prediction at different resolutions due to its implicit formulation of occupancy-prediction of an entire plane. Compared to unstructured representations such as implicit functions or point clouds, OPlanes benefits from its increased context of a per-plane prediction as opposed to a per-pixel/point prediction. Concurrently, Fourier occupancy field (FOF) (Feng et al. 2022) proposes to use a 2D field orthogonal to the view direction to represent the occupancy. Different from FOF, where coefficients for Fourier basis functions are estimated for each position on the 2D field, OPlanes directly regress to the occupancy value.

## 3 Method

### 3.1 Overview

Given an RGB image, a depth map, a mask highlighting the human of interest in the image as well as the intrinsic camera parameters, our goal is to reconstruct a spatially-

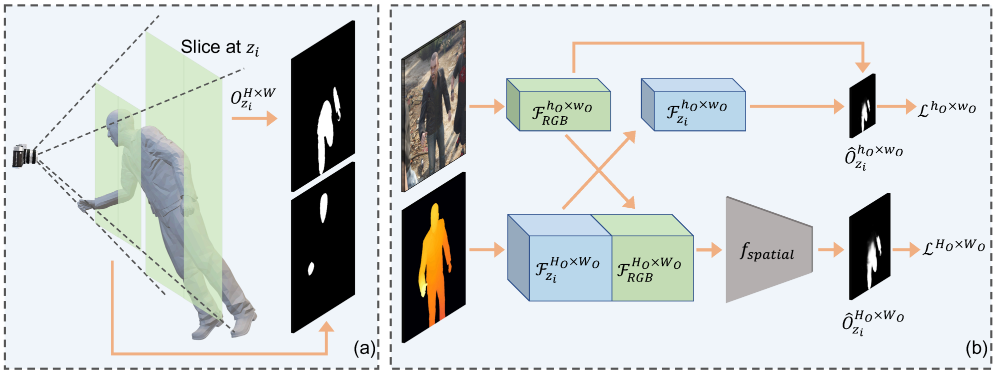
Slice at $z_i$
$O_{z_i}^{H\times W}$
(a)
$\mathcal{F}_{RGB}^{h_O\times w_O}$
$\mathcal{F}_{z_i}^{h_O\times w_O}$
$\mathcal{L}^{h_O\times w_O}$
$\widehat{O}_{z_i}^{h_O\times w_O}$
$\mathcal{F}_{z_i}^{H_O\times W_O}$
$\mathcal{F}_{RGB}^{H_O\times W_O}$
$f_{spatial}$
$\mathcal{L}^{H_O\times W_O}$
$\widehat{O}_{z_i}^{H_O\times W_O}$
(b)

Figure 2: Occupancy planes (OPlanes) overview. **(a)** Occupancy plane $O_{z_i}^{H\times W}$ stores the occupancy information (black plane on the right) at a specific slice (light green plane on the left) in the view frustum. White pixels indicate "inside" the mesh (Sec. 3.2). **(b)** Given RGB-D data and a mask, our approach takes a specific depth $z_i$ as input and predicts the corresponding occupancy plane $\widehat{O}_{z_i}^{H_O\times W_O}$ (Sec. 3.3). The convolutional neural network $f_{\text{spatial}}$ explicitly considers context information for each pixel on the occupancy plane, which we find to be beneficial. During training, we not only supervise $\widehat{O}_{z_i}^{H_O\times W_O}$ through loss $\mathcal{L}^{H_O\times W_O}$ but we also supervise the intermediate feature $\widehat{O}_{z_i}^{h_O\times w_O}$ with loss $\mathcal{L}^{h_O\times w_O}$ (Sec. 3.4).

aligned human mesh $\mathcal{M}$.

To generate the mesh $\mathcal{M}$, we introduce the *Occupancy Planes* (OPlanes) representation, a plane-based representation of the geometry at various depth levels. This representation is inspired by classical semantic segmentation, but extends segmentation planes to various depth levels. OPlanes can be used to generate an occupancy grid, from which the mesh $\mathcal{M}$ is obtained via a marching cube (Lorensen and Cline 1987) algorithm. We illustrate the framework in Fig. 2.

In the following we first introduce OPlanes in Sec. 3.2. Subsequently, Sec. 3.3 details the developed deep net to predict OPlanes, while training of the deep net is discussed in Sec. 3.4. Finally, Sec. 3.5 provides details about generation of the mesh from the predicted OPlanes.

## 3.2 Occupancy Planes (OPlanes) Representation

Given an image capturing a human of interest, *occupancy planes* (OPlanes) store the occupancy information of that human in the camera's view frustum. For this, the OPlanes representation consists of several 2D images, each of which store the mesh occupancy at a specific fronto-parallel slice through the camera's view frustum.

Concretely, let $[z_{\text{min}}, z_{\text{max}}]$ be the range of depth we are interested in, *i.e.*, $z_{\text{min}}, z_{\text{max}}$ are the near-plane and far-plane of the view frustum of interest. Further, let the set $\mathcal{Z}_N \triangleq \{z_1, \dots, z_N \,|\, z_{\text{min}} \leq z_i \leq z_{\text{max}}, \forall i\}$ contain the sampled depths of interest. The OPlanes representation $\mathcal{O}_{\mathcal{Z}_N}^{H\times W}$ for the depths of interest stored in $\mathcal{Z}_N$ refers to the set of planes

$$\mathcal{O}_{\mathcal{Z}_N}^{H\times W} \triangleq \left\{O_{z_1}^{H\times W}, O_{z_2}^{H\times W}, \dots, O_{z_N}^{H\times W} \,|\, z_i \in \mathcal{Z}_N\right\}. \tag{1}$$

Each OPlane $O_z^{H\times W} \in \{0,1\}^{H\times W}$ is a binary image of height $H$ and width $W$.

To compute the ground-truth binary values of the occupancy plane $O_z^{H\times W}$ at depth $z$, let $[x, y, 1]$ be a homogeneous pixel coordinate on the given image $I$. Given a depth $z$ of interest, the homogeneous pixel coordinate can be unprojected into the 3D space coordinate $[x_z, y_z, z] = z \cdot \pi^{-1}([x, y, 1])$, where $\pi(\cdot)$ denotes the perspective projection. Note, this unprojection differs from prior human mesh reconstruction works (Zheng et al. 2019; Saito et al. 2019, 2020; He et al. 2020) that assume an orthogonal projection with a weak-perspective camera. Instead, we utilize a perspective camera for more general use cases.

