PC-HMR: Pose Calibration for 3D Human Mesh Recovery from 2D Images/Videos

Tianyu Luan [1] *, Yali Wang [1]*, Junhao Zhang [1]*, Zhe Wang [3], Zhipeng Zhou [1], Yu Qiao [1 2]†

[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition,
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
[2]SIAT Branch, Shenzhen Institute of Artificial Intelligence and Robotics for Society
[3]University of California, Irvine
{ty.luan, yl.wang, zhangjh, zp.zhou, yu.qiao}@siat.ac.cn, zwang15@uci.edu

Abstract

The end-to-end Human Mesh Recovery (HMR) approach (Kanazawa et al. 2018) has been successfully used for 3D body reconstruction. However, most HMR-based frameworks reconstruct human body by directly learning mesh parameters from images or videos, while lacking explicit guidance of 3D human pose in visual data. As a result, the generated mesh often exhibits incorrect pose for complex activities. To tackle this problem, we propose to exploit 3D pose to calibrate human mesh. Specifically, we develop two novel Pose Calibration frameworks, i.e., Serial PC-HMR and Parallel PC-HMR. By coupling advanced 3D pose estimators and HMR in a serial or parallel manner, these two frameworks can effectively correct human mesh with guidance of a concise pose calibration module. Furthermore, since the calibration module is designed via non-rigid pose transformation, our PC-HMR frameworks can flexibly tackle bone length variations to alleviate misplacement in the calibrated mesh. Finally, our frameworks are based on generic and complementary integration of data-driven learning and geometrical modeling. Via plug-and-play modules, they can be efficiently adapted for both image/video-based human mesh recovery. Additionally, they have no requirement of extra 3D pose annotations in the testing phase, which releases inference difficulties in practice. We perform extensive experiments on the popular benchmarks, i.e., Human3.6M, 3DPW and SURREAL, where our PC-HMR frameworks achieve the SOTA results.

1 Introduction

3D human mesh reconstruction is an important computer vision task with wide real-life applications such as virtual try-on, robotics, etc. However, it is often challenging to reconstruct human body mesh from images or monocular videos, due to inherent ambiguity in unconstrained environments.

Recently, deep learning has proven to be promising to alleviate such limitation (Kanazawa et al. 2018, 2019; Zhang et al. 2019b; Nikos Kolotouros 2019; Kolotouros et al. 2019; Sun et al. 2019b; Wang, Shin, and Fowlkes 2020; Kocabas, Athanasiou, and Black 2020). In particular, a Human Mesh Recovery (HMR) framework (Kanazawa et al. 2018) has been gradually used as the mainstream architecture for end-to-end 3D reconstruction. However, HMR reconstructs body configuration by directly regressing 3D mesh parameters of SMPL (Loper et al. 2015), while lacking explicit and geometrical knowledge of 3D human pose. As a result, the generated mesh often fails to correctly capture human pose variations in complex activities, e.g., HMR achieves an unsatisfactory mesh on the right arm of the actor in Fig. 1.

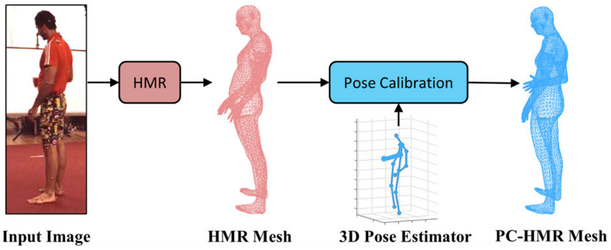


Figure 1: Motivation. The widely-used HMR (Kanazawa et al. 2018) reconstructs human body by directly regressing mesh parameters, while lacking explicit guidance from 3D pose. As a result, it often fails to capture complex pose variations, e.g., the right arm of this actor. To tackle such problem, we propose pose calibration for human mesh recovery (PC-HMR), which can effectively integrate advanced 3D estimator and HMR in a novel manner to correct mesh.

Hence, our basic idea is to use 3D human pose as guidance to reduce structural ambiguity in the 3D mesh. However, the ground truth 3D pose is usually unavailable in the testing phase. Fortunately, 3D pose estimation has recently made remarkable progress (Martinez et al. 2017; Cai et al. 2019; Pavllo et al. 2019), based on the fast development of deep learning. Inspired by this observation, we propose to integrate 3D pose estimators and HMR approaches in a unified fashion, so that we can take advantage of their complementary combination to calibrate human mesh.

Specifically, we introduce three contributions in this work. **First**, we develop two novel frameworks for Pose Calibration (PC), i.e., Serial and Parallel PC-HMR. Serial PC-HMR is a mesh-pose *AutoEncoding* framework. It can adaptively refine 3D pose to correct HMR mesh via multi-level encoding-decoding architecture. Parallel PC-HMR is a mesh-pose *TwoStream* framework. It can directly take advantages of off-the-shelf 3D pose estimators to deform mesh

[*] T.Luan, Y.Wang and J.Zhang contributed equally.

[†] Corresponding author.

in HMR. Based on such serial or parallel manner, these two frameworks can effectively take tradeoffs between reconstruction error and computation cost into account. **Second**, we design a concise pose calibration module for these two frameworks. By leveraging non-rigid pose transformation as guidance, this module can effectively reduce misplacement caused by bone length variations, and thus generate more natural calibrated mesh. **Third**, our frameworks are the generic integration of pose estimators and mesh generators. On one hand, it can flexibly extend human mesh recovery for both images and videos. On the other hand, such plug-and-play design significantly releases training difficulties, and does not need extra 3D pose annotations for inference. We evaluate our PC-HMR frameworks on three popular benchmarks, i.e., Human3.6M, 3DPW, and SURREAL, where we achieve the SOTA results on mesh reconstruction.

2 Related Works

3D Mesh Reconstruction. SMPL (Loper et al. 2015) has been widely used for 3D human mesh reconstruction. To boost its power in practice, a number of deep learning frameworks have been proposed by using SMPL as a mesh generation module (Kanazawa et al. 2018, 2019; Kocabas, Athanasiou, and Black 2020; Sun et al. 2019b; Nikos Kolotouros 2019; Bogo et al. 2016). In particular, HMR (Kanazawa et al. 2018) is one mainstream framework which regresses SMPL parameters directly from input images by end-to-end training. Following this research direction, several extensions have been introduced to improve 3D mesh reconstruction in images (Kolotouros et al. 2019; Kanazawa et al. 2019; Sun et al. 2019b; Guler and Kokkinos 2019; Zeng et al. 2020) or model temporal relations for recovering 3D mesh in videos (Kanazawa et al. 2019; Kocabas, Athanasiou, and Black 2020). Additionally, several non-parametric frameworks have been proposed via voxel-based methods (Saito et al. 2019; Huang et al. 2020). However, most HMR-based approaches lack explicit guidance of 3D pose. As a result, the body parts of generated mesh are often located at unsatisfactory positions. To alleviate such problem, we leverage advanced 3D pose estimators to calibrate mesh in two general PC-HMR frameworks.

