

Pose2Mesh: Graph Convolutional Network for 3D Human Pose and Mesh Recovery from a 2D Human Pose

Hongsuk Choi*, Gyeongsik Moon*, and Kyoung Mu Lee

ECE & ASRI, Seoul National University, Korea
 {redarknight,mks0601,kyoungmu}@snu.ac.kr

Abstract. Most of the recent deep learning-based 3D human pose and mesh estimation methods regress the pose and shape parameters of human mesh models, such as SMPL and MANO, from an input image. The first weakness of these methods is an appearance domain gap problem, due to different image appearance between train data from controlled environments, such as a laboratory, and test data from in-the-wild environments. The second weakness is that the estimation of the pose parameters is quite challenging owing to the representation issues of 3D rotations. To overcome the above weaknesses, we propose Pose2Mesh, a novel graph convolutional neural network (GraphCNN)-based system that estimates the 3D coordinates of *human mesh vertices* directly from the *2D human pose*. The 2D human pose as input provides essential human body articulation information, while having a relatively homogeneous geometric property between the two domains. Also, the proposed system avoids the representation issues, while fully exploiting the mesh topology using a GraphCNN in a coarse-to-fine manner. We show that our Pose2Mesh outperforms the previous 3D human pose and mesh estimation methods on various benchmark datasets. The codes are publicly available¹.

1 Introduction

3D human pose and mesh estimation aims to recover 3D human joint and mesh vertex locations simultaneously. It is a challenging task due to the depth and scale ambiguity, and the complex human body and hand articulation. There have been diverse approaches to address this problem, and recently, deep learning-based methods have shown noticeable performance improvement.

Most of the deep learning-based methods rely on human mesh models, such as SMPL [37] and MANO [54]. They can be generally categorized into a model-based approach and a model-free approach. The model-based approach trains a network to predict the model parameters and generates a human mesh by decoding them [5–7, 27, 31, 33, 47, 48, 52]. On the contrary, the model-free approach regresses the coordinates of a 3D human mesh directly [15, 32]. Both

* equal contribution

¹ https://github.com/hongsukchoi/Pose2Mesh_RELEASE

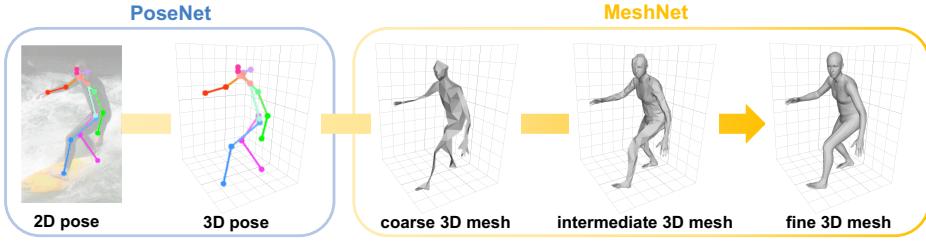


Fig. 1: The overall pipeline of Pose2Mesh.

approaches compute the 3D human pose by multiplying the output mesh with a joint regression matrix, which is defined in the human mesh models [37, 54].

Although the recent deep learning-based methods have shown significant improvement, they have two major drawbacks. First, when tested on in-the-wild data, the methods inherently suffer from the appearance domain gap between controlled and in-the-wild environment data. The data captured from the controlled environments [22, 26] is valuable train data in 3D human pose and estimation, because it has accurate 3D annotations. However, due to the significant difference of image appearance between the two domains, such as backgrounds and clothes, an image-based approach cannot fully benefit from the data. The second drawback is that the pose parameters of the human mesh models might not be an appropriate regression target, as addressed in Kolotouros et al. [32]. The SMPL pose parameters, for example, represent 3D rotations in an axis-angle, which can suffer from the non-unique problem (*i.e.*, periodicity). While many works [27, 33, 47] tried to avoid the periodicity by using a rotation matrix as the prediction target, it still has a non-minimal representation issue.

To resolve the above issues, we propose Pose2Mesh, a graph convolutional system that recovers 3D human pose and mesh from the 2D human pose, in a model-free fashion. It has two advantages over existing methods. First, the proposed system benefits from a relatively homogeneous geometric property of the input 2D poses from controlled and in-the-wild environments. They not only alleviates the appearance domain gap issue, but also provide essential geometric information on the human articulation. Also, the 2D poses can be estimated accurately from in-the-wild images, since many well-performing methods [11, 44, 58, 66] are trained on large-scale in-the-wild 2D human pose datasets [2, 35]. The second advantage is that Pose2Mesh avoids the representation issues of the pose parameters, while exploiting the human mesh topology (*i.e.*, face and edge information). It directly regresses the 3D coordinates of mesh vertices using a graph convolutional neural network (GraphCNN) with graphs constructed from the mesh topology.

We designed Pose2Mesh in a cascaded architecture, which consists of PoseNet and MeshNet. PoseNet lifts the 2D human pose to the 3D human pose. MeshNet takes both 2D and 3D human poses to estimate the 3D human mesh in a coarse-to-fine manner. During the forward propagation, the mesh features are initially processed in a coarse resolution and gradually upsampled to a fine resolution. Figure 1 depicts the overall pipeline of the system.

The experimental results show that the proposed Pose2Mesh outperforms the previous state-of-the-art 3D human pose and mesh estimation methods [27,31,32] on various publicly available 3D human body and hand datasets [22, 39, 70]. Particularly, our Pose2Mesh provides the state-of-the-art result on in-the-wild dataset [39], even when it is trained only on the controlled setting dataset [22].

We summarize our contributions as follows.

- We propose a novel system, Pose2Mesh, that recovers 3D human pose and mesh from the 2D human pose. The input 2D human pose lets Pose2Mesh robust to the appearance domain gap between controlled and in-the-wild environment data.
- Our Pose2Mesh directly regresses 3D coordinates of a human mesh using GraphCNN. It avoids representation issues of the model parameters and leverages the pre-defined mesh topology.
- We show that Pose2Mesh outperforms previous 3D human pose and mesh estimation methods on various publicly available datasets.

2 Related works

3D human body pose estimation. Current 3D human body pose estimation methods can be categorized into two approaches according to the input type: an image-based approach and a 2D pose-based approach. The image-based approach takes an RGB image as an input for 3D body pose estimation. Sun et al. [59] proposed to use compositional loss, which exploits the joint connection structure. Sun et al. [60] employed soft-argmax operation to regress the 3D coordinates of body joints in a differentiable way. Sharma et al. [56] incorporated a generative model and depth ordering of joints to predict the most reliable 3D pose that corresponds to the estimated 2D pose.

The 2D pose-based approach lifts the 2D human pose to the 3D space. Martinez et al. [40] introduced a simple network that consists of consecutive fully-connected layers, which lifts the 2D human pose to the 3D space. Zhao et al. [68] developed a semantic GraphCNN to use spatial relationships between joint coordinates. Our work follows the 2D pose-based approach, to make the Pose2Mesh more robust to the domain difference between the controlled environment of the training set and in-the-wild environment of the testing set.

3D human body and hand pose and mesh estimation. A model-based approach trains a neural network to estimate the human mesh model parameters [37, 54]. It has been widely used for the 3D human mesh estimation, since it does not necessarily require 3D annotation for mesh supervision. Pavlakos et al. [52] proposed a system that could be only supervised by 2D joint coordinates and silhouette. Omran et al. [47] trained a network with 2D joint coordinates, which takes human part segmentation as input. Kanazawa et al. [27] utilized adversarial loss to regress plausible SMPL parameters. Baek et al. [5] trained a CNN to estimate parameters of the MANO model using neural renderer [29]. Kolotouros et al. [31] introduced a self-improving system that consists of SMPL parameter regressor and iterative fitting framework [6].

Recently, the advance of fitting frameworks [6, 50] has motivated a model-free approach, which estimates human mesh coordinates directly. It enabled researchers to obtain 3D mesh annotation, which is essential for the model-free methods, from in-the-wild data. Kolotouros et al. [32] proposed a GraphCNN, which learns the deformation of the template body mesh to the target body mesh. Ge et al. [15] adopted a GraphCNN to estimate vertices of hand mesh. Moon et al. [45] proposed a new heatmap representation, called lixel, to recover 3D human meshes.

Our Pose2Mesh differs from the above methods, which are image-based, in that it uses the 2D human pose as an input. The proposed system can benefit from the data with 3D annotations, which are captured from controlled environments [22, 26], without the appearance domain gap issue.