From 3D meshes available in the training data we obtain a 3D point's ground-truth occupancy value as follows:

$$o([x_z, y_z, z]) = \begin{cases} 1, & \text{if } [x_z, y_z, z] \text{ inside the object,} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The value of the occupancy plane $O_z^{H\times W} \in \{0,1\}^{H\times W}$ at depth $z$ and at pixel location $x, y$ can be obtained from the ground-truth occupancy value via

$$O_z^{H\times W}[x, y] = o([x_z, y_z, z]), \tag{3}$$

where $O_z^{H\times W}[x, y]$ denotes the occupancy plane value of pixel $[x, y]$.

## 3.3 Occupancy Planes Prediction

At test time, ground-truth meshes are not available. Instead we are interested in predicting the OPlanes from 1) a given RGB image $I_{\text{RGB}} \in \mathbb{R}^{H\times W\times 3}$ illustrating a human, 2) a depth map $\mathtt{Depth} \in \mathbb{R}^{H\times W}$, 3) a mask $\mathtt{Mask} \in \{0,1\}^{H\times W}$, and 4) the calibrated camera's perspective projection $\pi$. Specifically, let $H_O \times W_O$ be the operating resolution where $H_O \leq H$ and $W_O \leq W$. We use a deep net to predict $N$ occupancy planes $\widehat{\mathcal{O}}^{H_O\times W_O} = \{\widehat{O}_{z_i}^{H_O\times W_O}\}_{i=1}^{N}$ at various depth levels $z_i \; \forall i \in \{1, \dots, N\}$, via

$$\widehat{O}_{z_i}^{H_O\times W_O} = f_{\text{spatial}}([\mathcal{F}_{\text{RGB}}^{H_O\times W_O}; \mathcal{F}_{z_i}^{H_O\times W_O}]). \tag{4}$$

Here, $[\cdot;\cdot]$ denotes the concatenation operation along the channel dimension.

In order to resolve the depth ambiguity, we design $f_{\text{spatial}}(\cdot)$ to be a simple fully convolutional network that aims to fuse spatial neighborhood information within each occupancy plane prediction $\widehat{O}_{z_i}^{H_O\times W_O}$. Note that this design differs from prior work, which predicts the occupancy for each point independently. In contrast, we find spatial neighborhood information is useful to improve occupancy prediction accuracy.

For an accurate prediction, the fully convolutional net $f_{\text{spatial}}(\cdot)$ operates on image features $\mathcal{F}_{\text{RGB}}^{H_O\times W_O} \in \mathbb{R}^{H_O\times W_O\times C}$ and depth features $\mathcal{F}_{z_i}^{H_O\times W_O} \in \mathbb{R}^{H_O\times W_O\times C}$. In the following we discuss the deep nets to compute the image features $\mathcal{F}_{\text{RGB}}^{H_O\times W_O}$ and the depth features $\mathcal{F}_{z_i}^{H_O\times W_O}$.

**Image feature $\mathcal{F}_{\text{RGB}}$.** The image feature $\mathcal{F}_{\text{RGB}}^{H_O\times W_O}$ is obtained by bilinearly upsampling a low-resolution feature map to the operating resolution $H_O \times W_O$. Concretely,

$$\mathcal{F}_{\text{RGB}}^{H_O\times W_O} = \texttt{UpSample}_{h_O\times w_O \to H_O\times W_O}(\mathcal{F}_{\text{RGB}}^{h_O\times w_O}), \quad (5)$$

where $\mathcal{F}_{\text{RGB}}^{h_O\times w_O} \in \mathbb{R}^{h_O\times w_O\times C}$ is the RGB feature at the coarse resolution of $h_O \times w_O$. $\texttt{UpSample}_{h_O\times w_O\to H_O\times W_O}$ refers to the standard bilinear upsampling.

The coarse resolution RGB feature is obtained via

$$\mathcal{F}_{\text{RGB}}^{h_O\times w_O} = f_{\text{RGB}}(f_{\text{FPN}}(\hat{I}_{\text{RGB}})), \quad (6)$$

where $\hat{I}_{\text{RGB}} \in \mathbb{R}^{H\times W\times 5}$ is the concatenation of $I_{\text{RGB}}$ and two simple features (see appendix). $f_{\text{FPN}}$ is the Feature Pyramid Network (FPN) backbone (Lin et al. 2017) and $f_{\text{RGB}}$ is another fully-convolutional network for further processing.

**Depth feature $\mathcal{F}_{z_i}$.** The depth feature $\mathcal{F}_{z_i}^{H_O\times W_O}$ for an occupancy plane at depth $z_i$ encodes for every pixel $[x, y]$ the difference between the query depth $z_i$ and the depth at which the object first intersects with the camera ray. Concretely, we obtain the depth feature via

$$\mathcal{F}_{z_i}^{H_O\times W_O} = f_{\text{depth}}(I_{z_i}^{H_O\times W_O}), \quad (7)$$

where $f_{\text{depth}}$ is a fully convolutional network to process the depth difference image $I_{z_i}^{H_O\times W_O}$.

The depth difference image $I_{z_i}^{H_O\times W_O}$ is constructed to capture the difference between the query depth $z_i$ and the depth at which the object first intersects with the camera ray. *I.e.*, for each pixel $[x, y]$,

$$I_{z_i}^{H_O\times W_O}[x, y] = \texttt{PE}(z_i - \texttt{Depth}[x, y]), \quad (8)$$

where $\texttt{PE}(\cdot)$ is the positional encoding operation (Vaswani et al. 2017). Intuitively, the depth difference image $I_{z_i}$ represents how far every point on the plane at depth $z_i$ is behind or in front of the front surface of the observed human.

### 3.4 Training

The developed deep net to predict OPlanes is fully differentiable. We use $\theta$ to subsume all trainable parameters within the spatial network $f_{\text{spatial}}$ (Eq. (4)), the FPN network $f_{\text{FPN}}$, the RGB network $f_{\text{RGB}}$ (Eq. (6)), and the depth network $f_{\text{depth}}$ (Eq. (7)). Further, we use $\widehat{\mathcal{O}}_\theta$ to refer to the predicted occupancy planes when using the parameter vector $\theta$. We train the deep net to predict OPlanes end-to-end with two losses by addressing

$$\min_\theta \mathcal{L}_\theta^{H_O\times W_O} + \mathcal{L}_\theta^{h_O\times w_O}. \quad (9)$$

Here, $\mathcal{L}_\theta^{H_O\times W_O}$ is the loss computed at the final prediction resolution of $H_O \times W_O$, while $\mathcal{L}_\theta^{h_O\times w_O}$ is used to supervise intermediate features at the resolution of $h_O \times w_O$. We discuss both losses next.

