Hierarchical Kinematic Human Mesh Recovery

Georgios Georgakis[1,2], Ren Li[1], Srikrishna Karanam[1(✉)], Terrence Chen[1], Jana Košecká[2], and Ziyan Wu[1]

[1] United Imaging Intelligence, Cambridge, MA, USA
{georgios.georgakis,ren.li,srikrishna.karanam,terrence.chen,
ziyan.wu}@united-imaging.com
[2] George Mason University, Fairfax, VA, USA
kosecka@cs.gmu.edu

Abstract. We consider the problem of estimating a parametric model of 3D human mesh from a single image. While there has been substantial recent progress in this area with direct regression of model parameters, these methods only implicitly exploit the human body kinematic structure, leading to sub-optimal use of the model prior. In this work, we address this gap by proposing a new technique for regression of human parametric model that is explicitly informed by the known hierarchical structure, including joint interdependencies of the model. This results in a strong prior-informed design of the regressor architecture and an associated hierarchical optimization that is flexible to be used in conjunction with the current standard frameworks for 3D human mesh recovery. We demonstrate these aspects by means of extensive experiments on standard benchmark datasets, showing how our proposed new design outperforms several existing and popular methods, establishing new state-of-the-art results. By considering joint interdependencies, our method is equipped to infer joints even under data corruptions, which we demonstrate by conducting experiments under varying degrees of occlusion.

1 Introduction

We consider 3D human mesh recovery, *i.e.*, fitting a parametric model to an image of a person that best explains the body pose and shape. With a variety of applications [1,2], there has been notable recent interest in this field [3–6].

The dominant paradigm for this problem involves an *encoder-regressor* architecture [5]; the deep CNN encoder takes the input image and produces the feature representation which the regressor processes to produce the model parameters.

G. Georgakis and R. Li—Joint first authors and contributed equally to this work done during their internships with United Imaging Intelligence, Cambridge MA, USA.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58520-4_45) contains supplementary material, which is available to authorized users.

A. Vedaldi et al. (Eds.): ECCV 2020, LNCS 12362, pp. 768–784, 2020.
https://doi.org/10.1007/978-3-030-58520-4_45

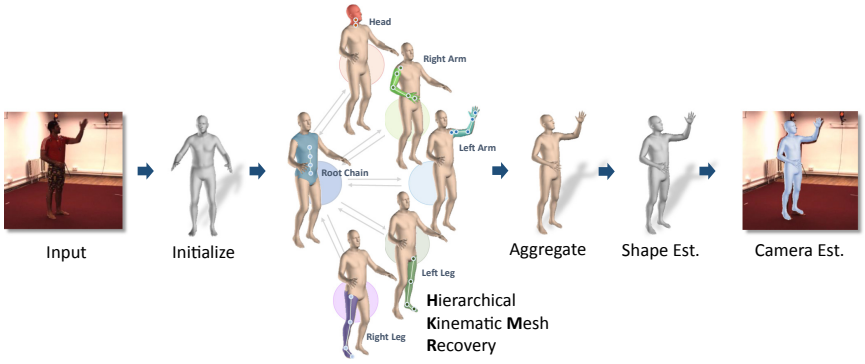


Fig. 1. We present HKMR, a new structure-informed design that hierarchically regresses the pose and shape parameters of the human body mesh.

A recent method [6] extends this to the *encoder-regressor-optimizer* framework by introducing an in-the-loop optimization step [3] which uses the output of the regressor as the starting point to iteratively optimize and generate more accurate model estimates. The regressor forms the core of both of these approaches and is typically realized with a block of fully-connected layers with non-linear activation units, taking feature vectors as input and producing the shape and rotation/pose parameter vectors as output.

However, it has been shown [7] that direct regression of rotation parameters is a very challenging task. The difficulty is exacerbated in our case of human joints due to multiple rotations and their dependencies, as noted in prior work [5,8]. Kendall *et al.* [7] further notes such regression tasks can be made more amenable, with significant performance improvements, by grounding regressor design considerations in geometry which in our context is the underlying structure of the model we are attempting to fit. However, existing *encoder-regressor* methods do not include such structural information in their design, leaving much room for performance improvement. This is even more pronounced in situations involving data corruptions (*e.g.*, occlusions), where intuitively structural information (*e.g.*, one joint dependent on or connected to another joint) can readily help infer these model parameters even when one or more joints are occluded.

To this end, we present a new architecture, called **HKMR**, and an associated hierarchical optimization technique, for 3D human mesh recovery. While Kolotouros *et al.* [8] avoids direct regression of model parameters and instead estimates the 3D mesh vertices, we investigate a more model-structure-informed design that exploits the strengths of such a representation. We use the popular SMPL model [9], which is based on the standard skeletal rig with a well-known hierarchical structure of chains with interdependent joints. Note however that HKMR can be used with other hierarchical human model instantiations as well.

HKMR defines six chains following the standard skeletal rig (a root chain and five dependent child chains: head, left/right arms, left/right legs). Each chain is

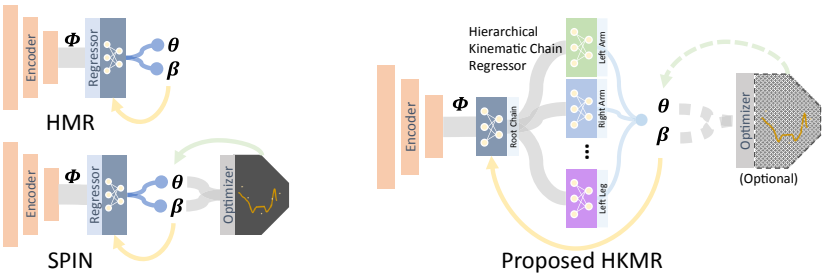


Fig. 2. HKMR can be used in either the *encoder-regressor* paradigm (HMR [5]) or the *encoder-regressor-optimizer* paradigm (SPIN [6]).

designed following the skeletal rig's kinematic model, with explicit interdependencies of joints. We repeat this for all the chains, with each non-root chain's predictions conditioned on the root chain's output, thus modeling HKMR with a set of hierarchically-nested operations (see Fig. 1). As one can note, this is in stark contrast to the existing paradigm [5] that simply operates in a *features-in-parameters-out* fashion without explicitly exploiting the underlying structure of the model. Furthermore, such a design for the regressor is particularly beneficial for parameter inference under data corruptions. By modeling hierarchical joint interdependencies, HKMR facilitates the inference of the current joint even if the previous joint is unreliable or unobserved due to occlusions or other reasons. We show HKMR leads to a new architecture for 3D human mesh estimation that substantially outperforms currently dominant baseline architectures for this problem [5,8]. Our method is flexible to be used in both the *encoder-regressor* and *encoder-regressor-optimizer* paradigms (see Fig. 2, where we show how the *optimizer* can be optionally added to our pipeline), and we demonstrate substantial performance improvements in both these cases.