GraphCNN for mesh processing. Recently, many methods consider a mesh as a graph structure and process it using the GraphCNN, since it can fully exploit mesh topology compared with simple stacked fully-connected layers. Wang et al. [65] adopted a GraphCNN to learn a deformation from an initial ellipsoid mesh to the target object mesh in a coarse-to-fine manner. Verma et al. [63] proposed a novel graph convolution operator for the shape correspondence problem. Ranjan et al. [53] also proposed a GraphCNN-based VAE, which learns a latent space of the human face meshes in a hierarchical manner.

3 PoseNet

3.1 Synthesizing errors on the input 2D pose

PoseNet estimates the root joint-relative 3D pose $\mathbf{P}^{3D} \in \mathbb{R}^{J \times 3}$ from the 2D pose, where J denotes the number of human joints. We define the root joint of the human body and hand as pelvis and wrist, respectively. The estimated 2D pose often contains errors [55], especially under severe occlusions or challenging poses. To make PoseNet robust to the errors, we synthesize 2D input poses by adding realistic errors on the ground truth 2D pose, following [10, 44], during the training stage. We represent the estimated 2D pose or the synthesized 2D pose as $\mathbf{P}^{2D} \in \mathbb{R}^{J \times 2}$.

3.2 2D input pose normalization

We apply standard normalization to \mathbf{P}^{2D} , following [10, 64]. For this, we subtract the mean from \mathbf{P}^{2D} and divide it by the standard deviation, which becomes $\bar{\mathbf{P}}^{2D}$. The mean and the standard deviation of \mathbf{P}^{2D} represent the 2D location and scale of the subject, respectively. This normalization is necessary because \mathbf{P}^{3D} is independent of scale and location of the 2D input pose \mathbf{P}^{2D} .

3.3 Network architecture

The architecture of the PoseNet is based on that of [10, 40]. The normalized 2D input pose $\bar{\mathbf{P}}^{2D}$ is converted to a 4096-dimensional feature vector through a fully-connected layer. Then, it is fed to the two residual blocks [21]. Finally, the output feature vector of the residual blocks is converted to $(3J)$ -dimensional vector, which represents \mathbf{P}^{3D} , by a full-connected layer.

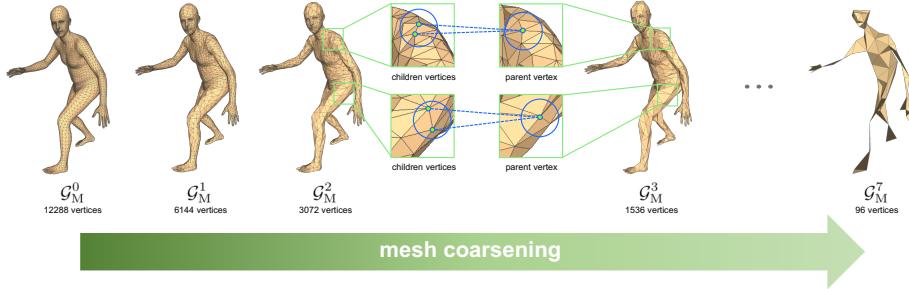


Fig. 2: The coarsening initially generates multiple coarse graphs from \mathcal{G}_M , and adds fake nodes without edges to each graph, following [13]. The numbers of vertices range from 96 to 12288 for body meshes and from 68 to 1088 for hand meshes.

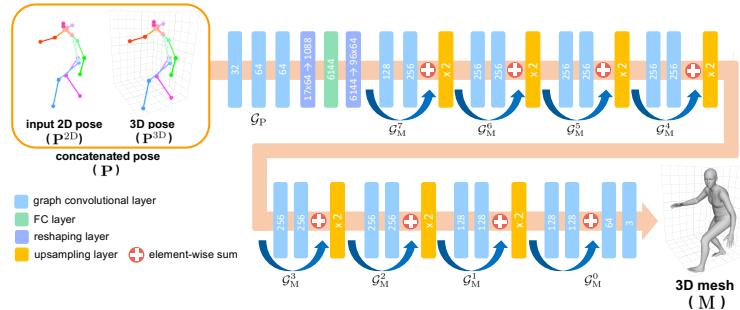


Fig. 3: The network architecture of MeshNet.

3.4 Loss function

We train the PoseNet by minimizing $L1$ distance between the predicted 3D pose \mathbf{P}^{3D} and groundtruth. The loss function L_{pose} is defined as follows:

$$L_{pose} = \|\mathbf{P}^{3D} - \mathbf{P}^{3D*}\|_1, \quad (1)$$

where the asterisk indicates the groundtruth.

4 MeshNet

4.1 Graph convolution on pose

MeshNet concatenates $\bar{\mathbf{P}}^{2D}$ and \mathbf{P}^{3D} into $\mathbf{P} \in \mathbb{R}^{J \times 5}$. Then, it estimates the root joint-relative 3D mesh $\mathbf{M} \in \mathbb{R}^{V \times 3}$ from \mathbf{P} , where V denotes the number of human mesh vertices. To this end, MeshNet uses the spectral graph convolution [8, 57], which can be defined as the multiplication of a signal $x \in \mathbb{R}^N$ with a filter $g_\theta = diag(\theta)$ in Fourier domain as follows:

$$g_\theta * x = U g_\theta U^T x, \quad (2)$$

where graph Fourier basis U is the matrix of the eigenvectors of the normalized graph Laplacian L [12], and $U^T x$ denotes the graph Fourier transform of x . Specifically, to reduce the computational complexity, we design MeshNet to be based on Chebysev spectral graph convolution [13].

Graph construction. We construct a graph of \mathbf{P} , $\mathcal{G}_P = (\mathcal{V}_P, A_P)$, where $\mathcal{V}_P = \mathbf{P} = \{\mathbf{p}_i\}_{i=1}^J$ is a set of J human joints, and $A_P \in \{0, 1\}^{J \times J}$ is an adjacency matrix. A_P defines the edge connections between the joints based on the human skeleton and symmetrical relationships [9], where $(A_P)_{ij} = 1$ if joints i and j are the same or connected, and $(A_P)_{ij} = 0$ otherwise. The normalized Laplacian is computed as $L_P = I_J - D_P^{-1/2} A_P D_P^{-1/2}$, where I_J is the identity matrix, and D_P is the diagonal matrix which represents the degree of each joint in \mathcal{V}_P as $(D_P)_{ij} = \sum_j (A_P)_{ij}$. The scaled Laplacian is computed as $\tilde{L}_P = 2L_P/\lambda_{\max} - I_J$.

Spectral convolution on graph. Then, MeshNet performs the spectral graph convolution on \mathcal{G}_P , which is defined as follows:

$$F_{\text{out}} = \sum_{k=0}^{K-1} T_k(\tilde{L}_P) F_{\text{in}} \Theta_k, \quad (3)$$

where $F_{\text{in}} \in \mathbb{R}^{J \times f_{\text{in}}}$ and $F_{\text{out}} \in \mathbb{R}^{J \times f_{\text{out}}}$ are the input and output feature maps respectively, $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ is the Chebysev polynomial [18] of order k , and $\Theta_k \in \mathbb{R}^{f_{\text{in}} \times f_{\text{out}}}$ is the k th Chebysev coefficient matrix, whose elements are the trainable parameters of the graph convolutional layer. f_{in} and f_{out} are the input and output feature dimensions respectively. The initial input feature map F_{in} is \mathbf{P} in practice, where $f_{\text{in}} = 5$. This graph convolution is K -localized, which means at most K -hop neighbor nodes from each node are affected [13, 30], since it is a K -order polynomial in the Laplacian. Our MeshNet sets $K = 3$ for all graph convolutional layers following [15].