**Final prediction supervision via $\mathcal{L}_\theta^{H_O\times W_O}$.** During training, we randomly sample $N$ depth values from the view frustum range $[z_{\text{min}}, z_{\text{max}}]$ to obtain the set of depth values of interest $\mathcal{Z}_N$ (Sec. 3.2). For this, we use $z_{\text{min}} = \min\{\texttt{Depth}[x, y] \,|\, \texttt{Mask}[x, y] == 1\}$ by only considering depth information within the target mask. Essentially, we find the depth value that is closest to the camera. We set $z_{\text{max}} = z_{\text{min}} + z_{\text{range}}$, where $z_{\text{range}}$ marks the depth range we are interested in. During training, $z_{\text{range}}$ is computed from the ground-truth range which covers the target mesh. During inference, we set $z_{\text{range}} = 2$ meters to cover the shapes and gestures of most humans.

The high resolution supervision loss $\mathcal{L}_\theta^{H_O\times W_O}$ consists of two terms. Namely $\mathcal{L}_\theta^{H_O\times W_O} \triangleq$

$$\lambda_{\text{BCE}} \cdot \mathcal{L}_{\text{BCE}}(\mathcal{O}^{H_O\times W_O}, \widehat{\mathcal{O}}_\theta^{H_O\times W_O}, \texttt{Mask}^{H_O\times W_O}, z_{\text{min}}, z_{\text{max}}) + \lambda_{\text{DICE}} \cdot \mathcal{L}_{\text{DICE}}(\mathcal{O}^{H_O\times W_O}, \widehat{\mathcal{O}}_\theta^{H_O\times W_O}, \texttt{Mask}^{H_O\times W_O}, z_{\text{min}}, z_{\text{max}}). \quad (10)$$

Here, $\mathcal{L}_{\text{BCE}}$ is the binary cross entropy (BCE) loss while $\mathcal{L}_{\text{DICE}}$ is the DICE loss (Milletari, Navab, and Ahmadi 2016). Both losses operate on the ground-truth OPlanes $\mathcal{O}^{H_O\times W_O}$ downsampled from the original resolution $H \times W$, the OPlanes $\widehat{\mathcal{O}}_\theta^{H_O\times W_O}$ predicted with the current deep net parameters $\theta$, and the human mask $\texttt{Mask}^{H_O\times W_O}$ downsampled from the raw mask. Note, we only consider points behind the human's front surface when computing the loss, *i.e.*, on a plane $\widehat{O}_{z_i}$, we only consider $\{[x, y]|z_i \geq \texttt{Detph}[x, y]\}$. For readability, we drop the superscript $H_O \times W_O$ in the following. The BCE loss is computed via $\mathcal{L}_{\text{BCE}} =$

$$\frac{1}{|\mathcal{Z}_N| \cdot \texttt{Sum}(\texttt{Mask})} \sum_{\substack{z_i\in\mathcal{Z}_N \\ x,y:\texttt{Mask}[x,y]=1}} \Bigg( O_{z_i}[x, y] \cdot \log \widehat{O}_{z_i}[x, y] + (1 - O_{z_i}[x, y]) \cdot \log(1 - \widehat{O}_{z_i}[x, y]) \Bigg), \quad (11)$$

where $\texttt{Sum}(\texttt{Mask})$ is the number of pixels within the target's segmentation mask and $x, y : \texttt{Mask}[x, y] = 1$ emphasizes that we only compute BCE loss on pixels within the mask.

Moreover, thanks to the occupancy plane representation inspired by semantic segmentation tasks, we can utilize the DICE loss from the semantic segmentation community to supervise the occupancy training. Specifically, we use $\mathcal{L}_{\text{DICE}} =$

$$\frac{1}{|\mathcal{Z}_N|} \sum_{z_i\in\mathcal{Z}_N} \frac{2 \cdot \texttt{Sum}(\texttt{Mask} \cdot O_{z_i} \cdot \widehat{O}_{z_i})}{\texttt{Sum}(\texttt{Mask} \cdot O_{z_i}) + \texttt{Sum}(\texttt{Mask} \cdot \widehat{O}_{z_i})}. \quad (12)$$

This is useful because there can be a strong imbalance between the number of positive and negative labels in an OPlane $O_{z_i}$ due to human gestures. The DICE loss has been shown to compellingly deal with such situations (Milletari, Navab, and Ahmadi 2016).

**Intermediate feature supervision via $\mathcal{L}_\theta^{h_O\times w_O}$.** Besides supervision of the final occupancy image $\widehat{O}_{z_i}$ discussed in the preceding section, we also supervise the intermediate features $\mathcal{F}_{\text{RGB}}^{h_O\times w_O}$ (Eq. (6)) via the loss $\mathcal{L}_\theta^{h_O\times w_O}$. Analogously to the high-resolution loss, we use two terms, *i.e.*, $\mathcal{L}_\theta^{h_O\times w_O} \triangleq$

$$\lambda_{\text{BCE}} \cdot \mathcal{L}_{\text{BCE}}(\mathcal{O}^{h_O\times w_O}, \widehat{\mathcal{O}}_\theta^{h_O\times w_O}, \texttt{Mask}^{h_O\times w_O}, z_{\text{min}}, z_{\text{max}})+$$
$$\lambda_{\text{DICE}} \cdot \mathcal{L}_{\text{DICE}}(\mathcal{O}^{h_O\times w_O}, \widehat{\mathcal{O}}_\theta^{h_O\times w_O}, \texttt{Mask}^{h_O\times w_O}, z_{\text{min}}, z_{\text{max}}). \quad (13)$$

Different from the high-resolution representation, we predict the OPlanes representation at the coarse resolution $h_O \times w_O$ via

$$\widehat{O}_{z_i}^{h_O\times w_O}[x,y] = \langle \mathcal{F}_{\text{RGB}}^{h_O\times w_O}[x,y,\cdot],\, \mathcal{F}_{z_i}^{h_O\times w_O}[x,y,\cdot]\rangle, \quad (14)$$

where $\langle\cdot,\cdot\rangle$ is the inner-product operation and $\mathcal{F}_{\text{RGB}}^{h_O\times w_O}[x,y,\cdot]$ represents the feature vector at the pixel location $[x,y]$. To obtain $\mathcal{F}_{z_i}^{h_O\times w_O}$, we feed the down-sampled difference image $I_{z_i}^{h_O\times w_O}$ into $f_{\text{depth}}$. Intuitively, we use the inner product to encourage the image feature $\mathcal{F}_{\text{RGB}}^{h_O\times w_O}$ to be strongly correlated to information from the depth feature $\mathcal{F}_{z_i}^{h_O\times w_O}$.