To summarize, our key contributions include:

- We present HKMR, a new parameter regressor that is flexible to be used in either of the two existing *encoder-regressor* or *encoder-regressor-optimizer* paradigms for 3D human mesh recovery.
- Our key insight is the design of the parameter regressor in a way that explicitly exploits structural constraints of the human body model, resulting in a strong model-design-informed architecture.
- We empirically show HKMR improves the performance of both *encoder-regressor* and *encoder-regressor-optimizer* methods and establishes new state-of-the-art results on several human mesh recovery benchmark datasets.
- HKMR is robust to occlusions, as evidenced by substantial relative performance improvements on data under a wide variety of occlusion conditions.

2 Related Work

There is a large body of work on human pose estimation, formulating the problem as one of predicting 2D keypoints [10–12], estimating 3D joints [13–15], or model-based parametric human body estimation [16–21]. Here, we discuss most relevant methods with particular focus on their structure-related design choices.

Flat-Regression Methods. A common paradigm has been the direct CNN-based regression of the body model parameters with focus on capturing multi-scale information [22,23] or encoding spatial relationships between keypoints [24–26]. With the increasing adoption of parametric human models such as SMPL [9], several methods shifted focus to regressing the model parameters [5,27–32]. The most representative in this line of work is of Kanazawa *et al.* [5] that learned an end-to-end image-to-SMPL-parameter regressor. Kolotouros *et al.* [6] extended this work by combining the initial regressor estimates with an in-the-loop SMPLify [3] optimization step, reporting the most competitive performance on benchmark datasets to date. However, these methods tend to ignore the inherent structure of the body model in their regressor design.

Structure-Aware Methods. Several methods have sought to include structural priors in their design. Earlier works considered pictorial structures [33,34] where the human body was modeled as a set of rigid templates with pairwise potentials. More recently, Cai *et al.* [35] and Kolotouros *et al.* [8] exploited variants of graph CNNs to encode the 2D keypoint or mesh structure. Ci *et al.* [36] went a step further and proposed a generic framework for graph CNNs, introducing a specific formulation focusing on local joint neighborhoods. Aksan *et al.* [37] explicitly modeled the joint dependencies by means of a structured prediction layer. Tang *et al.* [38] investigated the relationship between different parts of the human body to learn part-specific features for 2D keypoint prediction. Fang *et al.* [39] encoded human body dependencies by employing Bi-directional RNNs during 3D joint prediction, while Isack *et al.* [40] attempted to learn priors between keypoints by following a pre-specified prediction order and connectivity. In contrast to these approaches, our method explicitly encodes both the hierarchical structure as well as joint interdependencies through a kinematics model. Zhou *et al.* [41] also proposed kinematic modeling but is substantially different from our method. First, it does not model joint interdependencies in each chain, thus failing to take full advantage of the kinematic formulation. Next, it models all joints as part of one large chain, therefore not exploiting the skeletal rig's hierarchical nature. Finally, it only generates 3D joints and not the full mesh.

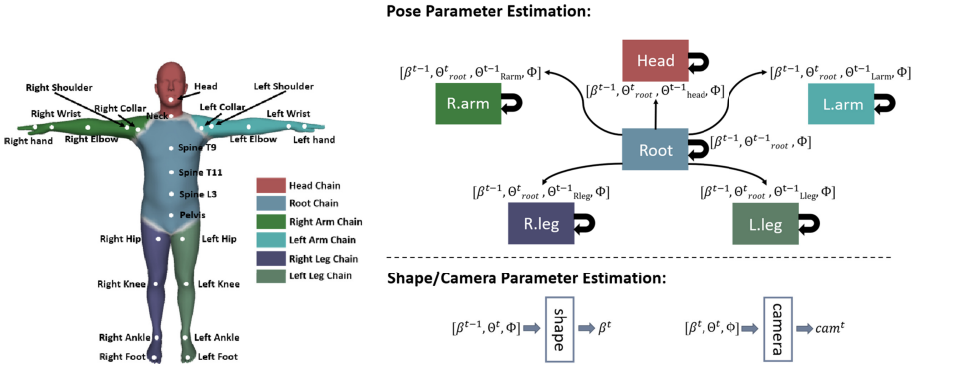
3 Approach

As noted in Sects. 1 and 2, existing mesh recovery methods formulate the problem as purely one of regressing the real-valued parameter vectors. While the network (during training) is regularized by priors learned from data (mixture models as in [3]) or even discriminator models [5], we contend this does not

fully exploit the knowledge we have about the structure of the SMPL model. From SMPL's design principles, we know it is motivated by the standard skeletal rig, and that this rig has an associated hierarchy. We argue that a parameter estimation procedure that is explicitly informed by this hierarchy constitutes a stronger integration of the structure of the SMPL model as opposed to HMR-like [5] methods that generate the parameter vectors by means of a structure-agnostic set of fully-connected units (see HMR in Fig. 2). Motivated by this intuition, we propose **HKMR**, a new architecture for the parameter regressor. As we demonstrate in Sect. 4, this leads to a new convolutional neural network architecture that substantially outperforms the currently dominant paradigm of *encoder-regressor* architectures [5,8], while also lending itself favorably applicable to *encoder-regressor-optimizer* approaches like SPIN [6].

3.1 3D Body Representation

We use the SMPL model of Loper *et al.* [9] to parameterize the 3D human mesh. SMPL is a differentiable model defined in the real-valued space of pose $\boldsymbol{\Theta} \in \mathbb{R}^{72}$ and shape $\boldsymbol{\beta} \in \mathbb{R}^{10}$ parameters. While $\boldsymbol{\Theta}$ models the relative 3D rotation of $K = 24$ joints in the axis-angle representation, $\boldsymbol{\beta}$ models the shape of the body as captured by the first 10 coefficients of a PCA projection of the shape space. SMPL defines a function $\mathcal{M}(\boldsymbol{\Theta}, \boldsymbol{\beta}) \in \mathbb{R}^{N \times 3}$ that produces the $N = 6890$ 3D vertices representing the human mesh. Starting from a template mesh, the desired body mesh is obtained by applying forward kinematics based on the joint rotations $\boldsymbol{\Theta}$ and by applying shape deformations conditioned on both $\boldsymbol{\Theta}$ and $\boldsymbol{\beta}$. Finally, the joint locations are defined as a linear combination of the mesh vertices, obtained by a linear regression function $\mathcal{X}(\mathcal{M}(\boldsymbol{\Theta}, \boldsymbol{\beta})) \in \mathbb{R}^{K \times 3}$.