4.2 Coarse-to-fine mesh upsampling

We gradually upsample \mathcal{G}_P to the graph of \mathbf{M} , $\mathcal{G}_M = (\mathcal{V}_M, A_M)$, where $\mathcal{V}_M = \mathbf{M} = \{\mathbf{m}_i\}_{i=1}^V$ is a set of V human mesh vertices, and $A_M \in \{0, 1\}^{V \times V}$ is an adjacency matrix defining edges of the human mesh. To this end, we apply the graph coarsening [14] technique to \mathcal{G}_M , which creates various resolutions of graphs, $\{\mathcal{G}_M^c = (\mathcal{V}_M^c, A_M^c)\}_{c=0}^C$, where C denotes the number of coarsening steps, following Defferrard et al. [13]. Figure 2 shows the coarsening process and a balanced binary tree structure of mesh graphs, where the i th vertex in \mathcal{G}_M^{c+1} is a parent node of the $2i-1$ th and $2i$ th vertices in \mathcal{G}_M^c , and $2|\mathcal{V}_M^{c+1}| = |\mathcal{V}_M^c|$. i starts from 1. The final output of MeshNet is \mathcal{V}_M , which is converted from \mathcal{V}_M^0 by a pre-defined indices mapping. During the forward propagation, MeshNet first upsamples the \mathcal{G}_P to the coarsest mesh graph \mathcal{G}_M^C by reshaping and a fully-connected layer. Then, it performs the spectral graph convolution on each resolution of mesh graphs as follows:

$$F_{\text{out}} = \sum_{k=0}^{K-1} T_k(\tilde{L}_M^c) F_{\text{in}} \Theta_k, \quad (4)$$

where \tilde{L}_M^c denotes the scaled Laplacian of \mathcal{G}_M^c , and the other notations are defined in the same manner as Equation 3. Following [15], MeshNet performs mesh upsampling by copying features of each parent vertex in \mathcal{G}_M^{c+1} to the corresponding children vertices in \mathcal{G}_M^c . The upsampling process is defined as follows:

$$F_c = \psi(F_{c+1}^T)^T, \quad (5)$$

where $F_c \in \mathbb{R}^{\mathcal{V}_M^c \times f_c}$ is the first feature map of \mathcal{G}_M^c , $F_{c+1} \in \mathbb{R}^{\mathcal{V}_M^{c+1} \times f_{c+1}}$ is the last feature map of \mathcal{G}_M^{c+1} , $\psi: \mathbb{R}^{f_{c+1} \times \mathcal{V}_M^{c+1}} \rightarrow \mathbb{R}^{f_c \times \mathcal{V}_M^c}$ denotes a nearest-neighbor upsampling function, and f_c and f_{c+1} are the feature dimensions of vertices in F_c and F_{c+1} respectively. The nearest upsampling function copies the feature of the i th vertex in \mathcal{G}_M^{c+1} to the $2i - 1$ th and $2i$ th vertices in \mathcal{G}_M^c . To facilitate the learning process, we additionally incorporate a residual connection between each resolution. Figure 3 shows the overall architecture of MeshNet.

4.3 Loss functions

To train our MeshNet, we use four loss functions.

Vertex coordinate loss. We minimize $L1$ distance between the predicted 3D mesh coordinates \mathbf{M} and groundtruth, which is defined as follows:

$$L_{\text{vertex}} = \|\mathbf{M} - \mathbf{M}^*\|_1, \quad (6)$$

where the asterisk indicates the groundtruth.

Joint coordinate loss. We use a $L1$ loss function between the groundtruth root-relative 3d pose and the 3D pose regressed from \mathbf{M} , to train our MeshNet to estimate mesh vertices aligned with joint locations. The 3D pose is calculated as $\mathcal{J}\mathbf{M}$, where $\mathcal{J} \in \mathbb{R}^{J \times V}$ is a joint regression matrix defined in SMPL or MANO model. The loss function is defined as follows:

$$L_{\text{joint}} = \|\mathcal{J}\mathbf{M} - \mathbf{P}^{3D*}\|_1, \quad (7)$$

where the asterisk indicates the groundtruth.

Surface normal loss. We supervise normal vectors of an output mesh surface to be consistent with groundtruth. This consistency loss improves surface smoothness and local details [65]. The loss function L_{normal} is defined as follows:

$$L_{\text{normal}} = \sum_f \sum_{\{i,j\} \subset f} \left| \left\langle \frac{\mathbf{m}_i - \mathbf{m}_j}{\|\mathbf{m}_i - \mathbf{m}_j\|_2}, n_f^* \right\rangle \right|, \quad (8)$$

where f and n_f^* denote a triangle face in the human mesh and a groundtruth unit normal vector of f , respectively. \mathbf{m}_i and \mathbf{m}_j denote the i th and j th vertices in f .

Surface edge loss. We define edge length consistency loss between predicted and groundtruth edges, following [65]. The edge loss is effective in recovering smoothness of hands, feet, and a mouth, which have dense vertices. The loss function L_{edge} is defined as follows:

$$L_{\text{edge}} = \sum_f \sum_{\{i,j\} \subset f} |\|\mathbf{m}_i - \mathbf{m}_j\|_2 - \|\mathbf{m}_i^* - \mathbf{m}_j^*\|_2|, \quad (9)$$

where f and the asterisk denote a triangle face in the human mesh and the groundtruth, respectively. \mathbf{m}_i and \mathbf{m}_j denote i th and j th vertex in f .

We define the total loss of our MeshNet, L_{mesh} , as a weighted sum of all four loss functions:

$$L_{\text{mesh}} = \lambda_v L_{\text{vertex}} + \lambda_j L_{\text{joint}} + \lambda_n L_{\text{normal}} + \lambda_e L_{\text{edge}}, \quad (10)$$

where $\lambda_v = 1$, $\lambda_j = 1$, $\lambda_n = 0.1$, and $\lambda_e = 20$.

5 Implementation Details

PyTorch [49] is used for implementation. We first pre-train our PoseNet, and then train the whole network, Pose2Mesh, in an end-to-end manner. Empirically, our two-step training strategy gives better performance than the one-step training. The weights are updated by the Rmsprop optimization [61] with a mini-batch size of 64. We pre-train PoseNet 60 epochs with a learning rate 10^{-3} . The learning rate is reduced by a factor of 10 after the 30th epoch. After integrating the pre-trained PoseNet to Pose2Mesh, we train the whole network 15 epochs with a learning rate 10^{-3} . The learning rate is reduced by a factor of 10 after the 12th epoch. In addition, we set λ_e to 0 until 7 epoch on the second training stage, since it tends to cause local optima at the early training phase. We used four NVIDIA RTX 2080 Ti GPUs for Pose2Mesh training, which took at least a half day and at most two and a half days, depending on the training datasets. In inference time, we use 2D pose outputs from Sun et al. [58] and Xiao et al. [66]. They run at 5 fps and 67 fps respectively, and our Pose2Mesh runs at 37 fps. Thus, the proposed system can process from 4 fps to 22 fps in practice, which shows the applicability to real-time applications.

6 Experiment

6.1 Datasets and evaluation metrics

Human3.6M. Human3.6M [22] is a large-scale indoor 3D body pose benchmark, which consists of 3.6M video frames. The groundtruth 3D poses are obtained using a motion capture system, but there are no groundtruth 3D meshes. As a result, for 3D mesh supervision, most of the previous 3D pose and mesh estimation works [27, 31, 32] used pseudo-groundtruth obtained from Mosh [36]. However, due to the license issue, the pseudo-groundtruth from Mosh is not currently publicly accessible. Thus, we generate new pseudo-groundtruth 3D meshes by fitting SMPL parameters to the 3D groundtruth poses using SMPLify-X [50]. For the fair comparison, we trained and tested previous state-of-the-art methods on the obtained groundtruth using their officially released code. Following [27, 51], all methods are trained on 5 subjects (S1, S5, S6, S7, S8) and tested on 2 subjects (S9, S11).

We report our performance for the 3D pose using two evaluation metrics. One is mean per joint position error (MPJPE) [22], which measures the Euclidean distance in millimeters between the estimated and groundtruth joint coordinates,

Table 1: The performance comparison between four combinations of regression target and network design tested on Human3.6M. ‘no. param.’ denotes the number of parameters of a network, which estimates SMPL parameters or vertex coordinates from the output of PoseNet.

target\network	FC			GraphCNN		
	MPJPE	PA-MPJPE	no. param.	MPJPE	PA-MPJPE	no. param.
SMPL param.	72.8	55.5	17.3M	79.1	59.1	13.5M
vertex coord.	119.6	95.1	37.5M	64.9	48.0	8.8M

Table 2: The performance comparison on Human3.6M between two upsampling schemes. GPU mem. and fps denote the required memory during training and fps in inference time respectively.

method	GPU mem.	fps	MPJPE
direct	10G	24	65.3
coarse-to-fine	6G	37	64.9

Table 3: The MPJPE comparison between four architectures tested on 3DPW.

architecture	MPJPE
2D→mesh	101.1
2D→3D→mesh	103.2
2D→3D+2D→mesh	100.5

after aligning the root joint. The other one is PA-MPJPE, which calculates MPJPE after further alignment (*i.e.*, Procrustes analysis (PA) [17]). \mathcal{JM} is used for the estimated joint coordinates. We only evaluate 14 joints out of 17 estimated joints following [27, 31, 32, 52].