3D Pose Estimation. Compared to 3D mesh reconstruction, 3D pose estimation has achieved more successes via deep learning. Basically, most current models can be categorized into two frameworks. The first is to directly estimate 3D pose from images, based on volumetric representation (Pavlakos et al. 2017; Sun et al. 2018; Pavlakos, Zhou, and Daniilidis 2018; Wang, Shin, and Fowlkes 2020; Wang et al. 2019). But these approaches may involve in complex post-processing steps. Based on the explosive improvement in 2D pose estimation (Newell, Yang, and Deng 2016; Chen et al. 2018), another framework is to estimate 2D pose from images and then lift 2D pose to 3D pose (Martinez et al. 2017; Zhao et al. 2019; Ci et al. 2019; Cai et al. 2019; Pavllo et al. 2019). Since these approaches take 2D joint locations as input, 3D human pose estimation simply focuses on learning depth of each joint. This releases learning difficulty and leads to better 3D pose. In this work, we use advanced 3D pose estimators as guidance to calibrate human mesh in the HMR-based reconstruction approaches. This would disentangle human mesh recovery respectively into shape and pose modeling, which effectively alleviates pose ambiguity in the generated mesh.

3 Method

In this section, we first introduce HMR and explain how to build up our PC-HMR frameworks. Then, we design a pose calibration module in our frameworks, which uses pose transformation as guidance to correct HMR mesh.

3.1 3D Human Mesh Recovery

HMR (Kanazawa et al. 2018) is a widely-used deep learning framework to generate 3D human mesh from images. First, it uses CNN to estimate 3D mesh parameters from an input image of human, i.e., $\mathbf{\Theta} = (\boldsymbol{\beta}, \boldsymbol{\theta}, \mathbf{R}, \mathbf{t}, s) = \text{CNN}(Img)$, where $\boldsymbol{\beta}$ refers to human body shape, $\boldsymbol{\theta}$ refers to relative 3D rotation of K joints, and $(\mathbf{R}, \mathbf{t}, s)$ represent camera parameters. Second, it feeds $(\boldsymbol{\beta}, \boldsymbol{\theta})$ into the well-known SMPL (Loper et al. 2015) to generate a triangulated mesh $\mathbf{M}^{hmr}$ with N vertices,

$$\mathbf{M}^{hmr} = [\mathbf{V}_1^{hmr}, \mathbf{V}_2^{hmr}, ..., \mathbf{V}_N^{hmr}]. \tag{1}$$

Third, it uses a linear mesh-to-pose projector to produce 3D pose $\mathbf{J}^{hmr}$,

$$\mathbf{J}^{hmr} = \mathbf{U}\mathbf{M}^{hmr}, \tag{2}$$

where $\mathbf{U}$ is a linear projection matrix. Finally, it applies $(\mathbf{R}, \mathbf{t}, s)$ within a weak-perspective camera model, which is a 3D-to-2D pose projector to obtain 2D pose $\mathbf{Z}^{hmr}$,

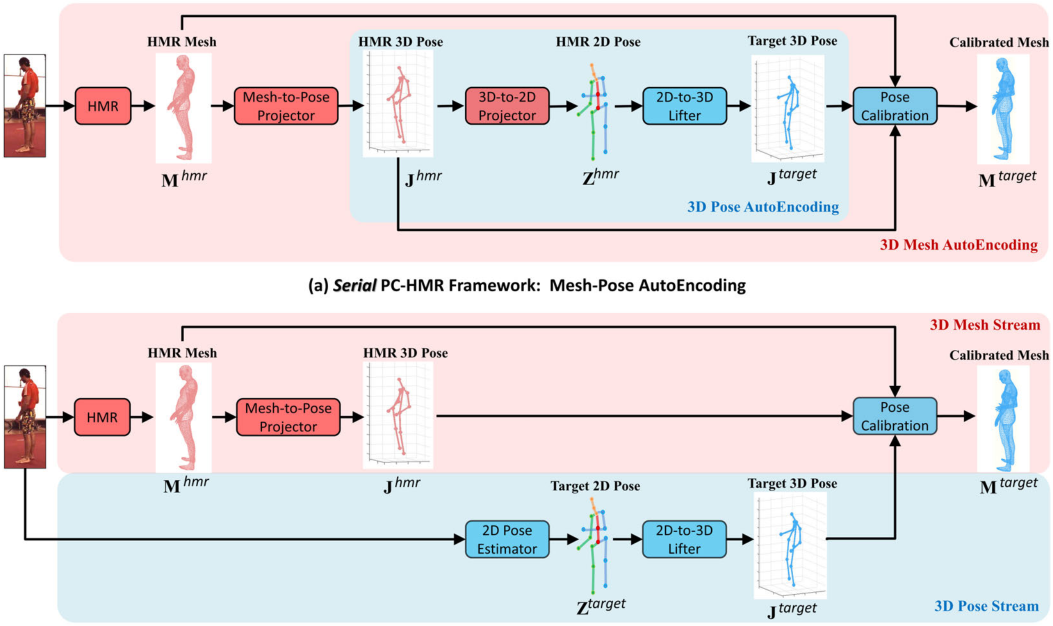
$$\mathbf{Z}^{hmr} = s\Pi(\mathbf{R}\mathbf{J}^{hmr}) + \mathbf{t}, \tag{3}$$

where Π represents the orthographic projection. By adding supervision on ($\mathbf{\Theta}$, $\mathbf{M}^{hmr}$, $\mathbf{J}^{hmr}$, $\mathbf{Z}^{hmr}$), HMR can be efficiently trained in an end-to-end manner. One can refer (Kanazawa et al. 2018) for more details.

3.2 Our PC-HMR Frameworks

Directly regressing mesh parameters $\mathbf{\Theta}$ would introduce interference of shape modeling when learning 3D pose. Hence, HMR-based approaches are often limited when capturing complex pose variations. To tackle this problem, we design two pose calibration frameworks, i.e., serial and parallel PC-HMR. With guidance of target (or reference) 3D pose $\mathbf{J}^{target}$ from pose estimators, serial / parallel PC-HMR can effectively calibrate HMR mesh $\mathbf{M}^{hmr}$ by mesh-pose autoencoding / twostream.

Serial PC-HMR. As shown in Fig. 2 (a), we first adapt HMR to be a multi-level autoencoding framework for calibration. **(I) 3D Pose AutoEncoding**. We use it to generate target 3D pose $\mathbf{J}^{target}$. Specifically, we notice that successful 3D pose estimators often consist of 2D pose estimator and 2D-to-3D lifter in the literature (Martinez et al. 2017; Pavllo et al. 2019). Inspired by this fact, we propose to use the HMR framework as a 2D pose estimator, and introduce 2D-to-3D pose lifter on top of 3D-to-2D pose projector in Eq. (3). In this case, pose projector becomes a pose encoder that encodes HMR 3D pose as HMR 2D pose ($\mathbf{J}^{hmr} \rightarrow$



(a) *Serial* PC-HMR Framework: Mesh-Pose AutoEncoding

(b) *Parallel* PC-HMR Framework: Mesh-Pose TwoStream

Figure 2: PC-HMR Frameworks. Our serial and parallel frameworks provide two generic manners to calibrate HMR mesh with explicit guidance of 3D pose. More details can be found in Section 3.2.