### 3.5 Inference

During inference, to reconstruct a mesh from predicted OPlanes $\widehat{\mathcal{O}}$, we first establish an occupancy grid before running a marching cube (Lorensen and Cline 1987) algorithm to extract the isosurface. Specifically, we uniformly sample $N$ depths in the view frustum between depth range $[z_{\text{min}}, z_{\text{min}} + 2.0]$, *e.g.*, $N = 256$. Here 2.0 is a heuristic depth range which covers most human poses (Sec. 3.4). The network predicts an occupancy for each pixel on those $N$ planes. Importantly, since OPlanes represent occupancy corresponding to slices through the view frustum, a marching cube algorithm is not directly applicable. Instead, we first establish a voxel grid to cover the view frustum between $[z_{\text{min}}, z_{\text{min}} + 2.0]$. Each voxel's occupancy is sampled from the predicted OPlanes before a marching cube method is used. We emphasize that the number of planes do not need to be the same during training and inference, which we will show later. This ensures that the OPlanes representation is memory efficient at training time while enabling accurate reconstruction at inference time.

## 4 Experiments

### 4.1 Implementation Details

Here we introduce key implementation details. Please see the appendix for more information. During training, the input has a resolution of $H = 512$ and $W = 512$. We operate at $H_O = 256$, $W_O = 256$, while the intermediate resolution is $h_O = 128$ and $w_O = 128$. During training, for each mesh, we randomly sample $N = 10$ planes in the range of $[z_{\text{min}}, z_{\text{max}}]$ at each training iteration. *I.e.*, the set $\mathcal{Z}_N$ contains 10 depth values. As mentioned in Sec. 3.4, during training, we set $z_{\text{max}}$ to be the ground-truth mesh's furthest depth.

The four deep nets, which we detail next, are mostly convolutional. We use *(in, out, k)* to denote the input/output channels and the kernel size of a convolutional layer.

**Spatial network** $f_{\textbf{spatial}}$ (Eq. (4)): It's a three-layer convolutional neural net (CNN) with a configuration of (256, 128, 3), (128, 128, 3), (128, 1, 1). We use group norm (Wu and He 2018) and ReLU activation.

**Feature pyramid network** $f_{\textbf{FPN}}$ (Eq. (6)): We use ResNet50 (He et al. 2016) as the backbone of our FPN network. We use the output of each stage's last residual block as introduced in (Lin et al. 2017). The final output of this FPN has 256 channels and a resolution of $\frac{H}{4} \times \frac{W}{4}$.

**RGB network** $f_{\textbf{RGB}}$ (Eq. (6)): It's a three-layer CNN with a configuration of (256, 128, 3), (128, 128, 3), (128, 128, 1). We use group norm (Wu and He 2018) and ReLU activation.

**Positional encoding** $\texttt{PE}$ (Eq. (8)): We follow (Vaswani et al. 2017) to define $\texttt{PE}(\text{pos}) =$

$$(\texttt{PE}_0(\text{pos}), \texttt{PE}_1(\text{pos}), \ldots, \texttt{PE}_{63}(\text{pos}), \texttt{PE}_{64}(\text{pos}))\,, \quad (15)$$

where $\texttt{PE}_{2t}(\text{pos}) = \sin(\frac{50\cdot\text{pos}}{200^{2t/64}})$ and $\texttt{PE}_{2t+1}(\text{pos}) = \cos(\frac{50\cdot\text{pos}}{200^{2t/64}})$.

**Depth difference network** $f_{\textbf{depth}}$ (Eq. (7)): It's a two-layer CNN with a configuration of (64, 128, 1), (128, 128, 1). We use group norm (Wu and He 2018) and ReLU activation.

To train the networks, we use the Adam (Kingma and Ba 2015) optimizer with a learning rate of 0.001. We set $\lambda_{\text{BCE}} = 1.0$ and $\lambda_{\text{DICE}} = 1.0$ (Eq. (10) and Eq. (13)). We set the batch size to 4 and train for 15 epochs. It takes around 22 hours to complete the training using an AMD EPYC 7543 32-Core Processor and an Nvidia RTX A6000 GPU.

### 4.2 Experimental Setup

**Dataset.** We utilize S3D (Hu et al. 2021) to train our OPlanes-based human reconstruction model. S3D is a photo-realistic synthetic dataset built on the game GTA-V, providing ground-truth meshes together with masks and depths. To construct our train and test set, we sample 27588 and 4300 meshes from its train and validation split respectively. This dataset differs from counterparts in prior works (Saito et al. 2019, 2020; He et al. 2020; Alldieck, Zanfir, and Sminchisescu 2022): there are no constraints on the appearance of humans in the images. In our dataset, humans appear with any gestures, any sizes, any position, and any level of occlusion. In contrast, humans in datasets of prior work usually appear in an upright position and are mostly centered in an image while exhibiting little to no occlusion. We think this setup strengthens the generalization ability. See Fig. 3 for some examples.

**Baselines.** We compare to PIFuHD (Saito et al. 2020) and IF-Net (Chibane, Alldieck, and Pons-Moll 2020). **1) PIFuHD:** since there is no training code available, we test with the officially-released checkpoints. Following the author's suggestion in the public code repository[1] to improve the reconstruction quality, we 1.1) remove the background with the ground-truth mask; 1.2) apply human pose detection (Osokin 2018) and crop the image accordingly to place the human of interest in the center of the image. **2) IF-Net:** we evaluate with the officially-released checkpoint. IF-Net uses a 3D voxel grid representation. We set the resolution of the grid to 256 to align with the pretrained checkpoint.

[1] https://github.com/facebookresearch/pifuhd

Table 1: **Quantitative results**. Each result averages three runs with different seeds and is reported in the format of mean±std. OPlanes improve upon PIFuHD by a margin (1st *vs*. 6th row) and outperform IF-Net in almost all metrics (2nd *vs*. 6th row). We also verify the design choices via an ablation study reported in the 3rd to 5th row. For all OPlanes results, we predict occupancy using 256 planes per mesh during inference, while using 10 or less planes per mesh when training.

| | | OPlane | $f_{\text{spatial}}$ Kernel Size | $\mathcal{L}_\theta^{h_O \times w_O}$ | #Planes in Train | IoU↑ | Cham-$\mathcal{L}_1$ ↓ | Normal Consist. ↑ |
|---|---|---|---|---|---|---|---|---|
| 1 | PIFuHD (Saito et al. 2020) | ✗ | - | - | - | 0.428 | 0.332 | 0.677 |
| 2 | IF-Net (Chibane, Alldieck, and Pons-Moll 2020) | ✗ | - | - | - | 0.584 | 0.216 | **0.802** |
| 3 | NoNeighborInfo | ✓ | $1 \times 1$ | ✓ | 5 | 0.679±0.013 | 0.158±0.007 | 0.738±0.005 |
| 4 | NoInterSupervision | ✓ | $3 \times 3$ | ✗ | 5 | 0.681±0.013 | 0.161±0.008 | 0.739±0.005 |
| 5 | LessPlanes | ✓ | $3 \times 3$ | ✓ | 5 | 0.684±0.013 | 0.158±0.008 | 0.747±0.005 |
| 6 | OursFull | ✓ | $3 \times 3$ | ✓ | 10 | **0.691**±0.013 | **0.155**±0.008 | 0.749±0.005 |