3DPW. 3DPW [39] is captured from in-the-wild and contains 3D body pose and mesh annotations. It consists of 51K video frames, and IMU sensors are leveraged to acquire the groundtruth 3D pose and mesh. We only use the test set of 3DPW for evaluation following [31]. MPJPE and mean per vertex position error (MPVPE) are used for evaluation. 14 joints from \mathcal{JM} , whose joint set follows that of Human3.6M, are evaluated for MPJPE as above. MPVPE measures the Euclidean distance in millimeters between the estimated and groundtruth vertex coordinates, after aligning the root joint.

COCO. COCO [35] is an in-the-wild dataset with various 2D annotations such as detection and human joints. To exploit this dataset on 3D mesh learning, Kolotouros et al. [31] fitted SMPL parameters to 2D joints using SMPLify [6]. Following them, we use the processed data for training.

MuCo-3DHP. MuCo-3DHP [42] is synthesized from the existing MPI-INF-3DHP 3D single-person pose estimation dataset [41]. It consists of 200K frames, and half of them have augmented backgrounds. For the background augmentation, we use images of COCO that do not include humans to follow Moon et al. [43]. Following them, we use this dataset only for the training.

FreiHAND. FreiHAND [70] is a large-scale 3D hand pose and mesh dataset. It consists of a total of 134K frames for training and testing. Following Zimmermann et al. [70], we report PA-MPVPE, F-scores, and additionally PA-MPJPE of Pose2Mesh. \mathcal{JM} is evaluated for the joint errors.

Table 4: The upper bounds of the two different graph convolutional networks that take a 2D pose and a 3D pose. Tested on Human3.6M.

test input	architecture	MPJPE	PA-MPJPE
2D pose GT	2D→mesh	55.5	38.4
3D pose from [43]	3D→mesh	56.3	43.2
3D pose GT	3D→mesh	29.0	23.0

6.2 Ablation study

To analyze each component of the proposed system, we trained different networks on Human3.6M, and evaluated on Human3.6M and 3DPW. The test 2D input poses used in Human3.6M and 3DPW evaluation are outputs from Integral Regression [60] and HRNet [58] respectively, which are obtained using groundtruth bounding boxes.

Regression target and network design. To demonstrate the effectiveness of regressing the 3D mesh vertex coordinates using GraphCNN, we compare MPJPE and PA-MPJPE of four different combinations of the regression target and the network design in Table 1. First, *vertex-GraphCNN*, our Pose2Mesh, substantially improves the joint errors compared to *vertex-FC*, which regresses vertex coordinates with a network of fully-connected layers. This proves the importance of exploiting the human mesh topology with GraphCNN, when estimating the 3D vertex coordinates. Second, *vertex-GraphCNN* provides better performance than both networks estimating SMPL parameters, while maintaining the considerably smaller number of network parameters. Taken together, the effectiveness of our mesh coordinate regression scheme using GraphCNN is clearly justified.

In this comparison, the same PoseNet and cascaded architecture are employed for all networks. On top of the PoseNet, *vertex-FC* and *param-FC* used a series of fully-connected layers, whereas *param-GraphCNN* added fully-connected layers on top of Pose2Mesh. For the fair comparison, when training *param-FC* and *param-GraphCNN*, we also supervised the reconstructed mesh from the predicted SMPL parameters with L_{vertex} and L_{joint} . The networks estimating SMPL parameters incorporated Zhou et al.’s method [69] for continuous rotations.

Coarse-to-fine mesh upsampling. We compare a coarse-to-fine mesh upsampling scheme and a direct mesh upsampling scheme. The direct upsampling method performs graph convolution on the lowest resolution mesh until the middle layer of MeshNet, and then directly upsamples it to the highest one (*e.g.*, 96 to 12288 for the human body mesh). While it has the same number of graph convolution layers and almost the same number of parameters, our coarse-to-fine model consumes half as much GPU memory and runs 1.5 times faster than the direct upsampling method. It is because graph convolution on the highest resolution takes much more time and memory than graph convolution on lower resolutions. In addition, the coarse-to-fine upsampling method provides a slightly

Table 5: The accuracy comparison between state-of-the-art methods and Pose2Mesh on Human3.6M. The dataset names on top are training sets.

method	Human3.6M		Human3.6M + COCO	
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
HMR [27]	184.7	88.4	153.2	85.5
GraphCMR [32]	148.0	104.6	78.3	59.5
SPIN [31]	85.6	55.6	72.9	51.9
Pose2Mesh (Ours)	64.9	48.0	67.9	49.9

Table 6: The accuracy comparison between state-of-the-art methods and Pose2Mesh on 3DPW. The dataset names on top are training sets.

method	Human3.6M			Human3.6M + COCO		
	MPJPE	PA-MPJPE	MPVPE	MPJPE	PA-MPJPE	MPVPE
HMR [27]	377.3	165.7	481.0	300.4	137.2	406.8
GraphCMR [32]	332.5	177.4	380.8	126.5	80.1	144.8
SPIN [31]	313.8	156.0	344.3	113.1	71.7	122.8
Pose2Mesh (Simple [66])	101.8	64.2	119.1	92.3	61.0	110.5
Pose2Mesh (HR [58])	100.5	63.0	117.5	91.4	60.1	109.3

lower joint error, as shown in Table 2. These results confirm the effectiveness of our coarse-to-fine upsampling strategy.

Cascaded architecture analysis. We analyze the cascaded architecture of Pose2Mesh to demonstrate its validity in Table 3. To be specific, we construct (a) a GraphCNN that directly takes a 2D pose, (b) a cascaded network that predicts mesh coordinates from a 3D pose from pretrained PoseNet, and (c) our Pose2Mesh. All methods are both trained by synthesized 2D poses. First, (a) outperforms (b), which implies a 3D pose output from PoseNet may lack geometry information in the 2D input pose. If we concatenate the 3D pose output with the 2D input pose as (c), it provides the lowest errors. This explains that depth information in 3D poses could positively affect 3D mesh estimation.

To further verify the superiority of the cascaded architecture, we explore the upper bounds of (a) and (d) a GraphCNN that takes a 3D pose in Table 13. To this end, we fed the groundtruth 2D pose and 3D pose to (a) and (d) as test inputs, respectively. Apparently, since the input 3D pose contains additional depth information, the upper bound of (d) is considerably higher than that of (a). We also fed state-of-the-art 3D pose outputs from [43] to (d), to validate the practical potential for performance improvement. Surprisingly, the performance is comparable to the upper bound of (a). Thus, our Pose2Mesh will substantially outperform (a) a graph convolution network that directly takes a 2D pose, if we can improve the performance of PoseNet. In summary, the above results prove the validity of our cascaded architecture of Pose2Mesh.

Table 7: The accuracy comparison between state-of-the-art methods and Pose2Mesh on FreiHAND.

method	PA-MPVPE	PA-MPJPE	F@5 mm	F@15 mm
Hasson et al. [19]	13.2	-	0.436	0.908
Boukhayma et al. [7]	13.0	-	0.435	0.898
FreiHAND [70]	10.7	-	0.529	0.935
Pose2Mesh (Ours)	7.6	7.4	0.683	0.973

6.3 Comparison with state-of-the-art methods

Human3.6M. We compare our Pose2Mesh with the previous state-of-the-art 3D body pose and mesh estimation methods on Human3.6M in Table 5. First, when we train all methods only on Human3.6M, our Pose2Mesh significantly outperforms other methods. However, when we train the methods additionally on COCO, the performance of the previous baselines increases, but that of Pose2Mesh slightly decreases. The performance gain of other methods is a well-known phenomenon [60] among image-based methods, which tend to generalize better when trained with diverse images from in-the-wild. Whereas, our Pose2Mesh does not benefit from more images in the same manner, since it only takes the 2D pose. We analyze the reason for the performance drop is that the test set and train set of Human3.6M have similar poses, which are from the same action categories. Thus, overfitting the network to the poses of Human3.6M can lead to better accuracy. Nevertheless, in both cases, our Pose2Mesh outperforms the previous methods in both MPJPE and PA-MPJPE. The test 2D input poses for Pose2Mesh are estimated by the method of Sun et al. [60] trained on MPII [3], using groundtruth bounding boxes.