$\mathbf{Z}^{hmr}$), and pose lifter becomes a pose decoder that decodes HMR 2D pose as target 3D pose ($\mathbf{Z}^{hmr} \rightarrow \mathbf{J}^{target}$),

$$\mathbf{J}^{target} = PoseLifter(\mathbf{Z}^{hmr}). \tag{4}$$

It is worth mentioning that, target 3D pose $\mathbf{J}^{target}$ tends to be better than HMR 3D pose $\mathbf{J}^{hmr}$ via our autoencoding manner. The main reason is that, $\mathbf{J}^{hmr}$ is generated from a simple linear projection of HMR mesh $\mathbf{M}^{hmr}$ (Eq. 2), which has limitation to capture pose variations as mentioned before. On the contrary, $\mathbf{J}^{target}$ is generated from an advanced 2D-to-3D pose lifter, which can effectively leverage data-driven deep learning to adjust 2D locations of $\mathbf{Z}^{hmr}$ and estimate depth of each joint. In our experiments, we investigate several well-known pose lifters to show its effectiveness. **(II) 3D Mesh AutoEncoding**. After obtaining target 3D pose $\mathbf{J}^{target}$, we design a pose calibration module to correct HMR mesh $\mathbf{M}^{hmr}$ as calibrated mesh $\mathbf{M}^{target}$. In particular, we add this calibration module on top of 2D-to-3D pose lifter, leading to a 3D mesh autoencoder. First, we use Mesh-to-Pose projector (Eq. 2) as a mesh encoder, which encodes HMR 3D mesh into HMR 3D pose ($\mathbf{M}^{hmr} \rightarrow \mathbf{J}^{hmr}$). Then, we use 3D pose autoencoder to map HMR 3D pose into target 3D pose ($\mathbf{J}^{hmr} \rightarrow \mathbf{J}^{target}$). Finally, we use the calibration module as a mesh decoder, which deforms HMR mesh as calibrated mesh ($\mathbf{M}^{hmr} \rightarrow \mathbf{M}^{target}$),

$$\mathbf{M}^{target} = Calibration(\mathbf{M}^{hmr} | \mathbf{J}^{hmr}, \mathbf{J}^{target}). \tag{5}$$

Note that, our calibration function uses $(\mathbf{J}^{hmr}, \mathbf{J}^{target})$ as condition. This is mainly because, this function leverages non-rigid pose transformation between $\mathbf{J}^{hmr}$ and $\mathbf{J}^{target}$ as effective guidance, to deform HMR mesh as calibrated mesh. We will further explain this module in Section 3.3. **(III) Training Serial PC-HMR**. The output of each encoder and decoder in our serial PC-HMR has its physical meaning such as 3D mesh, 3D pose or 2D pose. This makes our training procedure become convenient and flexible, i.e., we can train each module of our serial PC-HMR separately, and then fine-tune the entire framework. In our experiments, we first pretrain HMR (including mesh-to-pose and 3D-to-2D projectors) and 2D-to-3D pose lifter separately. Then, we fine-tune the entire framework, by adding 3D pose supervision on pose lifter and 3D mesh supervision on pose calibration module. As a result, our serial PC-HMR can effectively calibrate HMR mesh by 3D pose refinement.

Parallel PC-HMR. To obtain better target 3D pose for calibration, we adapt HMR as a twostream framework in Fig. 2 (b). **(I) 3D Pose Stream**. Instead of pose refinement in serial PC-HMR, we directly use advanced 3D pose estimator as extra stream to generate target 3D pose $\mathbf{J}^{target}$, e.g., we first apply 2D pose estimator to obtain 2D pose from the input image ($Img \rightarrow \mathbf{Z}^{target}$), and then use 2D-to-3D pose lifter to estimate target 3D pose from 2D pose ($\mathbf{Z}^{target} \rightarrow \mathbf{J}^{target}$). Note that, 2D pose $\mathbf{Z}^{target}$ is better than $\mathbf{Z}^{hmr}$ in serial PC-HMR. The main reason is that, $\mathbf{Z}^{hmr}$ is indirectly generated through HMR, Mesh-to-Pose projector, 3D-to-2D projector with complex interference of shape modeling. On the contrary, $\mathbf{Z}^{target}$ is directly generated from advanced 2D pose estimator, which is able to predict 2D joint locations accurately from an input image. Subsequently, by using a better 2D pose $\mathbf{Z}^{target}$, target 3D pose $\mathbf{J}^{target}$ from this extra stream tends to be more reliable than the one (Eq. 4) from serial PC-HMR. **(II) 3D Mesh Stream**. After obtaining target 3D pose, we introduce a 3D mesh stream, where we first obtain human mesh from HMR, and then correct it via pose calibration module (Eq. 5). **(III) Training Parallel PC-HMR**. Similar to serial PC-HMR,

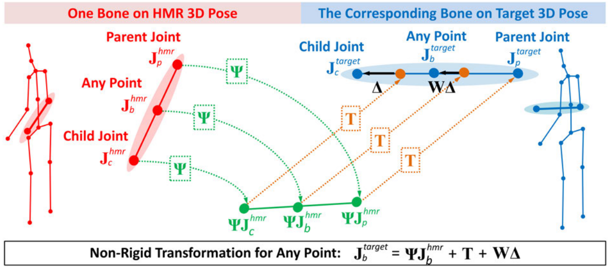


Figure 3: Non-Rigid Pose Transformation. For an arbitrary point $\mathbf{J}_b^{hmr}$ on a bone of HMR 3D pose, we design a non-rigid transformation to find its corresponding point $\mathbf{J}_b^{target}$ on the corresponding bone of target 3D pose. Note that, the bone length of target 3D pose can be different from that of HMR 3D pose. Our non-rigid transformation can effectively tackle this problem by bone extension or shortening (i.e., $\mathbf{W\Delta}$).

parallel PC-HMR can be trained in a flexible manner. In our experiments, we first pretrain mesh stream (HMR+Mesh-to-Pose Projector) and pose stream separately, and then fine-tune the entire framework by adding 3D pose supervision on pose lifter and 3D mesh supervision on calibration module.