(a) Chains and joints (b) Parameter estimation

Fig. 3. The kinematic chains and our hierarchical optimization workflow.

3.2 Hierarchical Kinematic Pose and Shape Estimation

Our architecture comprises an encoder to generate features (the same ResNet50 [42] to existing models [5,8]), followed by our proposed regressor that explicitly models the hierarchy between various body parts and the interdependency of the body joints within these parts. We define each body part by a standalone kinematic chain with joints at the same locations as the SMPL model (see Fig. 3(a)). As in the standard skeletal rig, we consider the chain representing the torso of the body as root, with all other chains (arms, leg, and head) hierarchically dependent on this root chain. We estimate $\boldsymbol{\Theta}$ and $\boldsymbol{\beta}$ in the same spirit as the iterative feedback of HMR [5] but is realized substantially differently. Given six chains corresponding to different parts of the body, we express $\boldsymbol{\Theta}$ as a concatenation of all the individual chain pose parameters: $\boldsymbol{\Theta} = [\boldsymbol{\theta}_{\text{root}}, \boldsymbol{\theta}_{\text{head}}, \boldsymbol{\theta}_{\text{R.arm}}, \boldsymbol{\theta}_{\text{L.arm}}, \boldsymbol{\theta}_{\text{R.leg}}, \boldsymbol{\theta}_{\text{L.leg}}]$, where each $\boldsymbol{\theta}$ represents chains' angle-axis pose parameters. For example root chain is comprised by 4 joints, $\boldsymbol{\theta}_{root}$ is of dimensionality 12. Starting from the mean pose and shape $\boldsymbol{\Theta}^0$ and $\boldsymbol{\beta}^0$, our method first estimates the next $\boldsymbol{\Theta}^t$, which is then used to update the next $\boldsymbol{\beta}^t$. This process is repeated for $t = T$ iterations resulting in final estimates $\hat{\boldsymbol{\Theta}} = [\hat{\boldsymbol{\theta}}_{\text{root}}, \hat{\boldsymbol{\theta}}_{\text{head}}, \hat{\boldsymbol{\theta}}_{\text{R.arm}}, \hat{\boldsymbol{\theta}}_{\text{L.arm}}, \hat{\boldsymbol{\theta}}_{\text{R.leg}}, \hat{\boldsymbol{\theta}}_{\text{L.leg}}]$ and $\hat{\boldsymbol{\beta}}$. This constitutes what we call the **outer iterative** process involving all the chains. Each chain c also has an **inner iterative** process estimating its pose $\hat{\boldsymbol{\theta}}_c^t$ at each outer iterative step t, *i.e.*, $\hat{\boldsymbol{\theta}}_c^t$ itself is updated for multiple iterations at the outer step t. In the following, we first describe the inner iterative process of each chain c, followed interaction between inner and outer iterative steps, leading up to the overall hierarchical process for parameter optimization.

Iterative Kinematic Chain Parameter Estimation. For simplicity of exposition, we focus on one chain here. As noted above, at each outer iterative step t, the chain c has an inner-iterative process to estimate its pose parameters $\boldsymbol{\theta}^t$. Specifically, the chain c takes as input its previous estimates at $t-1$, $\boldsymbol{\theta}^{t-1} = \boldsymbol{\theta}^I$ and $\boldsymbol{\beta}^{t-1} = \boldsymbol{\beta}^I$ and iteratively refines it yielding $\boldsymbol{\theta}^t$ at the current outer step t. In the following section we drop the superscripts $t-1$ and t for clarity.

We take inspiration from inverse kinematics, more specifically from iterative solvers for the problem. The 3D location of the chain's end-effector $e = [e_x, e_y, e_z]^T$ is related to the pose of the rigid bodies in the chain by a nonlinear function characterizing forward kinematics $\boldsymbol{e} = g(\boldsymbol{\theta})$. For inverse kinematics, we seek $\boldsymbol{\theta}$ (system configuration) that will realize it: $\boldsymbol{\theta} = g^{-1}(\boldsymbol{e})$. Considering the Taylor series expansion of g, we can characterize changes in the end-effector's current position e relative to changes in $\boldsymbol{\theta}$ in terms of the Jacobian matrix $\boldsymbol{J}(\boldsymbol{\theta})$ of partial derivatives of the system as $\Delta\boldsymbol{e} = \boldsymbol{J}\Delta\boldsymbol{\theta}$. Since we are interested in the inverse estimation (*i.e.*, how $\boldsymbol{\theta}$ changes with respect to $\boldsymbol{e}$), the pseudo-inverse of the Jacobian $\boldsymbol{J}^+$ is used to estimate the residual $\Delta\boldsymbol{\theta} = \boldsymbol{J}^+\Delta\boldsymbol{e}$, followed by the update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\Delta\boldsymbol{\theta}$, where α is a small scalar. This update is repeated in the inner-iterative process until a proximity to the goal end-effector position criterion is reached. This is the essence of iterative solvers for inverse kinematics problems frequently used for kinematic chains in robotics. With the forward kinematics model being a continuous function of $\boldsymbol{\theta}$ [41], we can incorporate this solution framework into an end-to-end learning paradigm. In other words, we

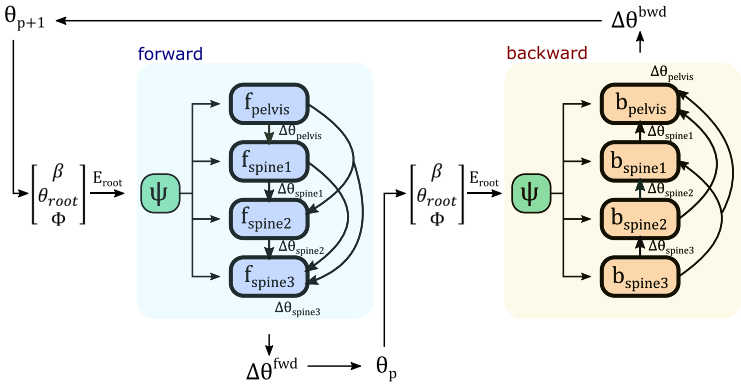


Fig. 4. The forward-backward cycle in Q_{root} that corresponds to the inner iterative procedure of each chain (here we use the root chain as example). E_{root} produces a common embedding ψ between the model parameters θ, β and image features Φ. Note θ_{root} is replaced by θ_p/θ_{p+1} at the end of the forward/backward process respectively.

design a learnable function for the chain c that predicts the residuals $\Delta\boldsymbol{\theta}$ and updates $\boldsymbol{\theta}$ iteratively with the available 3D joint annotations as supervision.