3DPW. We compare MPJPE, PA-MPJPE, and MPVPE of our Pose2Mesh with the previous state-of-the-art 3D body pose and mesh estimation works on 3DPW, which is an in-the-wild dataset, in Table 6. First, when the image-based methods are trained only on Human3.6M, they give extremely high errors. This verifies that the image-based methods suffer from the appearance domain gap between train and test data from controlled and in-the-wild environments respectively. In fact, since Human3.6M is an indoor dataset from the controlled environment, the image appearance from it are very different from in-the-wild image appearance. On the contrary, the 2D pose-based approach of Pose2Mesh can benefit from accurate 3D annotations of the lab-recorded 3D datasets [22] without the appearance domain gap issue, utilizing the homogeneous geometric property of 2D poses from different domains. Indeed, Pose2Mesh gives far lower errors on in-the-wild images from 3DPW, even when it is only trained on Human3.6M while other methods are additionally trained on COCO. The experimental results suggest that a 3D pose and mesh estimation approach may not necessarily require 3D data captured from in-the-wild environments, which is extremely challenging to acquire, to give accurate predictions. The test 2D input poses for Pose2Mesh are estimated by HRNet [58] and Simple [66] trained on

Table 8: The accuracy comparison between state-of-the-art methods and Pose2Mesh on Human3.6M and 3DPW. Different train sets are used. * means synthetic data from AMASS [38] is used. Refer to Section 13 for more details.

method	Human3.6M		3DPW		
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE	MPVPE
SMPLify [6]	-	82.3	-	-	-
Lassner et al. [34]	-	93.9	-	-	-
HMR [27]	88.0	56.8	-	81.3	-
NBF [47]	-	59.9	-	-	-
Pavlakos et al. [52]	-	75.9	-	-	-
Kanazawa et al. [28]	-	56.9	-	72.6	-
GraphCMR [32]	-	50.1	-	70.2	-
Arnab et al. [4]	77.8	54.3	-	72.2	-
SPIN [31]	-	41.1	-	59.2	116.4
Pose2Mesh (Ours)	64.9	46.3	88.9	58.3	106.3
Pose2Mesh (Ours)*	-	-	89.5	56.3	105.3

COCO, using groundtruth bounding boxes. The average precision (AP) of [58] and [66] are 85.1 and 82.8 on 3DPW test set, 72.1 and 70.4 on COCO validation set, respectively.

FreiHAND. We present the comparison between our Pose2Mesh and other state-of-the-art 3D hand pose and mesh estimation works in Table 7. The proposed system outperforms other methods in various metrics, including PA-MPVPE and F-scores. The test 2D input poses for Pose2Mesh are estimated by HRNet [58] trained on FreiHAND [70], using bounding boxes from Mask R-CNN [20] with ResNet-50 backbone [21].

Comparison with different train sets. We report MPJPE and PA-MPJPE of Pose2Mesh trained on Human3.6M, COCO, and MuCo-3DHP, and other methods trained on different train sets in Table 8. The train sets include Human3.6M, COCO, MPII [3], LSP [24], LSP-Extended [25], UP [34], and MPI-INF-3DHP [41]. Each method is trained on a different subset of them. In the table, the errors of [27, 31, 32] decrease by a large margin compared to the errors in Table 5 and 6. Although it shows that the image-based methods can improve the generalizability with weak-supervision on in-the-wild 2D pose datasets, Pose2Mesh still provides the lowest errors in 3DPW, which is the in-the-wild benchmark. This suggests that tackling the domain gap issue to fully benefit from the 3D data of controlled environments is an important task to recover accurate 3D pose and mesh from in-the-wild images. We measured the PA-MPJPE of Pose2Mesh on Human3.6M by testing only on the frontal camera set, following the previous works [27, 31, 32]. In addition, we used 2D human poses estimated from DarkPose [67] as input for 3DPW evaluation, which improved HRNet [58].

Figure 4 shows the qualitative results on COCO validation set and FreiHAND test set. Our Pose2Mesh outputs visually decent human meshes without post-



Fig.4: Qualitative results of the proposed Pose2Mesh. First to third rows: COCO, fourth row: FreiHAND.

processing, such as model fitting [32]. More qualitative results can be found in the supplementary material.

7 Discussion

Although the proposed system benefits from the homogeneous geometric property of input 2D poses from different domains, it could be challenging to recover various 3D *shapes* solely from the pose. While it may be true, we found that the 2D pose still carries necessary information to reason the corresponding 3D shape. In the literature, SMPLify [6] has experimentally verified that under the canonical body pose, utilizing 2D pose significantly drops the body shape fitting error compared to using the mean body shape. We show that Pose2Mesh can recover various body shapes from the 2D pose in the supplementary material.

8 Conclusion

We propose a novel and general system, Pose2Mesh, for 3D human mesh and pose estimation from a 2D human pose. The input 2D pose enables the system to benefit from the 3D data captured from the controlled settings without the appearance domain gap issue. The model-free approach using GraphCNN allows it to fully exploit mesh topology, while avoiding the representation issues of the 3D rotation parameters. We plan to enhance the shape recover capability of Pose2Mesh using denser keypoints or part segmentation, while maintaining the above advantages.

Acknowledgements. This work was supported by IITP grant funded by the Ministry of Science and ICT of Korea (No.2017-0-01780), and Hyundai Motor Group through HMG-SNU AI Consortium fund (No. 5264-20190101).

**Supplementary Material of
“Pose2Mesh: Graph Convolutional Network for
3D Human Pose and Mesh Recovery
from a 2D Human Pose”**

In this supplementary material, we present more experimental results that could not be included in the main manuscript due to the lack of space.

9 Qualitative results

9.1 Shape recovery

We trained and tested Pose2Mesh on SURREAL [62], which have various samples in terms of the body shape, to verify the capability of shape recovery. As shown in Figure 5, Pose2Mesh can recover a 3D body shape corresponding to an input image, though not perfectly. The shape features of individuals, such as the bone length ratio and fatness, are expressed in the outputs of Pose2Mesh. This implies that the information embedded in joint locations (*e.g.* the distance between hip joints) carries a certain amount of shape cue.

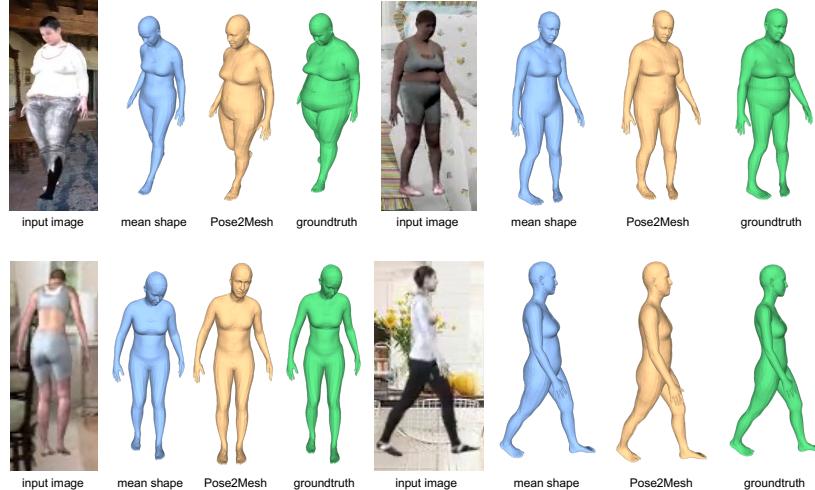


Fig. 5: The Pose2Mesh predictions compared with the groundtruth mesh, and the mesh decoded from groundtruth pose parameters and the mean shape parameters.

9.2 Additional results

Here, we present more qualitative results on COCO [35] validation set and FreiHAND [70] test set in Figure 6. The images at the fourth row show some of

the failure cases. Although the people on the first and second images appear to be overweight, the predicted meshes seem to be closer to the average shape. The right arm pose of the mesh in the third column is bent, though it appears straight.