Discussions. We discuss the proposed PC-HMR frameworks from the following aspects. **(I) Extension on Video-Based Mesh Recovery**. Our PC-HMR frameworks provide two generic manners to reconstruct human mesh with explicit guidance of 3D pose. Hence, they can be straightforwardly extended for video-based mesh reconstruction, by integrating video-based mesh generators and pose estimators in our plug-and-play fashion. In our experiments, we will show flexibility and robustness of our PC-HMR frameworks for both image/video-based mesh recovery. **(II) Accuracy-Efficiency Tradeoffs**. Our PC-HMR frameworks take tradeoffs between reconstruction error and computation cost into account, i.e., parallel framework achieves a smaller reconstruction error with extra 3D pose estimation stream, while serial framework maintains a lighter computation cost by 3D pose refinement with 2D-to-3D pose lifter. In practice, both can effectively calibrate HMR mesh. We can choose either of them, depending on accuracy-efficiency balance.

3.3 Calibration Module

In this section, we further explain pose calibration module (Eq. 5) in PC-HMR frameworks. To leverage target 3D pose $\mathbf{J}^{target}$ as guidance, we propose to establish transformation between $\mathbf{J}^{target}$ and HMR 3D pose $\mathbf{J}^{hmr}$. Then, we use this pose transformation as reference to deform HMR mesh $\mathbf{M}^{hmr}$ into our calibrated mesh $\mathbf{M}^{target}$. Note that, since $\mathbf{J}^{hmr}$ and $\mathbf{J}^{target}$ may not share the same bone lengths, our transformation is designed to be non-rigid to alleviate misplacement in the calibrated mesh. Moreover, our module is based on geometrical transformation with learnable parameters. Hence, it can take advantages of both physical and data-driven learning to calibrate HMR mesh effectively.

Non-Rigid Pose Transformation. Without loss of generality, we mainly describe how to make non-rigid transformation on one bone. Specifically, for one bone in the HMR 3D pose $\mathbf{J}^{hmr}$, we denote ($\mathbf{J}_p^{hmr}$, $\mathbf{J}_c^{hmr}$, $\mathbf{J}_b^{hmr}$) respectively as the parent joint of this bone, the child joint of this bone, and an arbitrary point on this bone between parent and child joints. For the corresponding bone in the target 3D pose $\mathbf{J}^{target}$, we use the similar notations such as ($\mathbf{J}_p^{target}$, $\mathbf{J}_c^{target}$, $\mathbf{J}_b^{target}$). The parent and child joints in the bone are defined according to human skeleton, i.e., the top joint in the bone is parent, and the bottom joint in the bone is child. Note that, the parent and child joints are given in both HMR and target 3D poses. Hence, *our goal is to build up non-rigid transformation of any point between* $\mathbf{J}_b^{hmr}$ *and* $\mathbf{J}_b^{target}$, based on the parent and child joints ($\mathbf{J}_p^{hmr}$, $\mathbf{J}_c^{hmr}$) and ($\mathbf{J}_p^{target}$, $\mathbf{J}_c^{target}$). The whole transformation process is shown in Fig. 3. **(I) Rotation** $bm\Psi$. We first compute the rotation matrix $\mathbf{\Psi}$, in order to rotate the bone in HMR pose along the direction of the corresponding bone in target pose. Specifically, based on Lie algebra, we can compute the rotation vector $\boldsymbol{\psi}$ between bone directions $\boldsymbol{b}^{hmr}$ and $\boldsymbol{b}^{target}$,

$$\boldsymbol{\psi} = arccos\left(\frac{\boldsymbol{b}^{hmr}\boldsymbol{b}^{target}}{\|\boldsymbol{b}^{hmr}\|\|\boldsymbol{b}^{target}\|}\right)\frac{\boldsymbol{b}^{hmr}\times\boldsymbol{b}^{target}}{\|\boldsymbol{b}^{hmr}\times\boldsymbol{b}^{target}\|}, \quad (6)$$

where $\boldsymbol{b}^{hmr} = \mathbf{J}_p^{hmr} - \mathbf{J}_c^{hmr}$, $\boldsymbol{b}^{target} = \mathbf{J}_p^{target} - \mathbf{J}_c^{target}$ and $\times$ is cross product. Then, we use Rodrigues rotation formula (Koks 2006) to transform rotation vector $\boldsymbol{\psi}$ into rotation matrix $\mathbf{\Psi}$,

$$\mathbf{\Psi} = cos\|\boldsymbol{\psi}\|\mathbf{I} + (1 - cos\|\boldsymbol{\psi}\|)\boldsymbol{\phi}\boldsymbol{\phi}^T + sin\|\boldsymbol{\psi}\|\boldsymbol{\phi}^{\wedge}, \quad (7)$$

where $\boldsymbol{\phi} = \frac{\boldsymbol{\psi}}{\|\boldsymbol{\psi}\|}$ is the unit vector of $\boldsymbol{\psi}$, $\boldsymbol{\phi}^T$ is transpose of $\boldsymbol{\phi}$, and $\boldsymbol{\phi}^{\wedge} = \begin{bmatrix} 0 & -\boldsymbol{\phi}_z & \boldsymbol{\phi}_y \\ \boldsymbol{\phi}_z & 0 & -\boldsymbol{\phi}_x \\ -\boldsymbol{\phi}_y & \boldsymbol{\phi}_x & 0 \end{bmatrix}$ is the cross product matrix of $\boldsymbol{\phi}$.

(II) Translation $\mathbf{T}$. After joint rotation, we need to align the rotated joint of HMR bone with the corresponding joint of target bone. Specifically, we use parent joint as reference, and compute the translation vector $\mathbf{T}$ for alignment,

$$\mathbf{T} = \mathbf{J}_p^{target} - \mathbf{\Psi}\mathbf{J}_p^{hmr}. \quad (8)$$

(III) Non-Rigid Term $\mathbf{\Delta}$. Given rotation and translation, we next transform the child joint of HMR bone $\mathbf{J}_c^{hmr}$ into the space of target bone,

$$\breve{\mathbf{J}}_c^{target} = \mathbf{\Psi}\mathbf{J}_c^{hmr} + \mathbf{T}. \quad (9)$$

However, there may be gap between $\breve{\mathbf{J}}_c^{target}$ and the given child joint of target pose $\mathbf{J}_c^{target}$, due to bone length variations between HMR and target poses. To fill up this gap, we propose a non-rigid term on the child joint,

$$\mathbf{\Delta} = \mathbf{J}_c^{target} - \breve{\mathbf{J}}_c^{target}. \quad (10)$$

As a result, for an arbitrary point on the bone of HMR pose $\mathbf{J}_b^{hmr}$, we can find its corresponding point on the target pose $\mathbf{J}_b^{target}$, according to a non-rigid transformation as follows,

$$\mathbf{J}_b^{target} = \mathbf{\Psi}\mathbf{J}_b^{hmr} + \mathbf{T} + \mathbf{W}\mathbf{\Delta}. \quad (11)$$

As shown in Eq. (11), we first transform $\mathbf{J}_b^{hmr}$ into the space of target pose, i.e., $\breve{\mathbf{J}}_b^{target} = \mathbf{\Psi}\mathbf{J}_b^{hmr} + \mathbf{T}$. Then, we further

