Deformable Mesh Transformer for 3D Human Mesh Recovery

Yusuke Yoshiyasu
National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Japan
yusuke-yoshiyasu@aist.go.jp

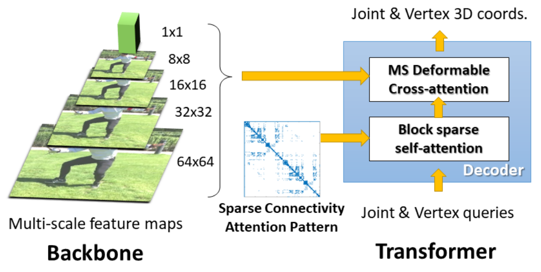
Abstract

We present **De***formable mesh trans***Former** *(DeFormer), a novel vertex-based approach to monocular 3D human mesh recovery. DeFormer iteratively fits a body mesh model to an input image via a mesh alignment feedback loop formed within a transformer decoder that is equipped with efficient body mesh driven attention modules: 1) body sparse self-attention and 2) deformable mesh cross attention. As a result, DeFormer can effectively exploit high-resolution image feature maps and a dense mesh model which were computationally expensive to deal with in previous approaches using the standard transformer attention. Experimental results show that DeFormer achieves state-of-the-art performances on the Human3.6M and 3DPW benchmarks. Ablation study is also conducted to show the effectiveness of the DeFormer model designs for leveraging multi-scale feature maps. Code is available at* https://github.com/yusukey03012/DeFormer.

1. Introduction

Human mesh recovery from a single image is an important problem in computer vision, with a wide range of applications like virtual reality, sports motion analysis and human-computer interactions. It is a challenging problem as it requires modeling of complex nonlinear mappings from an image to 3D body shape and pose.

In the past decade, remarkable progress has been accomplished in the field of 3D human mesh recovery based on convolutional neural networks (CNNs) [41]. A common way used in this field is a parametric approach that employs statistical human models parameterized by shape and pose parameters [33]. Here, CNNs features are regressed with body shape and pose parameters with approximately 100 dimensions. Then, a body mesh surface is obtained from these body parameter predictions by inputting them to the human body kinematics and statistical shape models. Leveraging multi-scale image feature maps and iteratively refining the body parameters based on global and local spatial contexts in an image, the pyramidal mesh alignment feedbacks (Py-


1x1
8x8
16x16
32x32
64x64
Multi-scale feature maps
Backbone
Sparse Connectivity Attention Pattern
Joint & Vertex 3D coords.
MS Deformable Cross-attention
Block sparse self-attention
Decoder
Joint & Vertex queries
Transformer

Model	H3.6M		3DPW	
	MPJPE↓	PA-MPJPE↓	MPJPE↓	PA-MPJPE↓
Parametric + multi-scale feat	54.2	37.2	74.2	45.3
Vertex Transformer + single-scale feat	51.2	34.5	74.7	45.6
Vertex Transformer + multi-scale feat	**44.8**	**31.6**	**72.9**	**44.3**

Figure 1. Summary of this work. We propose DeFormer, a memory efficient decoder-only transformer architecture for 3D human mesh recovery based on block sparse self-attention [16, 40, 51] and multi-scale (MS) deformable cross attention [56]. Leveraging MS feature maps efficiently in transformer, our method outperforms previous parametric approaches using MS feature maps [52, 53] and vertex transformer approaches using a single-scale feature [28, 29]. Further, by learning sparse self-attention patterns based on body mesh and skeleton connectivity, DeFormer can recover a dense mesh with 6.9K joint/vertex queries at interactive rates. The units of the MPJPE and PA-MPJPE scores are in millimeters. The lower is better.

MAF) model [53] produces a 3D mesh recovery result well-aligned to an input image.

Another paradigm for human mesh recovery is a vertex-based approach that directly regresses 3D vertex coordinates [14, 15, 26, 28, 29, 35]. Recently, transformer architectures have been applied to vertex-based human mesh recovery and show good reconstruction performances by capturing long-range interactions between body parts via self-attentions. Furthermore, as a vertex-based approach, they naturally possess potential in learning fine-grained effects like facial expressions, finger poses and clothing non-rigid deformations, if such supervisions are given. Despite their promising performances, the main challenge of the current transformer-based approaches based on the standard transformer attention is its memory cost, which limits the use of high-resolution image feature maps and a dense body mesh model within it for better 3D reconstruction. In fact, the current methods are limited to managing a single-scale coarse feature map with 7×7 grids and a coarse mesh with around 400 vertices [28, 29].