**Evaluation metrics.** We focus on evaluating the quality of the reconstructed geometry. Following prior works (Mescheder et al. 2019; Saito et al. 2019, 2020; He et al. 2020; Alldieck, Zanfir, and Sminchisescu 2022; Huang et al. 2020; He et al. 2021), we report the Volumetric Intersection over Union (IoU), the bi-directional Chamfer-$\mathcal{L}_1$ distance, and the Normal Consistency. Please refer to the supplementary material of (Mescheder et al. 2019) for more details on these metrics. To compute the IoU, we need a finite space to sample points. Since humans in our data appear anywhere in 3D space, the implicit assumption of prior works (Saito et al. 2019, 2020; He et al. 2020) that there exists a fixed bounding box for all objects does not hold. Instead, we use the view frustum between depth $z_{\text{min}}$ and $z_{\text{max}}$ as the bounding box. Note, for evaluation purposes, $z_{\text{max}}$ utilizes the heuristic $z_{\text{range}}$ of 2.0 meters (Sec. 3.4). We sample 100k points for an unbiased estimation. When computing the Chamfer distance, we need to avoid that the final aggregated results are skewed by a scale discrepancy between different objects. We follow (Fan, Su, and Guibas 2017; Mescheder et al. 2019) and let $\frac{1}{10}$ of each object's ground-truth bounding box's longest edge correspond to a unit of one when computing Chamfer-$\mathcal{L}_1$. To resolve the discrepancy between the orthogonal projection and the perspective projection, we utilize the iterative-closest-point (ICP) (Besl and McKay 1992) algorithm to register the reconstruction of baselines to the ground-truth, following (Alldieck, Zanfir, and Sminchisescu 2022). ICP is not applied to our OPlanes method since we directly reconstruct the human in the camera coordinate system.

## 4.3 Quantitative Results

In Tab. 1 we provide quantitative results, comparing to baselines in the 1st/2nd *vs*. 6th row. For a fair comparison when computing the results, we reconstruct the final geometry in a $256^3$ grid. Although 256 OPlanes are inferred, we train with only 10 planes per mesh in each iteration.

**PIFuHD (Saito et al. 2020):** The OPlanes representation outperforms the PIFuHD results by a margin. Specifically, our results exhibit a larger volume overlap with the ground-truth (0.691 *vs*. 0.428 on IoU, ↑ is better), more completeness and accuracy (0.155 *vs*. 0.332 on Chamfer distance, ↓ is better), and more fine-grained details (0.749 *vs*. 0.677 on normal consistency, ↑ is better).

**IF-Net (Chibane, Alldieck, and Pons-Moll 2020):** We also compare to the depth-based single-view reconstruction approach IF-Net. The results are presented in row 2 *vs*. 6 in Tab. 1. We find that IF-Net struggles to reconstruct humans which are partly occluded or outside the field-of-view (see Fig. 3 and Fig. 4 for some examples). More importantly, we observe IF-Net to yield inferior results with respect to IoU (0.584 *vs*. 0.691, ↑ is better) and Chamfer distance (0.216 *vs*. 0.155, ↓ is better). Notably, we find the high normal consistency of IF-Net to be due to the high-resolution voxel grid, which provides more details.

## 4.4 Analysis

To verify design choices, we conduct ablation studies. We report the results in Tab. 1's 3rd to 5th row.

**Per-point classification is not all you need:** To understand whether neighboring information is needed, we replace the $3 \times 3$ kernel in $f_{\text{spatial}}$ (Eq. (4), Sec. 4.1) with a $1 \times 1$ kernel, which essentially conducts per-point classification for each pixel on the OPlane. Comparing the 3rd *vs*. 5th row in Tab. 1 corroborates the importance of context as per-point classification yields inferior results. This shows that the conventional way to treat shape reconstruction as a point classification problem (Saito et al. 2019, 2020; He et al. 2020) may be sub-optimal. Specifically, without directly taking into account the context information, we observe lower IoU (0.679 *vs*. 0.684) and less normal consistency (0.738 *vs*. 0.747).

**Intermediate supervision is important:** To understand whether the supervision of intermediate features is needed, we train our OPlanes without $\mathcal{L}_\theta^{h_O \times w_O}$ (Eq. (13)). The results in Tab. 1's 4th *vs*. 5th row verify the benefits of intermediate supervision. Concretely, with intermediate feature supervision, we obtain a better IoU (0.684 *vs*. 0.681, ↑ is better), an improved Chamfer distance (0.158 *vs*. 0.161, ↓ is better), and a better normal consistency (0.747 *vs*. 0.739, ↑ is better).

**Training with more planes is beneficial:** We are curious about whether training with less planes harms the performance of our OPlanes model. For this, we sample only 5 planes per mesh when training the OPlanes model. The results in the 5th *vs*. 6th row in Tab. 1 demonstrate that training with more planes yields better results. Concretely, with more planes, we obtain better IoU (0.691 *vs*. 0.684, ↑ is better), smaller Chamfer distance (0.155 *vs*. 0.158, ↓ is better), and better normal consistency (0.749 *vs*. 0.747, ↑ is better).

## 4.5 Qualitative Results

**S3D.** We provide qualitative results in Fig. 3. OPlanes successfully handle various human gestures and different levels