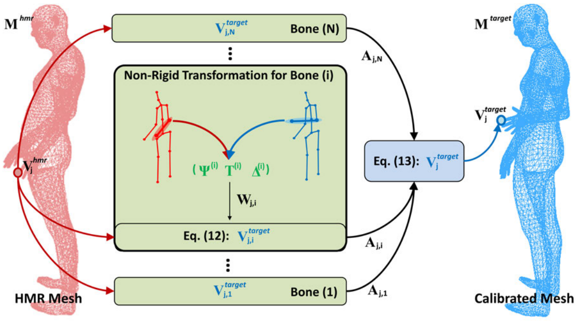


Figure 4: Mesh calibration with guidance of non-rigid 3D pose transformation. For the j-th vertex $\mathbf{V}_j^{hmr}$ in the HMR mesh, we first transform it as $\mathbf{V}_{j,i}^{target}$ in Eq. (12), according to pose transformation of the i-th bone $(\mathbf{\Psi}^{(i)}, \mathbf{T}^{(i)}, \mathbf{\Delta}^{(i)})$. Then, we summarize the contribution of all the bones by a learnable weight matrix $\mathbf{A}$, and produce the final prediction of the j-th vertex in the target mesh by Eq. (13).

improve $\breve{\mathbf{J}}_b^{target}$ by proportionally adjusting non-rigid term $\mathbf{W\Delta}$ with a learnable parameter matrix $\mathbf{W}$. In this case, our non-rigid transformation can extend or shorten $\breve{\mathbf{J}}_b^{target}$ along the target bone, in order to make correct alignment between HMR and target 3D poses.

Mesh Calibration. After obtaining pose transformation in Eq. (11), we use it as an distinct guidance to calibrate HMR mesh. Specifically, we assume that each vertex on the mesh follows the similar transformation in Eq. (11). In this case, we can transform the j-th vertex $\mathbf{V}_j^{hmr}$ in the HMR mesh $\mathbf{M}^{hmr}$ as

$$\mathbf{V}_{j,i}^{target} = \mathbf{\Psi}^{(i)}\mathbf{V}_j^{hmr} + \mathbf{T}^{(i)} + \mathbf{W}_{j,i}\mathbf{\Delta}^{(i)}, \qquad (12)$$

where $\mathbf{V}_{j,i}^{target}$ is the prediction of the corresponding vertex in the target mesh, according to pose transformation of the i-th bone $(\mathbf{\Psi}^{(i)}, \mathbf{T}^{(i)}, \mathbf{\Delta}^{(i)})$. Subsequently, we summarize the contribution of all the bones by a learnable weight matrix $\mathbf{A}$, and produce the final prediction of the j-th vertex in the target mesh $\mathbf{M}^{target}$,

$$\mathbf{V}_j^{target} = \sum\nolimits_i \mathbf{A}_{j,i}\mathbf{V}_{j,i}^{target}. \qquad (13)$$

We can see in Eq. (12)-(13) that, $\mathbf{V}_j^{target}$ is generated by non-rigid pose transformation with learnable parameters $\mathbf{W}_{j,i}$ and $\mathbf{A}_{j,i}$. Hence, our calibration module can integrate geometrical modeling and data-driven learning together to effectively calibrate HMR mesh.

4 Experiments

Datasets and Implementation Details. To evaluate our PC-HMR frameworks, we investigate extensive experiments on three popular benchmarks, i.e., Human3.6M (Ionescu et al. 2014), 3DPW (von Marcard et al. 2018) and SURREAL (Varol et al. 2017). Specifically, Human3.6M is a popular motion capture dataset. We use 5 subjects (S1, S5, S6, S7 and S8) for training and 2 subjects (S9 and S11) for testing. 3DPW is an in-the-wild dataset with multiple actors occurred in the same image. We use its official data split for training and testing. SURREAL is a large-scale synthetic dataset with SMPL body annotations. We directly evaluate its test set by our model pretrained on Human3.6M to show generalization capacity. Moreover, as suggested in the literature (Kanazawa et al. 2018; Sun et al. 2019b; Rong et al. 2019; Zimmermann and Brox 2017), we mainly use three protocols to measure accuracy of our generated mesh, i.e., Mean Per Joint Position Error (MPJPE), Procrustes Aligned MPJPE (PA-MPJPE), Mean Per Vertex Error (MPVE). We implement our PC-HMR frameworks as follows. First, we choose HMR (Kanazawa et al. 2018) as our basic architecture. Second, for Human3.6m and SURREAL, we use CPN (Hong et al. 2018) as 2D pose estimator, and VideoPose3D (Pavllo et al. 2019) as 2D-to-3D pose lifter in our framework. For 3DPW, we use PoseNet (Moon, Chang, and Lee 2019) as 3D pose estimator in our framework. Finally, we train all the modules separately, according to their official codes with default hyperparameter and supervision settings. Then, we fine-tune the entire framework using 3D mesh as supervision, where we set 90/50 training epochs with 1024/128 mini-batch size for Human3.6M and 3DPW. We set the learning rate as 0.001 for our calibration module and 1×10^{-5} for other modules. We implement our frameworks by PyTorch. All the codes and models will be released afterwards.

Human3.6M	MPJPE↓	PA-MPJPE↓	MPVE↓
Self-mocap (Tung et al. 2017)	98.4	-	145.8
HMR (Kanazawa et al. 2018)	88.0	56.8	96.1
HMMR (Kanazawa et al. 2019)	85.2	56.7	94.2
TemporalContext (Arnab, Doersch, and Zisserman 2019)	77.8	54.3	-
GraphCMR (Kolotouros, Pavlakos, and Daniilidis 2019)	-	50.1	-
VIBE (Kocabas, Athanasiou, and Black 2020)	65.9	41.5	-
Pose2Mesh (Choi, Moon, and Lee 2020)	64.9	47.0	-
HoloPose (Guler and Kokkinos 2019)	60.3	46.5	-
HKMR (Georgakis et al. 2020)	59.6	-	-
DSD-SATN (Sun et al. 2019b)	59.1	42.4	-
I2L-MeshNet (Moon and Lee 2020)	55.7	41.7	-
DaNet (Zhang et al. 2019a)	54.6	42.9	66.5
Occluded (Zhang, Huang, and Wang 2020)	-	41.7	-
SPIN (Kolotouros et al. 2019)	-	41.1	-
DecoMR (Zeng et al. 2020)	-	39.3	-
Our PC-HMR	**47.9**	**37.3**	**61.1**

3DPW	MPJPE↓	PA-MPJPE↓	MPVE↓
HMR (Kanazawa et al. 2018)	128.4	81.8	152.7
DSD-SATN (Sun et al. 2019b)	122.7	69.5	183.4
SPIN (Kolotouros et al. 2019)	96.9	59.2	116.4
VIBE (Kocabas, Athanasiou, and Black 2020)	93.5	**56.5**	113.4
I2L-MeshNet (Moon and Lee 2020)	93.2	58.6	-
Pose2Mesh (Choi, Moon, and Lee 2020)	89.2	58.9	-
Our PC-HMR	**87.8**	66.9	**108.6**