Specifically, we estimate the $\boldsymbol{\theta}$ for each kinematic chain c with a trainable neural network Q_c that takes as input values $\boldsymbol{\beta}^I$, $\boldsymbol{\theta}^I$, and image features $\boldsymbol{\Phi}$ extracted from the encoder. Given $\boldsymbol{\Phi}$, $\boldsymbol{\beta}^I$ and the chain-specific $\boldsymbol{\theta}^I$, we first learn a low-dimensional embedding $\boldsymbol{\psi} \in \mathbb{R}^d$ with Q_c's embedding module E_c. Our key insight is that the predicted angle of a certain joint affects the predictions of all the following joint angles in the chain. Any predicted angle in the chain changes the system configuration, consequently requiring the adjustment of the following angle predictions, which we do iteratively. To this end, we predict the $\Delta\boldsymbol{\theta}_{i,p}$ of the i^{th} joint ($i = 0, 1, \cdots, K_c - 1$, K_c: number of joints in c) starting from $i = 0$ at the inner iteration step p through a *forward* pass of the kinematic chain. In this process, we concatenate $\boldsymbol{\psi}$ and the estimated residuals of all previous joints in the following chain sequence:

$$\Delta\boldsymbol{\theta}_{i,p}^{\text{fwd}} = f_i([\boldsymbol{\psi}, \Delta\boldsymbol{\theta}_{i-1,p}, \Delta\boldsymbol{\theta}_{i-2,p}, \cdots, \Delta\boldsymbol{\theta}_{0,p}]). \tag{1}$$

where the function f_i is realized as one fully-connected layer for the joint i that outputs three real-valued numbers. See Fig. 4 for a visual summary. For the first joint $i = 0$ (`pelvis`) in the chain, we use $[\boldsymbol{\psi}, 0]$ as input to f_i. In other words, the second joint's prediction is dependent on the first (`spine1` depends on `pelvis`), the third is dependent on both first and second (`spine2` depends on both `spine1` and `pelvis`), and so on. When all residuals are predicted, the current estimate (at inner iteration step p) for the pose of the chain $\boldsymbol{\theta}_p = [\boldsymbol{\theta}_{0,p}, \cdots, \boldsymbol{\theta}_{K_c-1,p}]$ is updated as $\boldsymbol{\theta}_{i,p} = \boldsymbol{\theta}_{i,p-1} + \Delta\boldsymbol{\theta}_{i,p}^{\text{fwd}}$; the embedding $\boldsymbol{\psi}$ is then updated using E_c based on this new $\boldsymbol{\theta}_p$. Specifically, as above, this step takes $\boldsymbol{\Phi}$, $\boldsymbol{\beta}^I$ and the updated $\boldsymbol{\theta}_p$ as input, producing an updated $\boldsymbol{\psi}$. Since these joint angles can be affected by both the next and previous joints, after the forward update as above,

we additionally perform a *backward* pass:

$$\Delta\boldsymbol{\theta}^{\text{bwd}}_{i,p+1} = b_i([\boldsymbol{\psi}, \Delta\boldsymbol{\theta}_{i-1,p+1}, \Delta\boldsymbol{\theta}_{i-2,p+1}, \cdots, \Delta\boldsymbol{\theta}_{0,p+1}]) \tag{2}$$

where b_i is defined similarly to f_i above. Note that the notation p is used to differentiate between the forward and backward pass (i.e. θ_p referred to the estimated pose parameters after the forward pass, and θ_{p+1} to the corresponding backward). The backward update takes the updated embedding $\boldsymbol{\psi}$ from the forward update as the input and starts from the last joint $i = K_c - 1$ in the forward update (`spine3`, see backward in Fig. 4). Specifically, in predicting the $\Delta\boldsymbol{\theta}$ for `spine3`, the input to its corresponding b_i is $[\boldsymbol{\psi}, 0]$, as with the initial step in the forward update as above. Subsequently, every other joint prediction depends on all the preceding predictions (*e.g.*, `spine2` depends on `spine3`, `spine1` depends on both `spine2` and `spine3`, and so on; see backward in Fig. 4). Once the backward residuals are computed, the current estimate for the pose of the chain $\boldsymbol{\theta}_{p+1} = [\boldsymbol{\theta}_{0,p+1}, \cdots, \boldsymbol{\theta}_{K_c-1,p+1}]$ at inner iteration step $p+1$ (the forward pass above is step p) is updated as $\boldsymbol{\theta}_{i,p+1} = \boldsymbol{\theta}_{i,p} + \Delta\boldsymbol{\theta}^{\text{bwd}}_{i,p+1}$. Again, as above, given this updated $\boldsymbol{\theta}_{p+1}$, we update $\boldsymbol{\psi}$ using E_c. This forward-backward cycle (forward is inner iterative step p, backward is inner iterative step $p+1$) is repeated for multiple iterations. Following the inverse kinematics formulation, we seek to optimize the prediction of $\boldsymbol{\theta}$ so we can reach the position of the chain's end-effector e given the desired pose. This constitutes defining an e in the chain and using an L_1-like distance function to minimize the distance between e and its ground truth. In practice, since inverse kinematics can produce multiple solutions, we apply this loss on all joints in the chain, as we show next.

Hierarchical Optimization. As noted above, we denote the torso kinematic chain as root, and the others as its n_c dependent/children chains. The purpose of having this hierarchy is to allow the pose predicted by the root chain to affect how the rest of the chains operate. This is achieved by using the prediction of the root chain's $\boldsymbol{\theta}$ as part of the input to all other chains' neural networks. Specifically, while the root chain network takes $\boldsymbol{\beta}^{t-1}$, $\boldsymbol{\theta}^{t-1}_{\text{root}}$, and Φ as input in predicting the next $\boldsymbol{\theta}^{t}_{\text{root}}$, all other chains' networks take the previous $\boldsymbol{\beta}^{t-1}$, $\boldsymbol{\theta}^{t-1}_{\text{c}}$, Φ, and the current $\boldsymbol{\theta}^{t}_{\text{root}}$ as input in predicting the next $\boldsymbol{\theta}^{t}_{\text{c}}$. This is also visually summarized in Fig. 3(b). Since the kinematic chains operate only on the pose parameters, the shape parameters $\boldsymbol{\beta}$ remain constant during this process. We re-estimate $\boldsymbol{\beta}$ after every $\boldsymbol{\Theta}$ prediction cycle ends (*i.e.*, after each of the six chains have completed their inner-iterative estimation process). To this end, we define a shape-estimation neural network that takes the previous outer iteration step's shape prediction $\hat{\boldsymbol{\beta}}^{t-1}$, the current outer iteration's pose prediction $\hat{\boldsymbol{\Theta}}^{t}$, and the features Φ as input and produces the current outer iteration step's shape estimate $\hat{\boldsymbol{\beta}}^{t}$. This updated $\hat{\boldsymbol{\beta}}^{t}$ (along with $\hat{\boldsymbol{\Theta}}^{t}$) will then be used to initialize the next (outer iteration step $t+1$) $\boldsymbol{\Theta}$ prediction cycle. We supervise the prediction of both pose and shape parameters by applying an L_1 loss between the predicted 3D joint locations $\hat{\boldsymbol{X}}^{t} = \mathcal{X}(\mathcal{M}(\hat{\boldsymbol{\Theta}}^{t}, \hat{\boldsymbol{\beta}}^{t}))$ of the chain and their respective ground-truth: $L^{t}_{\text{3D}} = \sum_{i=1}^{N_{\text{3d}}} ||\hat{\boldsymbol{X}}^{t}_{i} - \boldsymbol{X}_i||_1$, where the subscript i represents the i^{th} joint and N_{3d} is the number of available annotated 3D joints. Note that if SMPL