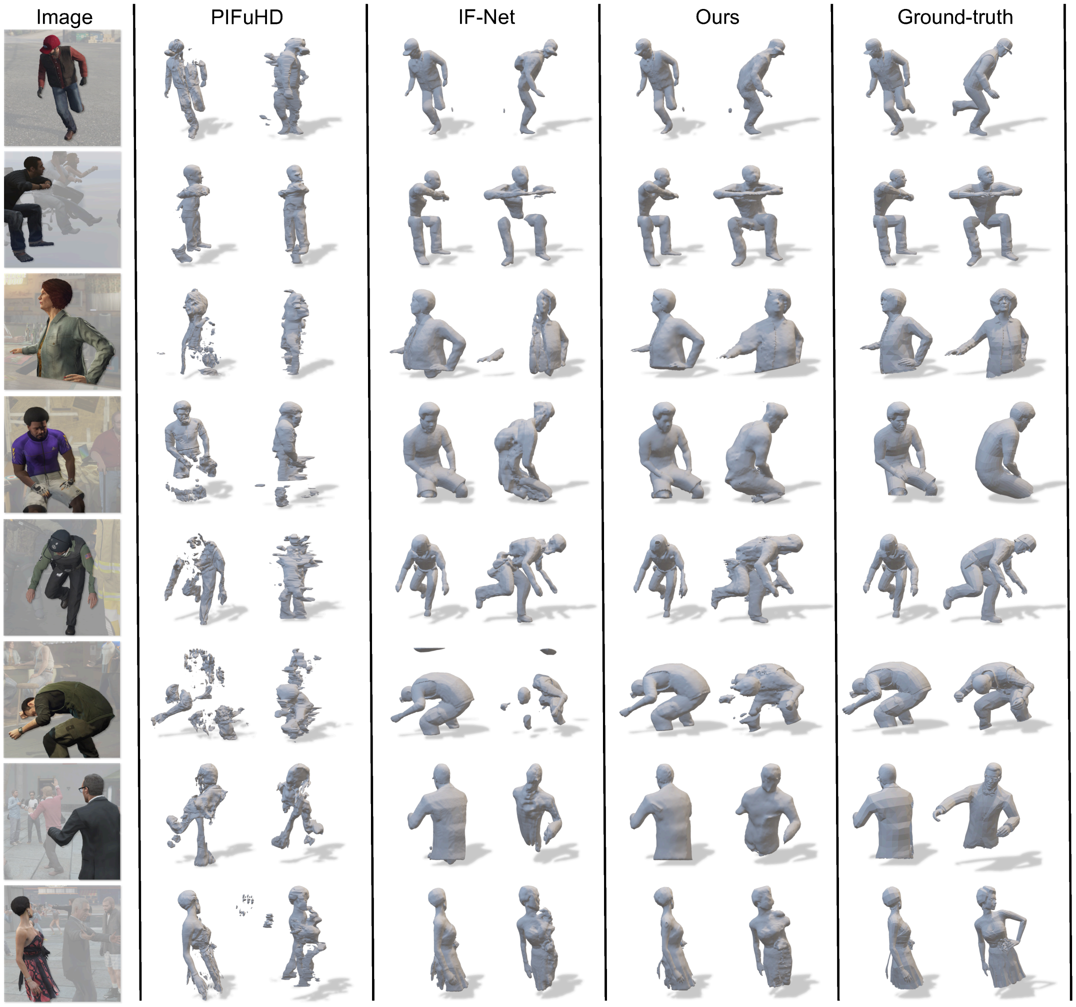

Figure 3: Qualitative results on S3D (Hu et al. 2021). For each reconstruction, we show two views. PIFuHD (Saito et al. 2020) and IF-Net (Chibane, Alldieck, and Pons-Moll 2020) struggle to obtain consistent geometry if parts of the human are invisible, while an OPlanes model faithfully reconstructs the visible portion.

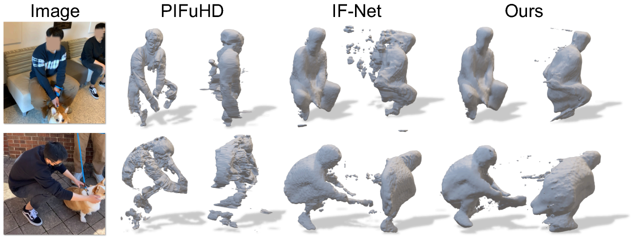

Figure 4: Transfer results on real-world RGB-D data captured with a 2020 iPad Pro.

of visibility while PIFuHD and IF-Net fail in those situations.

**Transferring Results to Real-World RGB-D Data.** In Fig. 4 we use real-world data collected in the wild to compare to PIFuHD and IF-Net. OPlanes results are obtained by directly applying the proposed OPlanes model trained on S3D, without fine-tuning or other adjustments. For this result, we use a 2020 iPad Pro equipped with a LiDAR sensor (ark 2021) and develop an iOS app to acquire the RGB-D images and camera matrices. The human masks are obtained by feeding RGB images into a Mask2Former (Cheng et al. 2022). 1) We observe PIFuHD results to be noisy and to contain holes. 2) For humans that are only partially visible, IF-Net seems to struggle. Our model benefits from the OPlanes representation which better exploits correlations within a plane. For this reason OPlanes better capture the human shape despite the model being trained on synthetic data.

## 5 Conclusion

We propose and study the occupancy planes (OPlanes) representation for reconstruction of 3D shapes of humans from a single RGB-D image. The resolution of OPlanes is more flexible than that of a classical voxel grid due to the implicit prediction of an entire plane. Moreover, prediction of an entire plane enables the model to benefit from correlations between predictions, which is harder to achieve for models which use implicit functions for individual 3D points. Due to these benefits we find OPlanes to excel in challenging situations, particularly for occluded or partially visible humans.

**Acknowledgements.** Work supported in part by NSF under Grants 1718221, 2008387, 2045586, 2106825, MRI 1725729, and NIFA award 2020-67021-32799. Thanks to NVIDIA for providing a GPU for debugging.

## References

2021. Augmented Reality - Apple Developer. https://developer.apple.com/augmented-reality/.

Aliev, K.-A.; Sevastopolsky, A.; Kolos, M.; Ulyanov, D.; and Lempitsky, V. 2020. Neural point-based graphics. In *ECCV*.

Alldieck, T.; Zanfir, M.; and Sminchisescu, C. 2022. Photorealistic Monocular 3D Reconstruction of Humans Wearing Clothing. In *CVPR*.

Anguelov, D.; Srinivasan, P.; Koller, D.; Thrun, S.; Rodgers, J.; and Davis, J. 2005. Scape: shape completion and animation of people. *ACM SIGGRAPH*.

Besl, P. J.; and McKay, N. D. 1992. A Method for Registration of 3-D Shapes. *TPAMI*.

Bogo, F.; Kanazawa, A.; Lassner, C.; Gehler, P.; Romero, J.; and Black, M. J. 2016. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *ECCV*.

Boykov, Y.; and Kolmogorov, V. 2004. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *TPAMI*.

Cao, Y.; Chen, G.; Han, K.; Yang, W.; and Wong, K.-Y. K. 2022. JIFF: Jointly-aligned Implicit Face Function for High Quality Single View Clothed Human Reconstruction. In *CVPR*.

Cheng, B.; Misra, I.; Schwing, A. G.; Kirillov, A.; and Girdhar, R. 2022. Masked-attention Mask Transformer for Universal Image Segmentation. *CVPR*.

Chibane, J.; Alldieck, T.; and Pons-Moll, G. 2020. Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. *CVPR*.

Fan, H.; Su, H.; and Guibas, L. J. 2017. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *CVPR*.

Farid, H.; and Simoncelli, E. P. 2004. Differentiation of discrete multidimensional signals. *IEEE TIP*.

Feng, Q.; Liu, Y.; Lai, Y.-K.; Yang, J.; and Li, K. 2022. FOF: Learning Fourier Occupancy Field for Monocular Real-time Human Reconstruction. *arXiv*.