SURREAL	MPJPE↓	PA-MPJPE↓	MPVE↓
HMR (Kanazawa et al. 2018)	73.6	55.4	85.1
Self-mocap (Tung et al. 2017)	64.4	-	74.5
Bodynet (Varol et al. 2018)	-	-	73.6
Our PC-HMR	**51.7**	**37.9**	**59.8**

Table 1: SOTA comparison for human mesh reconstruction on Human3.6M, 3DPW and SURREAL datasets. For most metrics and benchmarks, our parallel PC-HMR framework achieves the SOTA performance, e.g., for Human3.6M, it outperforms (Sun et al. 2019b; Choi, Moon, and Lee 2020) that also leverage pose estimators for mesh reconstruction. This shows our PC-HMR framework is a more effective manner to boost mesh recovery by human pose.

4.1 SOTA Comparison

In Table 1, our parallel PC-HMR framework achieves the SOTA performance, e.g., for Human3.6M, it outperforms

Designs	Mesh Generator		2D-to-3D Pose Lifter		2D Pose Estimator		MPVE↓	GFLOPs↓
	HMR (Kanazawa et al. 2018)	HMMR (Kanazawa et al. 2019)	SemanticGCN (Zhao et al. 2019)	VideoPose3D (Pavllo et al. 2019)	CPN (Hong et al. 2018)	HRNet (Sun et al. 2019a)		
Baseline	✓	-	-	-	-	-	96.1	3.5
	-	✓	-	-	-	-	94.2	3.6
Serial	✓	-	✓	-	-	-	91.9	3.5
	✓	-	-	✓	-	-	82.7	3.5
	-	✓	✓	-	-	-	90.3	3.6
	-	✓	-	✓	-	-	82.7	3.6
Parallel	✓	-	✓	-	✓	-	79.0	16.1
	✓	-	✓	-	-	✓	77.3	19.5
	✓	-	-	✓	✓	-	61.1	16.1
	✓	-	-	✓	-	✓	68.6	19.5
	-	✓	✓	-	✓	-	77.1	16.2
	-	✓	✓	-	-	✓	76.2	19.6
	-	✓	-	✓	✓	-	61.1	16.2
	-	✓	-	✓	-	✓	68.4	19.6

Table 2: Generalization capacity (Human3.6M). Our frameworks can be straightforwardly used for video-based mesh reconstruction without any difficulties, e.g., we apply video-based HMMR (Kanazawa et al. 2019) as mesh generator, or apply video-based VideoPose3D (Pavllo et al. 2019) as 2D-to-3D pose lifter. More explanations can be found in Section 4.2.

Method	HMR	Our PC-HMR	w/o Non-Rigid	w/o Fine-Tune	PC-Template	GT 3D Pose	Extra Self-Rotation	w/o Shape Compensation
MPVE	96.1	61.1	69.3	85.5	125.72	29.9	61.0	61.2

Table 3: Detailed designs of our PC-HMR (Human3.6M).

(Sun et al. 2019b; Choi, Moon, and Lee 2020) that also leverage pose estimators for reconstruction. This shows our PC-HMR framework is a more effective manner to boost mesh recovery by human pose. Additionally, for most metrics and benchmarks, our PC-HMR outperforms SPIN (Kolotouros et al. 2019) that plugs iterative SMPLify optimizer into HMR for further calibrating mesh parameters. It shows our pose calibration is a more preferable calibration design than traditional optimization.

4.2 Ablation Studies

Generalization Capacity. We mainly use Human3.6M to further investigate properties of our PC-HMR frameworks. In Table 2, we examine generalization capacity of our frameworks, via changing different mesh generators, 2D-to-3D pose lifters and 2D pose estimators. First, both serial and parallel frameworks outperform baselines, no matter which types of mesh generators, pose lifters and estimators we use. It shows the effectiveness of our pose calibration. Second, our proposed frameworks take tradeoffs between recovery error and computation cost, i.e., Serial PC-HMR is lighter while Parallel PC-HMR is more accurate. This provides more reasonable choices in practice, depending on which factor is important in the deployment. Third, both frameworks can be straightforwardly used for video-based mesh reconstruction without any difficulties. For example, we use video-based HMMR (Kanazawa et al. 2019) as mesh generator, or use video-based VideoPose3D (Pavllo et al. 2019) as 2D-to-3D pose lifter in our frameworks. We can see that, video-based frameworks outperform image-based frameworks, no matter which styles we use (serial or parallel). It shows the importance of using temporal information. Moreover, VideoPose3D (Pavllo et al. 2019) is more critical than HMMR (Kanazawa et al. 2019) for temporal modeling, e.g., when we choose VideoPose3D as pose lifter, the performance would be comparable no matter which mesh generators we use (HMR or HMMR). It indicates that, it is more effective to introduce temporal modeling for 3D human pose, without complex interference of learning mesh shape. Finally, we use 2D pose in the official code of VideoPose3D (Pavllo et al. 2019), where 2D pose estimator refers to CPN (Hong et al. 2018). Hence, CPN (Hong et al. 2018) and VideoPose3D (Pavllo et al. 2019) are more compatible to be a 3D pose estimator, which leads to the best accuracy in our parallel framework. For other parallel cases, the better 2D pose estimator (e.g. HRNet) achieves a better performance of mesh recovery as expected.

Detailed Designs. Since MPVE directly reflects recovery error of mesh surface, we use it to evaluate the detailed designs. Additionally, we choose the parallel framework for this study, due to its higher accuracy. **(I) w/o Non-Rigid**. For comparison, we delete the non-rigid term $\mathbf{W}_{j,i}\mathbf{\Delta}^{(i)}$ of Eq. (12) in our pose calibration module. As shown in Table 3, the non-rigid one achieves a smaller mesh error, showing its effectiveness. **(II) w/o Fine-Tune**. In our design, we fine-tune the entire framework after training each module separately. We delete this fine-tuning procedure for comparison. As expected, the setting of fine-tuning is better, since our pose calibration module becomes compatible for mesh recovery via end-to-end learning. Additionally, our pose calibration module can still boost HMR, even without fine-tuning. It mainly thanks to geometry modeling in this module. **(III) PC-Template**. In our parallel framework, we replace the HMR mesh stream by a template mesh, and operate pose calibration module to correct this template mesh to be target mesh of the input image. In Table 3, our PC-HMR outperforms PC-Template. It shows that HMR provides more preferable shape in the mesh, and thus it is reasonable to integrate our pose calibration module in HMR. **(IV) GT 3D Pose**. We use GT 3D pose for comparison. As expected, GT 3D pose can achieve a better performance. But still, our PC-HMR with estimated 3D pose significantly outperforms HMR. **(V) Extra Self-Rotation**. There may exist a relative rotation around the bone itself. Hence, we use a two-layer