parameter ground-truth is available, we can also use this to directly supervise the pose and shape parameters: $L^t_{\text{smpl}} = ||[\hat{\boldsymbol{\Theta}}^t, \hat{\boldsymbol{\beta}}^t] - [\boldsymbol{\Theta}, \boldsymbol{\beta}]||_2^2$.

Camera Parameter Estimation. In order to fully utilize the 2D joints annotations available in most datasets, we define a camera-estimation network that takes the predicted parameters $\hat{\boldsymbol{\Theta}}^t$, $\hat{\boldsymbol{\beta}}^t$ at the outer step t, and image features $\boldsymbol{\Phi}$ to estimate the camera parameters that model a weak-perspective projection as in HMR [5], giving translation $\boldsymbol{\rho}^t \in \mathbb{R}^2$ and scale $s^t \in \mathbb{R}$. Consequently, 2D joints $\hat{\boldsymbol{x}}$ can be derived from the 3D joints $\hat{\boldsymbol{X}}^t$ as $\hat{\boldsymbol{x}}^t = s\Pi(\hat{\boldsymbol{X}}^t) + \boldsymbol{\rho}^t$, where Π is an orthographic projection. We then supervise the estimated 2D joints with an L1 loss: $L^t_{\text{2D}} = \sum_{i=1}^{N_{\text{2d}}} ||\hat{\boldsymbol{x}}^t_i - \boldsymbol{x}_i||_1$, where the subscript i represents the i^{th} joint and N_{2d} is the number of available annotated 2D joints.

Pose Prior. To ensure realism, we follow Pavlakos *et al.* [43] and train a VAE to learn a distribution over plausible human poses. The VAE encodes the 69-D $\boldsymbol{\Theta}$ (corresponding to 23 joints excluding the global orientation) to the latent vector $\boldsymbol{Z}_{\boldsymbol{\Theta}}$, which is then used, via re-parameterization [44], by the decoder to reconstruct $\boldsymbol{\Theta}$. We use the MoSh dataset [45], comprising about 6 million synthetic human poses and shapes, to train our VAE with the standard learning objective. Once trained, we discard the decoder and only use the encoder to ensure the pose predicted by our regressor is physically plausible. To this end, we use this encoder to compute the latent representation $\boldsymbol{Z}_{\hat{\boldsymbol{\Theta}}^t}$ of the predicted $\hat{\boldsymbol{\Theta}}^t$ at the current outer iteration step t. We then enforce $\boldsymbol{Z}_{\hat{\boldsymbol{\Theta}}^t}$ to follow the same unit normal distribution used to train the encoder, which is realized with the KL divergence loss at outer iteration step t: $L^t_{\text{KL}} = KL(\boldsymbol{Z}_{\hat{\boldsymbol{\Theta}}^t}||\mathcal{N}(\mathbf{0}, \mathcal{I}))$.

3.3 Overall Learning Objective

During training, we add up all the losses over the T outer iterations (λs are the corresponding loss weights):

$$L = \sum_{t=1}^{T} \lambda_{\text{smpl}} L^t_{\text{smpl}} + \lambda_{\text{3D}} L^t_{\text{3D}} + \lambda_{\text{2D}} L^t_{\text{2D}} + \lambda_{\text{KL}} L^t_{\text{KL}}, \tag{3}$$

With Eq. 3, we perform a single backward pass during which the individual chain models, the shape model, and the camera model are optimized jointly.

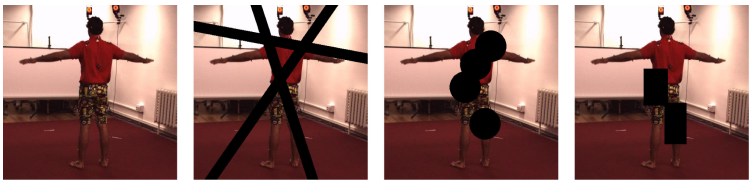
3.4 In-the-Loop Optimization

SPIN [6] introduced the *encoder-regressor-optimizer* paradigm with an in-the-loop SMPLify [3] optimization step. Like *encoder-regressor* above, our method can be used in this framework as well. Since SPIN [6] used an HMR-inspired [5] regressor, our proposed regressor can be used as a direct drop-in replacement, and we show in Sect. 4 this results in performance improvements.

4 Experiments and Results

Datasets. Following HMR [5], we use the training splits of LSP [46], LSP-extended [47], MPII [48], MS COCO [49], Human3.6M [50] and MPI-INF-3DHP [51] for network training. We use the same encoder as HMR [5] (ResNet50) and exactly follow their experimental setup, reporting results for both protocols P1 and P2 of Human3.6M (P1 uses videos of all cameras whereas P2 uses only the frontal one - camera 3). We use the mean per joint position error (MPJPE) without any Procrustes post-processing as our evaluation metric. Additional implementation information results/discussion are in the supplementary material.

Occlusions. In order to demonstrate robustness to occlusions, given an image, we generate multiple synthetically occluded images. Specifically, following Sarandi *et al.* [52], we use three occlusion patterns (oriented bars, circles, rectangles) and generate three sets of data, one under each such occlusion pattern (see Fig. 5 for an illustration). Note that these synthetic occlusions are used only at test time, not during model training.



(a) Original (b) Oriented bar (c) Circle (d) Rectangle

Fig. 5. Examples of occlusion patterns.