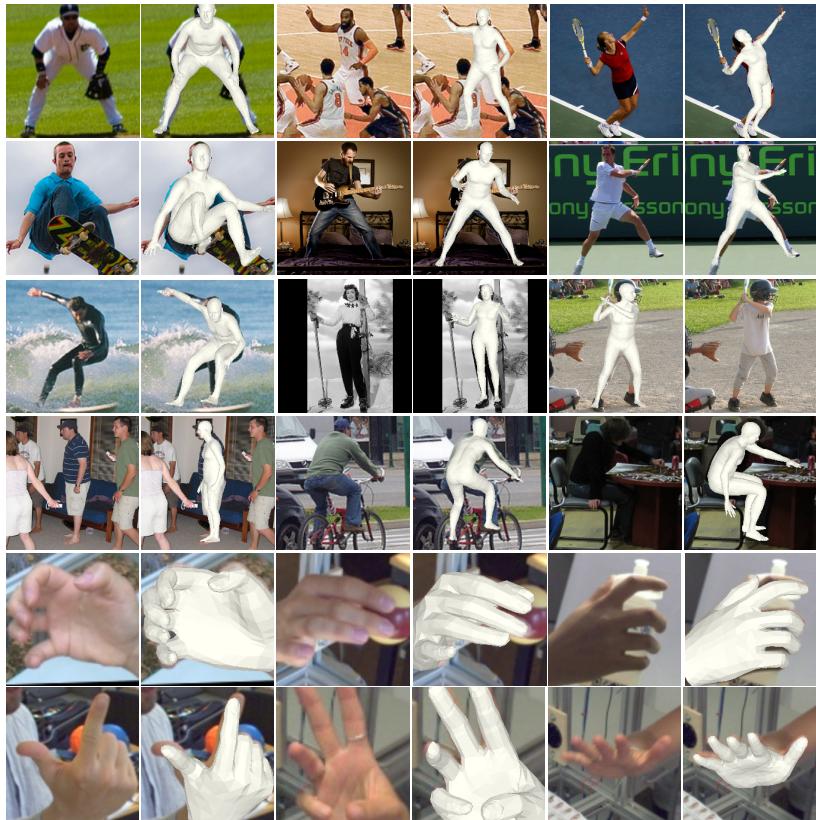


Fig. 6: Additional qualitative results on COCO and FreiHAND.

9.3 Comparison with the state-of-the-art

We present the qualitative comparison between our Pose2Mesh and GraphCMR [32] in Figure 7. We regard GraphCMR as a suitable comparison target, since it is also the model-free method and regresses coordinates of human mesh defined by SMPL [37] using GraphCNN like ours. As the figure shows, our Pose2Mesh provides much more visually pleasant mesh results than GraphCMR. Based on the loss function analysis in Section 7 and the visual results of GraphCMR, we conjecture that the surface losses such as the normal loss and the edge loss are the reason for the difference.

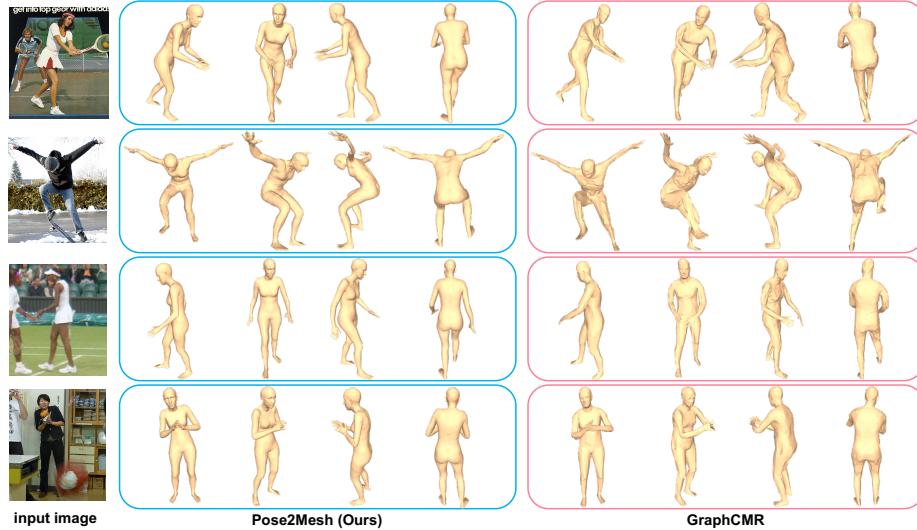


Fig. 7: The mesh quality comparison between our Pose2Mesh and GraphCMR [32].

10 Details of PoseNet

10.1 Network architecture

Figure 8 shows the detailed network architecture of PoseNet. First, the normalized input 2D pose vector is converted to a 4096-dimensional feature vector by a fully-connected layer. Then, it is fed to the two residual blocks, where each block consists of a fully connected layer, 1D batch normalization, ReLU activation, and the dropout. The dimension of the feature map in the residual block is 4096, and the dropout probability is set to 0.5. Finally, the output from the residual block is converted to $(3J)$ -dimensional vector, the 3D pose vector, by a fully-connected layer. The 3D pose vector represents the root-relative 3D pose coordinates.

10.2 Accuracy of PoseNet

We present MPJPE and PA-MPJPE of PoseNet on the benchmarks in Table 9. For the Human3.6M benchmark [22], 14 common joints out of 17 Human3.6M defined joints are evaluated following [27, 31, 32, 52]. For the 3DPW benchmark [39], COCO defined 17 joints are evaluated and \mathcal{JM} from the groundtruth SMPL meshes are used as groundtruth. The 2D pose outputs from [60] and [58] are taken as test inputs on Human3.6M and 3DPW respectively. For the FreiHAND benchmark, only FreiHAND train set is used during training, and 21 MANO [54] hand joints are evaluated by the official evaluation website. The 2D pose outputs from [58] are taken as test inputs.

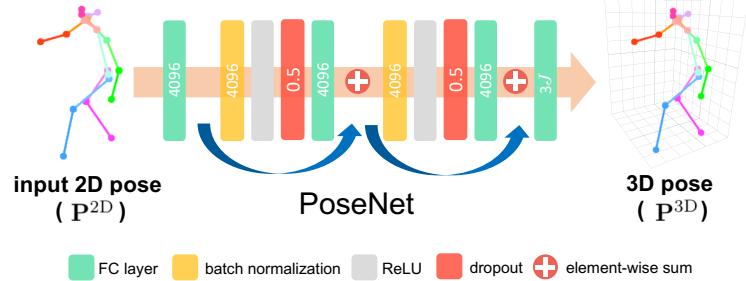


Fig. 8: The detailed network architecture of PoseNet.

Table 9: The MPJPE and PA-MPJPE of PoseNet on each benchmark.

train set	Human3.6M		Human3.6M + COCO	
benchmark	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
Human3.6M	65.1	48.4	66.7	48.9
3DPW	105.0	62.9	99.2	61.0

benchmark	PA-MPJPE
FreiHAND	8.56

11 Pre-defined joint sets and graph structures

We use different pre-defined joint sets and graph structures for Human3.6M, 3DPW, SURREAL, and FreiHAND benchmarks, as shown in Figure 9. To be specific, we employ Human3.6M body joints, COCO body joints, SMPL body joints, MANO hand joints for Human3.6M, 3DPW, SURREAL, FreiHAND benchmarks, respectively, in both training and testing stages. For the COCO joint set, we additionally define pelvis and neck joints that connect the upper body and lower body. The pelvis and neck coordinates are calculated as the middle point of right-left hips and right-left shoulders, respectively.

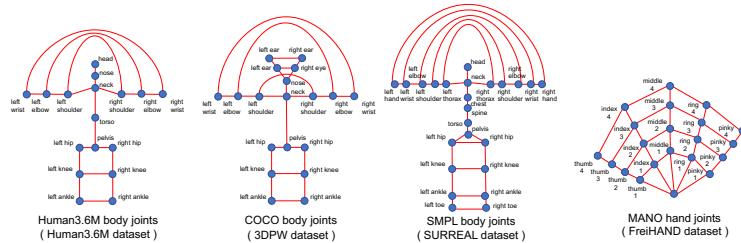


Fig. 9: The joint sets and graph structures of each dataset that are used in Pose2Mesh.

12 Pseudo-groundtruth SMPL parameters of Human3.6M

Mosh [36] method can compute SMPL parameters from the marker data in Human3.6M. Since Human3.6M does not provide 3D mesh annotations, most of the previous 3D pose and mesh estimation papers [27, 31, 32, 52] used the SMPL parameters obtained by Mosh method as the groundtruth for the supervision. However, due to the license issue, the SMPL parameters are not currently available. Furthermore, the source code of Mosh is not publicly released.