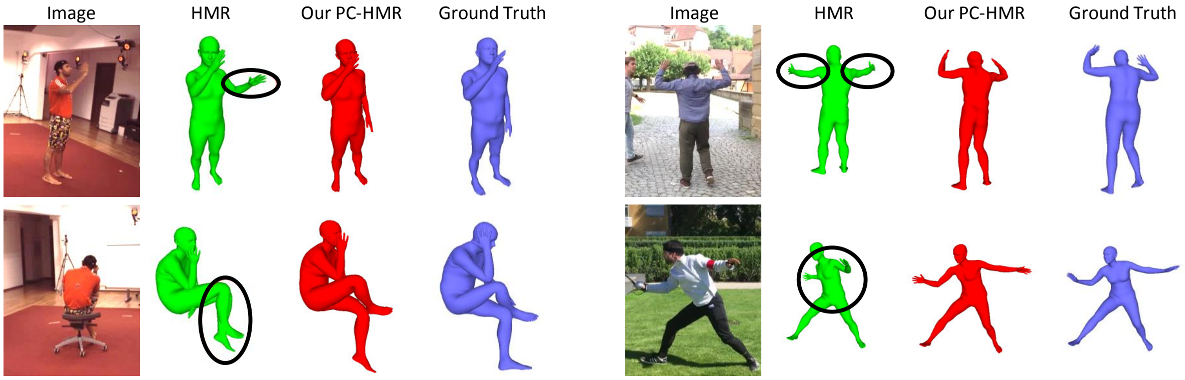


Figure 5: Human Mesh Visualization. The left/right columns are respectively from Human3.6M/3DPW.

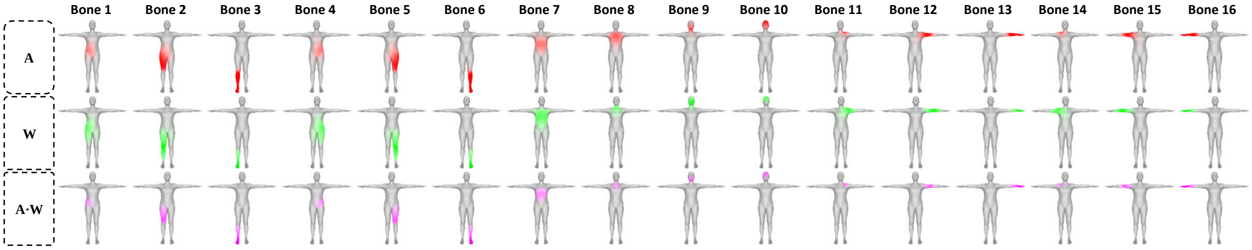


Figure 6: Weight Visualization. For bone i, we visualize the trained $\mathbf{A}(:,i)$, $\mathbf{W}(:,i)$, and $\mathbf{A}(:,i)\cdot\mathbf{W}(:,i)$ on the mesh, where the grey color indicates weights are close to zero, while the bright color means weights are close to one.

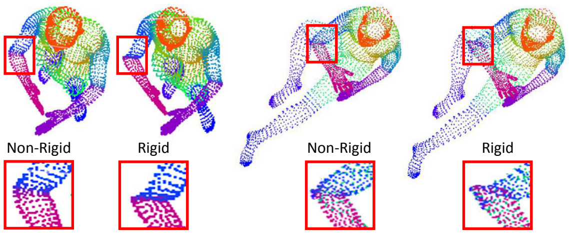


Figure 7: Non-Rigid vs. Rigid Transformation.

MLP to estimate it, where we take mesh parameters $\mathbf{\Theta}$ in HMR and $\mathbf{\Psi}$ in our pose calibration module as input. Then we multiply the self rotation matrix with Eq. (7) as final rotation matrix in our calibration module. In Table 3, the results are comparable between our default and extra self-rotation setting. For simplicity, we use our default setting in the experiments. **(VI) w/o Shape Compensation**. To reduce detailed shape error, we use a two-layer MLP to estimate 3D rotation of keypoints in the calibrated mesh, with input of mesh parameters $\mathbf{\Theta}$ in HMR and $\mathbf{\Psi}$ in our pose calibration module. Then we feed the estimated 3D rotation into pose blend shape function of SMPL as post-processing compensation in our framework. In Table 3, the w/o shape compensation setting is slightly worse. Hence, we use our default setting.

4.3 Visualization

Mesh Visualization. We visualize HMR (baseline) and our PC-HMR (parallel) in Fig. 5. As expected, our PC-HMR can generate 3D mesh with more reliable pose, even for occluded (e.g., Human3.6M) or in-the-wild (e.g., 3DPW) scenarios. It indicates the effectiveness of our model.

Weight Visualization. We visualize the trained weights of our pose calibration module, i.e., $\mathbf{A}$ in Eq. (13), $\mathbf{W}$ in Eq. (12) and $\mathbf{A}\cdot\mathbf{W}$. The parallel framework is used as illustration in Fig. 6. First, $\mathbf{A}(:,i)$ controls the importance of bone i when obtaining each final vertex in the calibrated mesh. Hence, bone i has more contribution on the vertices around it. Second, according to Eq. (12)-(13), $\mathbf{A}(:,i)\cdot\mathbf{W}(:,i)$ controls the proportion of non-rigid term $\mathbf{\Delta}^{(i)}$ in bone i. Via further learning $\mathbf{W}(:,i)$, we can see that $\mathbf{A}(:,i)\cdot\mathbf{W}(:,i)$ is gradually highlighted around the child joint of each bone, in order to proportionally adjust bone length in the calibrated mesh. Both facts allow us to calibrate HMR mesh smoothly and reasonably with non-rigid transformation.

Calibration Module. We further show non-rigid transformation in our pose calibration module. We use parallel framework as illustration in Fig. 7. As expected, the generated mesh with our non-rigid transformation is much more natural, without significant misplacement.

5 Conclusion

In this paper, we design two generic plug-and-play PC-HMR frameworks to calibrate human mesh with explicit guidance of 3D pose. They leverage a non-rigid pose calibration module to couple HMR mesh generators and 3D pose estimators in the serial or parallel manner, so that they can be flexibly applied for image/video-based mesh recovery, and have no requirement of 3D pose annotations in the testing. The extensive experiments on popular benchmarks show our frameworks significantly boost recovery performance.

Acknowledgments

This work is partially supported by Guangdong Special Support Program (2016TX03X276), and National Natural Science Foundation of China (61876176,U1713208), Shenzhen Basic Research Program (CXB201104220032A), the Joint Lab of CAS-HK.

References

Arnab, A.; Doersch, C.; and Zisserman, A. 2019. Exploiting temporal context for 3D human pose estimation in the wild. In *CVPR*.

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*.