Baseline Architecture Evaluation. We first evaluate the efficacy of our proposed regressor design. In Table 1, we compare HKMR's to the currently dominant methods in the encoder-regressor paradigm- HMR [5] and CMR [8]. Note that we use exactly the same encoder (ResNet50) as these methods, so we essentially compare our proposed regressor with their regressors. Here, "Standard" refers to the standard/existing Human3.6M test set (same as in HMR [5]), whereas each of the last three columns refers to the Human3.6M test set with the corresponding synthetically generated occlusion pattern. We make several observations from these results. First, on the standard Human3.6M test set, our proposed method outperforms both HMR [5] and CMR [8] for both P1 and P2 protocols. For example, the MPJPE of our method for P1 is 71.08 mm, almost 17 mm lower than that of HMR. Similarly, our MPJPE is almost 4 mm lower than CMR in both P1 and P2. Next, these trends hold even under all the three types of occlusions. For example, for "Bar", our MPJPE is approximately 21 mm and 24 mm lower than HMR for P1 and P2. Finally, our method takes about

Table 1. MPJPE (lower the better) baseline architecture evaluation on Human3.6M.

	#Param	Standard		Bar		Circle		Rectangle	
		P1	P2	P1	P2	P1	P2	P1	P2
HMR [5]	26.8M	87.97	88.00	98.74	98.54	95.28	91.71	100.23	99.61
CMR [8]	42.7M	74.70	71.90	82.99	78.85	83.50	79.24	89.01	84.73
HKMR	26.2M	**71.08**	**67.74**	**78.34**	**74.91**	**77.60**	**71.38**	**81.33**	**76.79**

0.044 s per image, whereas the corresponding numbers for CMR and HMR are 0.062 s and 0.009 s on a Tesla V100 GPU.

Robustness to Degree of Occlusions. We next evaluate the robustness of our regressor as we vary the degree of occlusion in the data. For "Bar", we do this by increasing the width of the bar uniformly from 10 pixels (degree of occlusion or DoC of 1) to 50 pixels (degree of occlusion or DoC of 5). For "Circle", we increase the radius of the circle from 10 pixels (DoC 1) to 50 pixels (DoC 5). Finally, for "Rectangle", we increase the area of the rectangle from 3,000 pixels (DoC 1) to 15,000 pixels (DoC 5). In each case, we perform the occlusion on one joint at a time and compute the average MPJPE over all the joints. The results of this experiment are shown in Fig. 6, where the first row shows the average MPJPE for protocol P1, whereas the second row shows this number for P2. As expected, as the DoC increases, the average MPJPE increases for all three methods. However, this increase is lower for our proposed method, resulting in generally lower average MPJPE values when compared to both HMR and CMR.

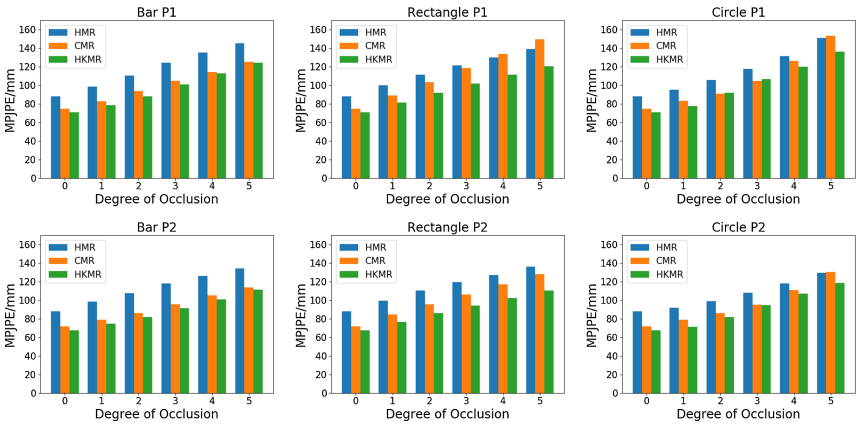


Fig. 6. Robustness to occlusions: HMR, CMR, and our proposed method.

Ablation Study. We next conduct an ablation study to understand the impact of various design considerations in our proposed regressor. The results are shown

Table 2. MPJPE (lower the better) ablation results on Human3.6M.

	No joint hierarchy	Forward only	Discriminator	Full model
P1	77.10	75.99	74.21	**71.08**
P2	74.28	72.10	71.72	**67.74**

in Table 2. "No joint hierarchy" indicates the scenario in our regressor design where each joint prediction is dependent on just the immediately preceding joint, not all the previous joints. "Forward only" corresponds to conducting only the *forward* pass in the inner-iterative process, not the full forward-backward cycle. "Discriminator" indicates our full model but using a discriminator [5] instead of our VAE for the pose prior. "Full model" indicates our proposed HKMR. As can be seen from the MPJPE values for both P1 and P2, modeling all the joint interdependencies hierarchically as in HKMR is important, with an MPJPE gain of almost 6 mm for both protocols. Similarly, there is a 5 mm gain when modeling both forward and backward joint dependence, and finally, our full HKMR model with the VAE performs better than the discriminator, resulting in a 3–4 mm MPJPE gain across both protocols. Finally, we also conduct experiments with a varying number of inner and outer iterations (see Table 3). One can note that while increasing the number of outer or inner iterations reduces MPJPE, too many iterations actually leads to diminishing returns. In our experiments, we found 3 outer and 4 inner iterations worked best.

Table 3. MPJPE (lower the better) ablation results with varying numbers of outer (left) and inner (right) iterations on Human3.6M.

#Outer Iter	1	2	3	4
P1	93.79	80.96	**71.08**	73.16
P2	88.61	76.41	**67.74**	69.40

#Inner Iter	2	4	6
P1	83.34	**71.08**	73.61
P2	82.94	**67.74**	69.55

In-the-Loop Optimization Evaluation. As discussed in Sect. 3.4, the in-the-loop optimization techniques follow the encoder-regressor-optimizer paradigm, with SPIN [6] the first method to be proposed under this framework. SPIN used the same regressor design as HMR [5] for predicting the model parameters. Clearly, our proposed method is applicable to this scheme as a simple drop-in replacement of the HMR regressor with our proposed regressor (HKMR$_{MF}$). We present the results of this experiment in Table 4. As can be noted from the results, our proposed regressor gives lower MPJPE compared to SPIN [6] for both P1 and P2. In fact, our MPJPE result of 64.02 mm and 59.62 mm establishes a new state-of-the-art on Human3.6M. Furthermore, we observe similar trends under the three different kinds of occlusions as well. Qualitative results in Fig. 7

Table 4. Model fitting in-the-loop evaluation on Human3.6M and MPII invisible joints.