Gabeur, V.; Franco, J.-S.; Martin, X.; Schmid, C.; and Rogez, G. 2019. Moulding humans: Non-parametric 3d human shape estimation from single images. In *ICCV*.

Guan, P.; Weiss, A.; Balan, A. O.; and Black, M. J. 2009. Estimating human shape and pose from a single image. In *ICCV*.

Guler, R. A.; and Kokkinos, I. 2019. Holopose: Holistic 3d human reconstruction in-the-wild. In *CVPR*.

Häne, C.; Tulsiani, S.; and Malik, J. 2017. Hierarchical surface prediction for 3d object reconstruction. In *3DV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

He, T.; Collomosse, J. P.; Jin, H.; and Soatto, S. 2020. Geo-PIFu: Geometry and Pixel Aligned Implicit Functions for Single-view Human Reconstruction. In *NeurIPS*.

He, T.; Xu, Y.; Saito, S.; Soatto, S.; and Tung, T. 2021. ARCH++: Animation-Ready Clothed Human Reconstruction Revisited. In *ICCV*.

Hong, Y.; Zhang, J.; Jiang, B.; Guo, Y.; Liu, L.; and Bao, H. 2021. Stereopifu: Depth aware clothed human digitization via stereo vision. In *CVPR*.

Hu, Y.-T.; Wang, J.; Yeh, R. A.; and Schwing, A. G. 2021. SAIL-VOS 3D: A Synthetic Dataset and Baselines for Object Detection and 3D Mesh Reconstruction from Video Data. In *CVPR*.

Huang, Z.; Xu, Y.; Lassner, C.; Li, H.; and Tung, T. 2020. ARCH: Animatable Reconstruction of Clothed Humans. In *CVPR*.

Kanazawa, A.; Black, M. J.; Jacobs, D. W.; and Malik, J. 2018. End-to-end recovery of human shape and pose. In *CVPR*.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. *ArXiv*.

Kolotouros, N.; Pavlakos, G.; and Daniilidis, K. 2019. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*.

Lassner, C.; Romero, J.; Kiefel, M.; Bogo, F.; Black, M. J.; and Gehler, P. V. 2017. Unite the people: Closing the loop between 3d and 2d human representations. In *CVPR*.

Lin, K.; Wang, L.; and Liu, Z. 2021. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*.

Lin, T.-Y.; Dollár, P.; Girshick, R. B.; He, K.; Hariharan, B.; and Belongie, S. J. 2017. Feature Pyramid Networks for Object Detection. In *CVPR*.

Lombardi, S.; Simon, T.; Saragih, J.; Schwartz, G.; Lehrmann, A.; and Sheikh, Y. 2019. Neural volumes: Learning dynamic renderable volumes from images. *arXiv*.

Lorensen, W. E.; and Cline, H. E. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH*.

Maturana, D.; and Scherer, S. 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*.

Meagher, D. 1982. Geometric modeling using octree encoding. *Computer graphics and image processing*.

Mescheder, L. M.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *CVPR*.

Milletari, F.; Navab, N.; and Ahmadi, S.-A. 2016. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In *3DV*.

Osman, A. A.; Bolkart, T.; and Black, M. J. 2020. Star: Sparse trained articulated human body regressor. In *ECCV*.

Osokin, D. 2018. Real-time 2D Multi-Person Pose Estimation on CPU: Lightweight OpenPose. In *arXiv:1811.12004*.

Pavlakos, G.; Choutas, V.; Ghorbani, N.; Bolkart, T.; Osman, A. A.; Tzionas, D.; and Black, M. J. 2019. Expressive body capture: 3d hands, face, and body from a single image. In *CVPR*.

Peng, S.; Zhang, Y.; Xu, Y.; Wang, Q.; Shuai, Q.; Bao, H.; and Zhou, X. 2021. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *CVPR*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*.

Ren, Z.; Agarwala, A.; Russell, B.; Schwing, A. G.; and Wang, O. 2022. Neural Volumetric Object Selection. In *CVPR*.

Ren, Z.; Zhao, X.; and Schwing, A. G. 2021. Class-agnostic Reconstruction of Dynamic Objects from Videos. In *NeurIPS*.

Saito, S.; Huang, Z.; Natsume, R.; Morishima, S.; Kanazawa, A.; and Li, H. 2019. PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. In *ICCV*.

Saito, S.; Simon, T.; Saragih, J. M.; and Joo, H. 2020. PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. In *CVPR*.

Sclaroff, S.; and Pentland, A. 1991. Generalized implicit functions for computer graphics. *ACM Siggraph*.

Shade, J.; Gortler, S.; He, L.-w.; and Szeliski, R. 1998. Layered depth images. In *Computer graphics and interactive techniques*.

Srinivasan, P. P.; Tucker, R.; Barron, J. T.; Ramamoorthi, R.; Ng, R.; and Snavely, N. 2019. Pushing the boundaries of view extrapolation with multiplane images. In *CVPR*.

Tong, J.; Zhou, J.; Liu, L.; Pan, Z.; and Yan, H. 2012. Scanning 3d full human bodies using kinects. *IEEE TVCG*.

Tucker, R.; and Snavely, N. 2020. Single-view view synthesis with multiplane images. In *CVPR*.

Varol, G.; Ceylan, D.; Russell, B.; Yang, J.; Yumer, E.; Laptev, I.; and Schmid, C. 2018. Bodynet: Volumetric inference of 3d human body shapes. In *ECCV*.

Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NeurIPS*.

Wang, L.; Zhao, X.; Yu, T.; Wang, S.; and Liu, Y. 2020. Normalgan: Learning detailed 3d human from a single rgb-d image. In *ECCV*.

Wang, N.; Zhang, Y.; Li, Z.; Fu, Y.; Liu, W.; and Jiang, Y.-G. 2018. Pixel2Mesh: Generating 3d mesh models from single rgb images. In *ECCV*.

Wu, M.; Wang, Y.; Hu, Q.; and Yu, J. 2020. Multi-view neural human rendering. In *CVPR*.

Wu, Y.; Chen, Z.; Liu, S.; Ren, Z.; and Wang, S. 2022. CASA: Category-agnostic Skeletal Animal Reconstruction. In *NeurIPS*.

Wu, Y.; and He, K. 2018. Group Normalization. In *ECCV*.

Xiang, D.; Joo, H.; and Sheikh, Y. 2019. Monocular total capture: Posing face, body, and hands in the wild. In *CVPR*.

Xu, Y.; Zhu, S.-C.; and Tung, T. 2019. Denserac: Joint 3d pose and shape estimation by dense render-and-compare. In *ICCV*.

Yang, J.; Franco, J.-S.; Hétroy-Wheeler, F.; and Wuhrer, S. 2016. Estimation of human body shape in motion with wide clothing. In *ECCV*.