Cai, Y.; Ge, L.; Liu, J.; Cai, J.; Cham, T.-J.; Yuan, J.; and Thalmann, N. M. 2019. Exploiting Spatial-Temporal Relationships for 3D Pose Estimation via Graph Convolutional Networks. In *ICCV*.

Chen, Y.; Wang, Z.; Peng, Y.; Zhang, Z.; Yu, G.; and Sun, J. 2018. Cascaded Pyramid Network for Multi-Person Pose Estimation. In *CVPR*.

Choi, H.; Moon, G.; and Lee, K. M. 2020. Pose2Mesh: Graph Convolutional Network for 3D Human Pose and Mesh Recovery from a 2D Human Pose. In *ECCV*.

Ci, H.; Wang, C.; Ma, X.; and Wang, Y. 2019. Optimizing Network Structure for 3D Human Pose Estimation. In *ICCV*.

Georgakis, G.; Li, R.; Karanam, S.; Chen, T.; Kosecka, J.; and Wu, Z. 2020. Hierarchical Kinematic Human Mesh Recovery. In *ECCV*.

Guler, R. A.; and Kokkinos, I. 2019. Holopose: Holistic 3d human reconstruction in-the-wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 10884–10894.

Hong, S.; Jung, W.; Woo, I.; and Kim, S. W. 2018. Cascaded Pyramid Network for 3D Human Pose Estimation Challenge. *arXiv:1810.01616* .

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

Ionescu, C.; Papava, D.; Olaru, V.; and Sminchisescu, C. 2014. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE T-PAMI* .

Kanazawa, A.; Black, M. J.; Jacobs, D. W.; and Malik, J. 2018. End-to-end Recovery of Human Shape and Pose. In *CVPR*.

Kanazawa, A.; Zhang, J. Y.; Felsen, P.; and Malik, J. 2019. Learning 3D Human Dynamics from Video. In *CVPR*.

Kocabas, M.; Athanasiou, N.; and Black, M. J. 2020. VIBE: Video Inference for Human Body Pose and Shape Estimation. In *CVPR*.

Koks, D. 2006. *Explorations in mathematical physics: the concepts behind an elegant language*. Springer Science & Business Media.

Kolotouros, N.; Pavlakos, G.; Black, M. J.; and Daniilidis, K. 2019. Learning to Reconstruct 3D Human Pose and Shape via Model-fitting in the Loop. In *ICCV*.

Kolotouros, N.; Pavlakos, G.; and Daniilidis, K. 2019. Convolutional Mesh Regression for Single-Image Human Shape Reconstruction. In *CVPR*.

Loper, M.; Mahmood, N.; Romero, J.; Pons-Moll, G.; and Black, M. J. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics* .

Martinez, J.; Hossain, R.; Romero, J.; and Little, J. J. 2017. A simple yet effective baseline for 3d human pose estimation. In *ICCV*.

Moon, G.; Chang, J.; and Lee, K. M. 2019. Camera Distance-aware Top-down Approach for 3D Multi-person Pose Estimation from a Single RGB Image. In *ICCV*.

Moon, G.; and Lee, K. M. 2020. I2L-MeshNet: Image-to-Lixel Prediction Network for Accurate 3D Human Pose and Mesh Estimation from a Single RGB Image. In *ECCV*.

Newell, A.; Yang, K.; and Deng, J. 2016. Stacked Hourglass Networks for Human Pose Estimation.

Nikos Kolotouros, Georgios Pavlakos, K. D. 2019. Convolutional Mesh Regression for Single-Image Human Shape Reconstruction. *CVPR* .

Pavlakos, G.; Zhou, X.; and Daniilidis, K. 2018. Ordinal Depth Supervision for 3D Human Pose Estimation. In *CVPR*.

Pavlakos, G.; Zhou, X.; Derpanis, K. G.; and Daniilidis, K. 2017. Coarse-to-Fine Volumetric Prediction for Single-Image 3D Human Pose. In *CVPR*.

Pavllo, D.; Feichtenhofer, C.; Grangier, D.; and Auli, M. 2019. 3D human pose estimation in video with temporal convolutions and semi-supervised training. In *CVPR*.

Rong, Y.; Liu, Z.; Li, C.; Cao, K.; and Change Loy, C. 2019. Delving Deep into Hybrid Annotations for 3D Human Recovery in the Wild. In *ICCV*.

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. *ICCV* .

Sun, K.; Xiao, B.; Liu, D.; and Wang, J. 2019a. Deep High-Resolution Representation Learning for Human Pose Estimation. In *CVPR*.

Sun, X.; Xiao, B.; Wei, F.; Liang, S.; and Wei, Y. 2018. Integral Human Pose Regression. In *ECCV*.

Sun, Y.; Ye, Y.; Liu, W.; Gao, W.; Fu, Y.; and Mei, T. 2019b. Human Mesh Recovery from Monocular Images via a Skeleton-disentangled Representation. In *ICCV*.

Tung, H.-Y. F.; Tung, H.-W.; Yumer, E.; and Fragkiadaki, K. 2017. Self-supervised Learning of Motion Capture. In *NeurIPS*.

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*.

Varol, G.; Romero, J.; Martin, X.; Mahmood, N.; Black, M. J.; Laptev, I.; and Schmid, C. 2017. Learning from Synthetic Humans. In *CVPR*.

von Marcard, T.; Henschel, R.; Black, M.; Rosenhahn, B.; and Pons-Moll, G. 2018. Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera. In *ECCV*.

Wang, Z.; Chen, L.; Rathore, S.; Shin, D.; and Fowlkes, C. 2019. Geometric Pose Affordance: 3D Human Pose with Scene Constraints. In *Arxiv*.

Wang, Z.; Shin, D.; and Fowlkes, C. 2020. Predicting Camera Viewpoint Improves Cross-dataset Generalization for 3D Human Pose Estimation. In *ECCVW*.

Zeng, W.; Ouyang, W.; Luo, P.; Liu, W.; and Wang, X. 2020. 3D Human Mesh Regression with Dense Correspondence. In *CVPR*.

Zhang, H.; Cao, J.; Lu, G.; Ouyang, W.; and Sun, Z. 2019a. DaNet: Decompose-and-aggregate Network for 3D Human Shape and Pose Estimation. In *ACMMM*.

Zhang, J. Y.; Felsen, P.; Kanazawa, A.; and Malik, J. 2019b. Predicting 3D Human Dynamics from Video. In *ICCV*.

Zhang, T.; Huang, B.; and Wang, Y. 2020. Object-Occluded Human Shape and Pose Estimation From a Single Color Image. In *CVPR*.

Zhao, L.; Peng, X.; Tian, Y.; Kapadia, M.; and Metaxas, D. N. 2019. Semantic Graph Convolutional Networks for 3D Human Pose Regression. *CVPR* .

Zimmermann, C.; and Brox, T. 2017. Learning to Estimate 3D Hand Pose from Single RGB Images. In *ICCV*.