Human3.6M	Standard		Bar		Circle		Rectangle	
MPJPE (mm)↓	P1	P2	P1	P2	P1	P2	P1	P2
SPIN [6]	65.60	62.23	74.40	68.61	74.06	67.03	77.21	70.35
HKMR$_{MF}$	**64.02**	**59.62**	**70.10**	**64.91**	**69.60**	**63.22**	**70.10**	**64.91**

MPII	MPJPE	PCK
Invisible	(pixel)↓	(%)↑
SPIN [6]	59.52	62.16
HKMR$_{MF}$	**55.56**	**66.24**

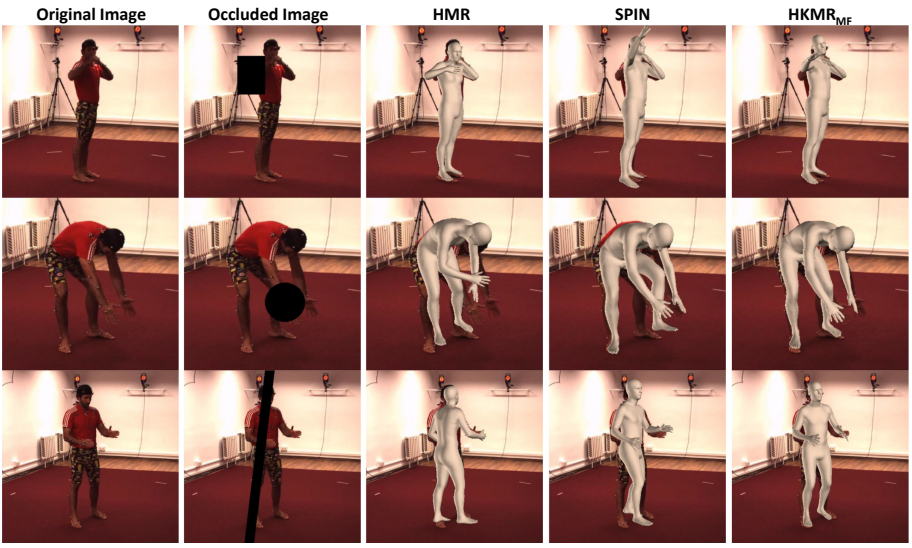


Fig. 7. Our proposed method results in more accurate fits: right hand in first row, left leg in second row, and root orientation in third row.

reinforce this point. For instance, our method results in a better fit in the right hand region (first row) and left leg region (second row) even under occlusions. To further demonstrate the efficacy of our method under occlusions, we also report results on the MPII invisible joints validation dataset [48] in Table 4 (right), where we see substantial gains compared to SPIN for both MPJPE and PCK. Figure 8 shows some qualitative results on this data where we see obvious differences - our method results in substantially better model fits even in challenging cases such as those in columns 1–3.

Comparison with the State of the Art. Finally, we compare the results of our method with competing state-of-the-art methods on the Human3.6M and LSP test set (see Table 5). Note that we follow the same protocol as Kolotouros [6] in the evaluation on the LSP dataset. From Table 5, our proposed method gives the highest accuracy and F1-score on the LSP dataset and the lowest MPJPE for both protocols on the Human3.6M dataset, thereby establishing new state-of-the-art results on both these datasets. To further put these numbers in perspective, our method substantially outperforms even competing

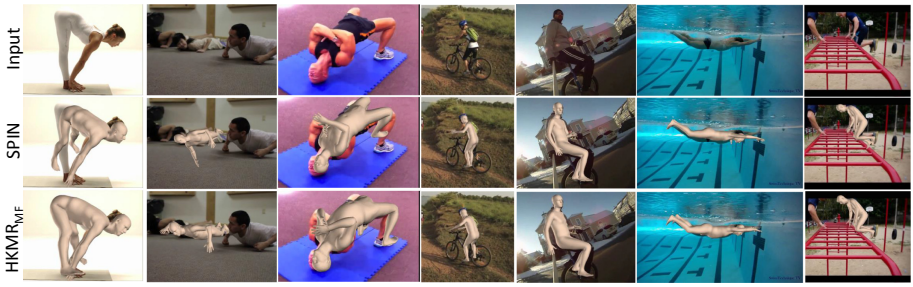


Fig. 8. Qualitative results on the invisible parts in the MPII validation set.

Table 5. Segmentation (% ↑) and MPJPE (mm ↓) on LSP and Human3.6M ("-" indicates result is not reported/unavailable in original papers).

LSP	FB Seg.		Part Seg.	
	acc.	f1	acc.	f1
Oracle [3]	92.17	**0.88**	88.82	0.67
SMPLify [3]	91.89	**0.88**	87.71	0.64
SMPLify+[28]	92.17	**0.88**	88.24	0.64
HMR [5]	91.67	0.87	87.12	0.60
CMR [8]	91.46	0.87	88.69	0.66
TexturePose [21]	91.82	0.87	89.00	0.67
SPIN [6]	91.83	0.87	89.41	0.68
HKMR$_{MF}$	**92.23**	**0.88**	**89.59**	**0.69**

Human3.6M	P1	P2
HMR [5]	87.97	88.00
Arnab *et al.* [20]	-	77.80
HoloPose [16]	-	64.28
CMR [8]	74.70	71.90
DaNet [17]	-	61.50
DenseRaC [18]	76.80	-
VIBE [19]	-	65.60
SPIN [6]	65.60	62.23
HKMR$_{MF}$	**64.02**	**59.62**

methods that use extra information, *e.g.*, Arnab *et al.* [20] that uses temporal image sequences as opposed to just one single frame, or DenseRaC [18] that uses UV maps as an additional source of supervision. Finally, even compared to the recently published work of Kocabas *et al.* [19], our proposed method gives almost 6 mm lower MPJPE. In fact, even Kocabas *et al.* [19] uses extra temporal information as opposed to just one single frame in our case. The strong performance of our method despite these seemingly disadvantageous factors further substantiates the motivation and competitiveness of our technique. As discussed previously, note that all numbers in Table 5 are standard MPJPE values without any Procrustes (rigid transformation) post-processing.

5 Summary

We presented a new architecture for regressing the pose and shape parameters of a parametric human mesh model that is explicitly informed by the structural knowledge of the model being fit. The proposed new design is quite flexible, which we demonstrated by means of applicability to both the popular encoder-regressor and encoder-regressor-optimizer paradigms for 3D human mesh estimation. By means of extensive experiments on standard benchmark datasets, we

demonstrated the efficacy of our proposed new design, establishing new state-of-the-art performance and improved robustness under a wide variety of data occlusions.