Yu, A.; Li, R.; Tancik, M.; Li, H.; Ng, R.; and Kanazawa, A. 2021. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*.

Yu, T.; Guo, K.; Xu, F.; Dong, Y.; Su, Z.; Zhao, J.; Li, J.; Dai, Q.; and Liu, Y. 2017. Bodyfusion: Real-time capture of human motion and surface geometry using a single depth camera. In *ICCV*.

Zhang, C.; Pujades, S.; Black, M. J.; and Pons-Moll, G. 2017. Detailed, accurate, human shape estimation from clothed 3D scan sequences. In *CVPR*.

Zhao, X.; Ma, F.; Güera, D.; Ren, Z.; Schwing, A. G.; and Colburn, A. 2022. Generative Multiplane Images: Making a 2D GAN 3D-Aware. In *ECCV*.

Zheng, Z.; Yu, T.; Wei, Y.; Dai, Q.; and Liu, Y. 2019. DeepHuman: 3D Human Reconstruction From a Single Image. In *ICCV*.

Zhou, T.; Tucker, R.; Flynn, J.; Fyffe, G.; and Snavely, N. 2018. Stereo magnification: Learning view synthesis using multiplane images. *ACM SIGGRAPH*.

# Supplementary Material: Occupancy Planes for Single-view RGB-D Human Reconstruction

This supplementary material is structured as follows:

1. Sec. A: Implementation details
2. Sec. B: Additional quantitative results

## A Implementation Details

### A.1 Input to Image Feature Extractor

To extract the image features $\mathcal{F}_{\text{RGB}}^{h_O \times w_O} = f_{\text{RGB}}(f_{\text{FPN}}(\hat{I}_{\text{RGB}}))$ (see Eq. (6)), instead of feeding the raw RGB image $I_{\text{RGB}} \in \mathbb{R}^{H \times W \times 3}$ into the FPN backbone, we first concatenate the image $I_{\text{RGB}}$ with two simply-processed one-channel features which are concatenated to the image along the channel dimension. We therefore use $\hat{I}_{\text{RGB}} \in \mathbb{R}^{H \times W \times 5}$, which will be fed into the FPN, *i.e.*, $\mathcal{F}_{\text{RGB}}^{h_O \times w_O} = f_{\text{RGB}}(f_{\text{FPN}}(\hat{I}_{\text{RGB}}))$. The two one-channel features are: 1) for each pixel, we compute the distance to the visibility mask's boundary; 2) we detect edges with the help of a Farid filter (Farid and Simoncelli 2004).

### A.2 Visualization

Even though occupancy planes learn a high-quality inductive bias for single-view RGB-D human reconstruction, floating artifacts behind the visible surfaces are possible as inferring invisible parts is an ill-posed problem. For a smooth visualization, we utilize 1) the smoothing function in `PyMCubes` [2]; 2) the GraphCut algorithm (Boykov and Kolmogorov 2004) in `medpy` [3]. Importantly, note that we only apply this post-processing for visualization purposes and we *never* use this post-processing when reporting quantitative results.

## B More Quantitative Results

### B.1 Performance across Various Visibility Levels

Since OPlanes can deal with humans of various visibilities, we are interested in understanding how the proposed approach performs across different partial visibility levels. We present results with respect to different visibility levels in Tab. S1. To compute the visibility we use three steps: 1) we uniformly sample 100k points within the complete mesh of the human; 2) we project those 100k 3D points onto the 2D image and count the number of points which are in view; 3) the level of partial visibility is computed as the ratio of in-view points, *i.e.*, the number of in-view points divided by 100k. We also explicitly consider the fully visible humans in the 4th row of Tab. S1. Results for different visibility ranges are provided in the 1st to 3rd row of Tab. S1. As expected, the more visible the human, the better the model performs. Specifically, comparing full visibility to low visibility (4th *vs.* 1st row), we obtain higher IoU (0.707 *vs.* 0.668), smaller Chamfer distance (0.109 *vs.* 0.289), and more normal consistency (0.759 *vs.* 0.703). However, it is notable that the drop in performance is not very severe.

To verify this, we also report IF-Net and PIFuHD results for each visibility range in Tab. S1. Specifically, comparing the 4th *vs.* 1st row, we observe: 1) for IoU (↑ is better), IF-Net's performance drops from 0.644 to 0.365 and PIFuHD results drop from 0.533 to 0.131; 2) for Chamfer distance (↓ is better), IF-Net results deteriorate from 0.134 to 0.444 and PIFuHD results worsen from 0.214 to 0.702; 3) for Normal consistency (↑ is better), IF-Net results drop from 0.828 to 0.715 while PIFuHD results drop from 0.734 to 0.543. Summarizing the three observations, we find OPlanes to be more robust to partial visibility.

Table S1: **Performance across various visibility levels**. For each cell, we report in the format of OPlane / IF-Net / PIFuHD. Each OPlane result is averaged over three runs with different seeds and is reported in the format of `mean`±`std`. The column "Visibility Range" refers to the range of the visibility percentage of a mesh. The higher the more visible. #Data denotes the number of evaluation entries in the corresponding range (row of the table). We report the overall performance in the 5th row while the 1st to 4th rows provide more fine-grained results. Note, the 4th row presents results for fully-visible objects. As expected, when the visibility drops, the performance drops too.

| | Partial Visibility | Visibility Range | #Data | IoU↑ | Cham-$\mathcal{L}_1$ ↓ | Normal Consistency ↑ |
|---|---|---|---|---|---|---|
| 1 | Low | [0.069, 0.379) | 64 | 0.668±0.021 / 0.365 / 0.131 | 0.289±0.023 / 0.444 / 0.702 | 0.703±0.008 / 0.715 / 0.543 |
| 2 | Middle | [0.379, 0.690) | 552 | 0.618±0.013 / 0.376 / 0.172 | 0.291±0.012 / 0.461 / 0.654 | 0.710±0.006 / 0.731 / 0.559 |
| 3 | High | [0.690, 1.000) | 2167 | 0.699±0.013 / 0.602 / 0.429 | 0.149±0.008 / 0.204 / 0.322 | 0.753±0.005 / 0.805 / 0.672 |
| 4 | Full | [1.000, 1.000] | 1517 | 0.707±0.013 / 0.644 / 0.533 | 0.109±0.006 / 0.134 / 0.214 | 0.759±0.005 / 0.828 / 0.734 |
| 5 | - | [0.069, 1.000] | 4300 | 0.691±0.013 / 0.584 / 0.428 | 0.155±0.008 / 0.216 / 0.332 | 0.749±0.005 / 0.802 / 0.677 |

[2] https://github.com/pmneila/PyMCubes

[3] https://github.com/loli/medpy/