For the 3D mesh supervision, we alternatively obtain groundtruth SMPL parameters by applying SMPLify-X [50] on the groundtruth 3D joint coordinates of Human3.6M. Although the obtained SMPL parameters are not perfectly aligned to the groundtruth 3D joint coordinates, we confirmed that the error of the SMPLify-X is much less than those of current state-of-the-art 3D human pose estimation methods, as shown in Table 10. Thus, we believe using SMPL parameters obtained by SMPLify-X as groundtruth is reasonable. For the fair comparison, all the previous works and our system are trained on our SMPL parameters from SMPLify-X.

During the fitting process of SMPLify-X, we adopted a neutral gender SMPL body model. However, we empirically found that the fitting process produces gender-specific body shapes, which correspond to each subject. As a result, since most of the subjects in the training set of Human3.6M are female, our Pose2Mesh trained on Human3.6M tends to produce female body shape meshes. We tried to fix the identity code of the SMPL body model obtained from the T-pose; however, it produces higher errors. Thus, we did not fix the identity code for each subject.

Table 10: The MPJPE comparison between SMPLify-X fitting results and state-of-the-art 3D human pose estimation methods. “*” takes multi-view RGB images as inputs.

methods	MPJPE
Moon et al. [43]	53.3
Sun et al. [60]	49.6
Iskakov et al. [23]*	20.8
SMPLify-X from GT 3D pose	13.1

13 Synthetic data from AMASS

We leverage additional synthetic data from AMASS [38] to boost the performance of Pose2Mesh. AMASS is a new database that unifies 15 different optical marker-based mocap datasets within a common framework. It created SMPL parameters from mocap data by a method named Mosh++. We used CMU [1]

and BML-Movi [16] from the database in training PoseNet and only CMU in training Pose2Mesh.

To be specific, we generated paired 2D pose-3D mesh data by projecting a 3D pose obtained from a mesh to the image plane, using camera parameters from Human3.6M. As shown in Table 11, when AMASS is added, both the joint error and surface error decrease. Exploiting AMASS data in this fashion is not possible for [27], [32], and [31], since they need pairs of image and 2D/3D annotations.

Table 11: The MPJPE and MPVPE of our Pose2Mesh on 3DPW with accumulative training datasets. We used psuedo-groundtruth SMPL parameters of COCO obtained by NeuralAnnot [46]. The 2D pose outputs from [58] are used for input to Pose2Mesh.

train sets	MPJPE	PA-MPJPE	MPVPE
Human3.6M+COCO	92.6	57.9	109.4
Human3.6M+COCO+AMASS	89.5	56.3	105.3

14 Synthesizing the input 2D poses in the training stage

14.1 Detailed description of the synthesis

As described in Section 4.1 of the main manuscript, we synthesize the input 2D poses by adding randomly generated errors on the groundtruth 2D poses in the training stage. For this, we generate errors following Chang et al. [10] and Moon et al. [44] for Human3.6M and COCO body joint sets, respectively. On the other hand, for FreiHAND benchmark, we used detection outputs from [58] on the training set as the input poses in the training stage, since there are no verified synthetic errors for the hand joints.

14.2 Effect of synthesizing the input 2D poses

To demonstrate the validity of the synthesizing process, we compare MPJPE and PA-MPJPE of Pose2Mesh trained with the groundtruth 2D poses, and the synthesized input 2D poses in Table 12. For Human3.6M, only Human3.6M train set is used for the training, and for 3DPW benchmark, Human3.6M and COCO are used for the training. The test 2D input poses used in Human3.6M and 3DPW evaluation are outputs from Integral Regression [60] and HRNet [58] respectively, using groundtruth bounding boxes. Apparently, when our Pose2Mesh is trained with the synthesized input 2D poses, Pose2Mesh performs far better on both benchmarks. This proves that the synthesizing process makes Pose2Mesh more robust to the errors in the input 2D poses and increases the estimation accuracy.

Table 12: The MPJPE and PA-MPJPE comparison according to input type in the training stage.

input pose when training	Human3.6M		3DPW	
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
2D pose GT	70.4	50.6	153.7	94.4
2D pose synthesized (Ours)	64.9	48.7	91.4	60.1

15 Train/test with groundtruth input poses

We present the upper bounds of Pose2Mesh, PoseNet, and MeshNet on Human3.6M and 3DPW benchmarks by training and testing with groundtruth input poses in Table 13. Pose2Mesh and PoseNet take the groundtruth 2D pose as an input, while MeshNet takes the groundtruth 3D pose as an input. As the table shows, the upper bound of Pose2Mesh is similar to that of PoseNet, which implies that the 3D pose errors of Pose2Mesh follow those of PoseNet as analyzed in Section 7.2 of the main manuscript. In addition, the upper bound of MeshNet indicates that we can recover highly accurate 3D human meshes if we can estimate nearly perfect 3D poses.

The MPJPE and PA-MPJPE of Pose2Mesh and MeshNet are measured on the 3D pose regressed from the mesh output, while the accuracy of PoseNet is measured on the lifted 3D pose. For the Human3.6M benchmark, only Human3.6M train set is used to train the network. For the 3DPW benchmark, Human3.6M, COCO, AMASS train sets are used to train the network.

Table 13: The upper bounds of Pose2Mesh, PoseNet, and MeshNet on Human3.6m and 3DPW benchmarks.

networks	Human3.6M		3DPW	
	MPJPE	PA-MPJPE	MPJPE	PA-MPJPE
Pose2Mesh with 2D pose GT	51.1	35.3	65.1	34.6
PoseNet with 2D pose GT	50.6	41.3	66.1	43.8
MeshNet with 3D pose GT	13.9	9.9	10.8	8.1

16 Effect of each loss function

We analyze the effect of joint coordinate loss L_{joint} , surface normal loss L_{normal} , and surface edge loss L_{edge} on reconstructing a 3D human mesh in Table 14 and Figure 10. Human3.6M is used for the training and testing. As the table shows, training without L_{joint} has a relatively distinctive effect on MPJPE and PA-MPJPE, while other settings show numerically negligible differences. On the other hand, as the figure shows, training without L_{normal} or L_{edge} clearly

decreases the visual quality of the mesh output, while training without L_{joint} has nearly no effect on the visual quality of the meshes. To be specific, training without L_{normal} impairs the overall smoothness of the mesh and local details of mouth, hands, and feet. Similarly, training without L_{edge} ruins the details of body parts that have dense vertices, especially mouth, hands, and feet, by making serious artifacts caused by flying vertices.

Table 14: The MPJPE and PA MPJPE comparison between the networks trained from various combinations of loss functions.

settings	MPJPE	PA-MPJPE
full supervision (Ours)	64.9	48.7
without L_{joint}	66.9	50.1
without L_{normal}	64.6	48.5
without L_{edge}	64.8	48.7

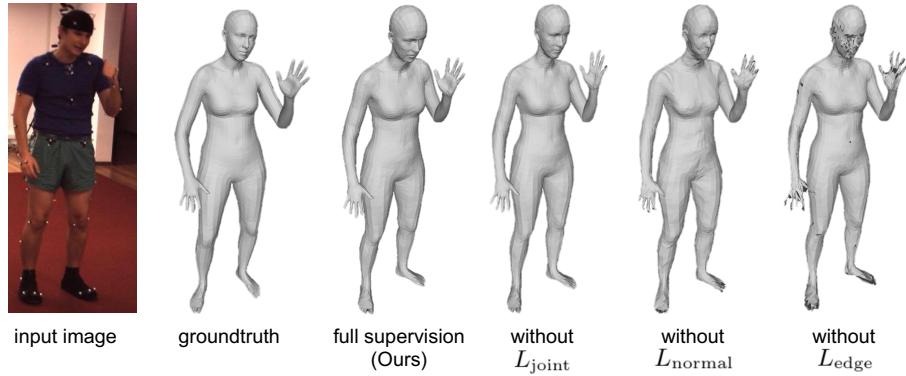


Fig. 10: Qualitative results for the ablation study on the effectiveness of each loss function.