References

1. Singh, V., et al.: DARWIN: deformable patient avatar representation with deep image network. In: Descoteaux, M., Maier-Hein, L., Franz, A., Jannin, P., Collins, D.L., Duchesne, S. (eds.) MICCAI 2017. LNCS, vol. 10434, pp. 497–504. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66185-8_56
2. Martínez-González, A., Villamizar, M., Canévet, O., Odobez, J.-M.: Real-time convolutional networks for depth-based human pose estimation. In: IROS. IEEE (2018)
3. 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: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision - ECCV 2016. LNCS, vol. 9905. Springer, Cham (2016)
4. Tung, H.-Y., Tung, H.-W., Yumer, E., Fragkiadaki, K.: Self-supervised learning of motion capture. In: NIPS. Curran Associates Inc. (2017)
5. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: CVPR. IEEE (2018)
6. Kolotouros, N., Pavlakos, G., Black, M.J., Daniilidis, K.: Learning to reconstruct 3D human pose and shape via model-fitting in the loop. In: ICCV. IEEE (2019)
7. Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: CVPR. IEEE (2017)
8. Kolotouros, N., Pavlakos, G., Daniilidis, K.: Convolutional mesh regression for single-image human shape reconstruction. In: CVPR. IEEE (2019)
9. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: a skinned multi-person linear model. ACM TOG **34**(6), 1–16 (2015)
10. Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., Sheikh, Y.: OpenPose: real-time multi-person 2D pose estimation using part affinity fields. arXiv preprint arXiv:1812.08008, 2018
11. Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J.: Human pose estimation with iterative error feedback. In: CVPR. IEEE (2016)
12. Güler, R.A., Neverova, N., Kokkinos, I.: Dense human pose estimation in the wild. In: CVPR. IEEE, Densepose (2018)
13. Mehta, D., et al.: VNect: real-time 3D human pose estimation with a single RGB camera. ACM TOG **36**(4), 1–14 (2017)
14. Moreno-Noguer, F.: 3D human pose estimation from a single image via distance matrix regression. In: CVPR. IEEE (2017)
15. Pavlakos, G., Zhou, X., Daniilidis, K.: Ordinal depth supervision for 3D human pose estimation. In: CVPR. IEEE (2018)
16. Guler, R.A., Kokkinos, I.: HoloPose: holistic 3D human reconstruction in-the-wild. In: CVPR. IEEE (2019)
17. Zhang, H., Cao, J., Guo, L., Ouyang, W., Sun, Z.: DaNet: decompose-and-aggregate network for 3D human shape and pose estimation. In: ACM MM. ACM (2019)
18. Xu, Y., Zhu, S.-C., Tung, T.: DenseRaC: joint 3D pose and shape estimation by dense render-and-compare. In: ICCV. IEEE (2019)

19. Kocabas, M., Athanasiou, N., Black, M.J.: VIBE: video inference for human body pose and shape estimation. In: CVPR. IEEE (2020)
20. Arnab, A., Doersch, C., Zisserman, A.: Exploiting temporal context for 3D human pose estimation in the wild. In: CVPR. IEEE (2019)
21. Pavlakos, G., Kolotouros, N., Daniilidis, K.: TexturePose: supervising human mesh estimation with texture consistency. In: ICCV. IEEE (2019)
22. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
23. Pavlakos, G., Zhou, X., Derpanis, K.G., Daniilidis, K.: Coarse-to-fine volumetric prediction for single-image 3d human pose. In: CVPR. IEEE (2017)
24. Wei, S.-E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR. IEEE (2016)
25. Toshev, A., Szegedy, C.: DeepPose: human pose estimation via deep neural networks. In: CVPR. IEEE (2014)
26. Martinez, J., Hossain, R., Romero, J., Little, J.J.: A simple yet effective baseline for 3D human pose estimation. In: ICCV. IEEE (2017)
27. Tan, V., Budvytis, I., Cipolla, R.: Indirect deep structured learning for 3D human body shape and pose prediction. In: BMVC. BMVA Press (2017)
28. Pavlakos, G., Zhu, L., Zhou, X., Daniilidis, K.: Learning to estimate 3D human pose and shape from a single color image. In: CVPR. IEEE (2018)
29. 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: 3DV. IEEE (2018)
30. Yao, P., Fang, Z., Wu, F., Feng, Y., Li., J.: DenseBody: directly regressing dense 3D human pose and shape from a single color image. arXiv preprint arXiv:1903.10153 (2019)
31. Varol, G., et al.: BodyNet: volumetric inference of 3D human body shapes. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 20–38. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_2
32. 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: CVPR. IEEE (2017)
33. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV **61**(1), 55–79 (2005)
34. Yang, Y., Ramanan, D.: Articulated human detection with flexible mixtures of parts. IEEE T-PAMI **35**(12), 2878–2890 (2012)
35. Cai, Y., et al.: Exploiting spatial-temporal relationships for 3D pose estimation via graph convolutional networks. In: ICCV. IEEE (2019)
36. Ci, H., Wang, C., Ma, X., Wang, Y.: Optimizing network structure for 3D human pose estimation. In: ICCV. IEEE (2019)
37. Aksan, E., Kaufmann, M., Hilliges, O.: Structured prediction helps 3D human motion modelling. In: ICCV, pp. 7144–7153. IEEE (2019)
38. Tang, W., Ying, W.: Does learning specific features for related parts help human pose estimation? In: CVPR. IEEE (2019)
39. Fang, H.-S., Xu, Y.,Wang, W., Liu, X., Zhu, S.-C.: Learning pose grammar to encode human body configuration for 3D pose estimation. In: AAAI. AAAI (2018)
40. Isack, H., et al.: RePose: learning deep kinematic priors for fast human pose estimation. arXiv preprint arXiv:2002.03933 (2020)

41. Zhou, X., Sun, X., Zhang, W., Liang, S., Wei, Y.: Deep kinematic pose regression. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 186–201. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_17
42. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. IEEE (2016)
43. Pavlakos, G., et al.: Expressive body capture: 3D hands, face, and body from a single image. In: CVPR. IEEE (2019)
44. Kingma, D.P., Welling. M.: Auto-encoding variational Bayes. In: ICLR (2014)
45. Loper, M., Mahmood, N., Black, M.J.: MoSh: motion and shape capture from sparse markers. ACM TOG **33**(6), 1–13 (2014)
46. Johnson, S., Everingham, M.: Clustered pose and nonlinear appearance models for human pose estimation. In: BMVC. BMVA Press (2010)
47. Johnson, S., Everingham, M.: Learning effective human pose estimation from inaccurate annotation. In: CVPR. IEEE (2011)
48. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: new benchmark and state of the art analysis. In: CVPR. IEEE (2014)
49. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
50. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6M: large scale datasets and predictive methods for 3D human sensing in natural environments. IEEE T-PAMI **36**(7), 1325–1339 (2013)
51. Mehta, D., et al.: Monocular 3D human pose estimation in the wild using improved CNN supervision. In: 3DV. IEEE (2017)
52. Sárándi, I., Linder, T., Arras, K.O., Leibe, B.: How robust is 3D human pose estimation to occlusion? arXiv preprint arXiv:1808.09316 (2018)