References

1. The Carnegie Mellon University (CMU) Graphics Laboratory Motion Capture Database. <http://mocap.cs.cmu.edu/>
2. Andriluka, M., Iqbal, U., Insafutdinov, E., Pishchulin, L., Milan, A., Gall, J., Schiele, B.: Posetrack: A benchmark for human pose estimation and tracking. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
3. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2d human pose estimation: New benchmark and state of the art analysis. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
4. Arnab, A., Doersch, C., Zisserman, A.: Exploiting temporal context for 3D human pose estimation in the wild. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
5. Baek, S., In Kim, K., Kim, T.K.: Pushing the envelope for rgb-based dense 3d hand pose estimation via neural rendering. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
6. Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J.: Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In: The European Conference on Computer Vision (ECCV) (2016)
7. Boukhayma, A., de Bem, R., Torr, P.H.: 3d hand shape and pose from images in the wild. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
8. Bruna, J., Zaremba, W., Szlam, A., Lecun, Y.: Spectral networks and locally connected networks on graphs. In: The International Conference on Learning Representations (ICLR) (2014)
9. Cai, Y., Ge, L., Liu, J., Cai, J., Cham, T.J., Yuan, J., Thalmann, N.M.: Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
10. Chang, J.Y., Moon, G., Lee, K.M.: Absposelifter: Absolute 3d human pose lifting network from a single noisy 2d human pose. arXiv preprint arXiv:1910.12029 (2020)
11. Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., Sun, J.: Cascaded pyramid network for multi-person pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
12. Chung, F.R.K.: Spectral Graph Theory. American Mathematical Society (1997)
13. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems 29 (NIPS) (2016)
14. Dhillon, I.S., Guan, Y., Kulis, B.: Weighted graph cuts without eigenvectors a multilevel approach. The IEEE transactions on pattern analysis and machine intelligence (TPAMI) (2007)
15. Ge, L., Ren, Z., Li, Y., Xue, Z., Wang, Y., Cai, J., Yuan, J.: 3d hand shape and pose estimation from a single rgb image. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
16. Ghorbani, S., Mahdaviani, K., Thaler, A., Kording, K., Cook, D.J., Blohm, G., Troje, N.F.: Movi: A large multipurpose motion and video dataset. arXiv preprint arXiv:2003.01888 (2020)
17. Gower, J.C.: Generalized procrustes analysis. Psychometrika (1975)

18. Hammond, D., Vandergheynst, P., Gribonval, R.: Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis* (2009)
19. Hasson, Y., Varol, G., Tzionas, D., Kalevatykh, I., Black, M.J., Laptev, I., Schmid, C.: Learning joint reconstruction of hands and manipulated objects. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
20. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
22. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *The IEEE transactions on pattern analysis and machine intelligence (TPAMI)* (2014)
23. Iskakov, K., Burkov, E., Lempitsky, V., Malkov, Y.: Learnable triangulation of human pose. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
24. Johnson, S., Everingham, M.: Clustered pose and nonlinear appearance models for human pose estimation. In: British Machine Vision Conference (BMVC). Citeseer (2010)
25. Johnson, S., Everingham, M.: Learning effective human pose estimation from inaccurate annotation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2011)
26. Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B., Matthews, I., Kanade, T., Nobuhara, S., Sheikh, Y.: Panoptic studio: A massively multiview system for social motion capture. In: The IEEE International Conference on Computer Vision (ICCV) (2015)
27. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
28. Kanazawa, A., Zhang, J.Y., Felsen, P., Malik, J.: Learning 3d human dynamics from video. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
29. Kato, H., Ushiku, Y., Harada, T.: Neural 3d mesh renderer. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
30. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: The International Conference on Learning Representations (ICLR) (2017)
31. Kolotouros, N., Pavlakos, G., Black, M.J., Daniilidis, K.: Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
32. Kolotouros, N., Pavlakos, G., Daniilidis, K.: Convolutional mesh regression for single-image human shape reconstruction. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
33. Lassner, C., Romero, J., Kiefel, M., Bogo, F., Black, M.J., Gehler, P.V.: Unite the people: Closing the loop between 3d and 2d human representations. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
34. Lassner, C., Romero, J., Kiefel, M., Bogo, F., Black, M.J., Gehler, P.V.: Unite the people: Closing the loop between 3d and 2d human representations. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

35. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: The European Conference on Computer Vision (ECCV) (2014)
36. Loper, M., Mahmood, N., Black, M.J.: Mosh: Motion and shape capture from sparse markers. In: Special Interest Group on GRAPHics and Interactive Techniques (SIGGRAPH) (2014)
37. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: A skinned multi-person linear model. In: Special Interest Group on GRAPHics and Interactive Techniques (SIGGRAPH) (2015)
38. Mahmood, N., Ghorbani, N., Troje, N.F., Pons-Moll, G., Black, M.J.: Amass: Archive of motion capture as surface shapes. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
39. von Marcard, T., Henschel, R., Black, M., Rosenhahn, B., Pons-Moll, G.: Recovering accurate 3d human pose in the wild using imus and a moving camera. In: European Conference on Computer Vision (ECCV) (2018)
40. Martinez, J., Hossain, R., Romero, J., Little, J.J.: A simple yet effective baseline for 3d human pose estimation. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
41. Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In: International Conference on 3D Vision (3DV) (2017)
42. Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., Theobalt, C.: Single-shot multi-person 3d pose estimation from monocular rgb. In: International Conference on 3D Vision (3DV) (2018)
43. Moon, G., Chang, J.Y., Lee, K.M.: Camera distance-aware top-down approach for 3d multi-person pose estimation from a single rgb image. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
44. Moon, G., Chang, J.Y., Lee, K.M.: Posefix: Model-agnostic general human pose refinement network. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
45. Moon, G., Lee, K.M.: I2L-MeshNet: Image-to-lixel prediction network for accurate 3D human pose and mesh estimation from a single RGB image. In: The European Conference on Computer Vision (ECCV) (2020)
46. Moon, G., Lee, K.M.: Neuralannot: Neural annotator for in-the-wild expressive 3d human pose and mesh training sets. arXiv preprint arXiv:2011.11232 (2020)
47. Omran, M., Lassner, C., Pons-Moll, G., Gehler, P., Schiele, B.: Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In: International Conference on 3D Vision (3DV) (2018)
48. Panteleris, P., Oikonomidis, I., Argyros, A.: Using a single rgb frame for real time 3d hand pose estimation in the wild. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2018)
49. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
50. Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A.A., Tzionas, D., Black, M.J.: Expressive body capture: 3d hands, face, and body from a single image. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
51. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Coarse-to-fine volumetric prediction for single-image 3d human pose. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

52. Pavlakos, G., Zhu, L., Zhou, X., Daniilidis, K.: Learning to estimate 3d human pose and shape from a single color image. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
53. Ranjan, A., Bolkart, T., Sanyal, S., Black, M.J.: Generating 3d faces using convolutional mesh autoencoders. In: The European Conference on Computer Vision (ECCV) (2018)
54. Romero, J., Tzionas, D., Black, M.J.: Embodied hands: Modeling and capturing hands and bodies together. In: Special Interest Group on GRAPHics and Interactive Techniques (SIGGRAPH) (2017)
55. Ruggero Ronchi, M., Perona, P.: Benchmarking and error diagnosis in multi-instance pose estimation. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
56. Sharma, S., Varigonda, P.T., Bindal, P., Sharma, A., Jain, A.: Monocular 3d human pose estimation by generation and ordinal ranking. In: The IEEE International Conference on Computer Vision (ICCV) (2019)
57. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* (2013)
58. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
59. Sun, X., Shang, J., Liang, S., Wei, Y.: Compositional human pose regression. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
60. Sun, X., Xiao, B., Wei, F., Liang, S., Wei, Y.: Integral human pose regression. In: The European Conference on Computer Vision (ECCV) (2018)
61. Tielemans, T., Hinton, G.: Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning (2012)
62. Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M.J., Laptev, I., Schmid, C.: Learning from synthetic humans. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
63. Verma, N., Boyer, E., Verbeek, J.: Feastnet: Feature-steered graph convolutions for 3d shape analysis. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
64. Wandt, B., Rosenhahn, B.: Repnet: Weakly supervised training of an adversarial reprojection network for 3d human pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
65. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: The European Conference on Computer Vision (ECCV) (2018)
66. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: The European Conference on Computer Vision (ECCV) (2018)
67. Zhang, F., Zhu, X., Dai, H., Ye, M., Zhu, C.: Distribution-aware coordinate representation for human pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
68. Zhao, L., Peng, X., Tian, Y., Kapadia, M., Metaxas, D.N.: Semantic graph convolutional networks for 3d human pose regression. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

69. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
70. Zimmermann, C., Ceylan, D., Yang, J., Russell, B., Argus, M., Brox, T.: Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In: The IEEE International Conference on Computer Vision (ICCV) (2019)