In this paper, we propose a novel vertex-based transformer approach to 3D human mesh recovery, named **De**formable mesh trans**Former** (DeFormer). DeFormer iteratively fits a human mesh model to an image using the mesh-alignment feedback loop formed in a transformer decoder. In order to leverage multi-scale feature maps and a dense mesh model in the human mesh recovery transformer model, we design a decoder network architecture using block sparse attention and multi-scale (MS) deformable attention [56] as illustrated in Fig. 1. The block sparse self-attention module exploits the sparse connectivity patterns established from a human body mesh and its skeleton. This sparsifies self-attention access patterns and reduces memory consumption. The MS deformable cross attention module aggregates multi-scale image features by attending only to a small set of key sampling points around the mesh vertex of the current reconstruction. DeFormer therefore can attend to visual feature maps in both coarse and fine levels, which contain not only global contexts but also local spatial information. It can then extract useful contextualized features for human mesh recovery and regress corrective displacements to refine the estimated mesh shape. Consequently, DeFormer can exploit high-resolution image feature maps and a dense mesh model that are not accessible to previous approaches based on the standard transformer attention [14, 28, 29] due to time/memory computational demands. As a result, DeFormer achieves better performance than previous approaches in 3D human mesh recovery.

The main contributions of this paper include:

- **DeFormer**: A novel decoder-only transformer architecture for monocular 3D human mesh recovery based on the new body-mesh-driven attention modules that leverage multi-scale visual features and a dense mesh reasonably efficiently. DeFormer achieves new SOTA results on the Human3.6M and 3DPW benchmarks.
- **Body Sparse Self-Attention** that exploits the sparse connectivity patterns extracted from a human body mesh model and its skeleton to restrict self-attention access patterns, improving memory efficiency.
- **Deformable Mesh cross Attention (DMA)** that efficiently aggregates and exploits multi-scale image feature maps by attending only to a small set of key sampling points around the reference points obtained from the current body joint and mesh vertex reconstructions.

2. Related Work

Human mesh recovery Human mesh recovery approaches can be roughly divided into three categories from a shape representation perspective: parametric body models [10, 22, 25, 27, 53], volumetric shape models [42, 54] and vertex-based approaches [28]. Please refer to the detailed taxonomy provided in the survey paper [41].

Parametric human mesh recovery approaches employ statistical human models parameterized by shape and pose parameters [10, 19, 22, 25, 27]. Kanazawa et al. [22] proposed an end-to-end learning framework of human body and shape using the SMPL human statistical model [33] and generative adversarial networks (GANs). In [22], global feature vectors are regressed with body parameters in a feedback loop manner which minimizes discrepancy between reconstruction and an input image. PyMAF [53] introduced a visual feature pyramid into a mesh alignment feedback loop, leveraging multi-scale spatial feature maps.

Vertex-based approaches [14, 15, 15, 26, 28, 29] directly regress the 3D vertex coordinates of the deformed mesh. GraphCMR [26] uses Graph Convolutional Networks (GCN) built from the adjacency matrix of the template human mesh model to retain the topology of the mesh. Recently, transformer architectures have been applied to 3D human mesh recovery to model long-range interactions between body parts. Mesh transformer (METRO) [28] uses a transformer encoder to regress 3D vertex coordinates of a mesh from CNNs global feature vectors. Mesh graphormer [29] extends METRO by incorporating vertex-to-vertex mesh connectivity, again in the form of the adjacency matrix, using graph neural networks to further capture short-range interactions between body parts. It also incorporates coarse-scale image grid features. Concurrently, Cho et al. [14] proposed a transformer encoder-decoder architecture for human mesh recovery. They focus on improving training and inference time, by introducing mask attentions in self-attention layers based on the adjacency matrix obtained from the template mesh.

Our DeFormer extends the vertex transformer approaches [3, 5]. Unlike the encoder-only [3] or encoder-decoder [5] architectures, we design a novel decoder-only transformer to avoid memory intensive self-attentions of image features in the encoder [2], because we do not need to learn highly nonlinear mapping from low resolution image features to vertex/joint 3D coordinates. Instead, we leverage multi-scale spatial context from a HRNet variant backbone [38], exploit it through DMA and better align a body model in image space. BodySparse-SA exploits body connectivity prior knowledge and sparsifies attention patterns to improve memory efficiency. Thus, the way we exploit body connectivity is different from [1, 4, 5] that use such knowledge in GCNs or attention mask. BodySparse-SA is the first transformer approach that can deal with a full resolution SMPL mesh with 6.9K vertices.

Efficient transformer In order to address a quadratic complexity of transformer attention w.r.t sequence length, many efficient transformer architectures have been proposed [40]. In language domain, much efforts have been spent to tackle this problem with the goal of processing longer input sentences. Most methods use fixed predefined sparse attention patterns on keys [9, 13, 36, 51]. Longformer [9], External transformer construction (ETC) [6] and Bigbird [51] use attention patterns having local input tokens that only attend to the tokens in fixed radius and a small set of global tokens that attend to all tokens in the sequence. ETC proposes a global-local attention mechanism that divides a sequence into global and local tokens and constructs attention patterns by adding global tokens externally. Our work can be thought of a sparse version of ETC where the joint queries act as global ones that attend to other joints in an unrestricted manner, while attending sparsely to mesh vertices. Using graphs as has been done in graph attention networks [44, 49] could remove redundant connections but such graph-based attention approaches only consider the first order neighbors that are one-hop away. Graphomer [29] relies on the standard transformer full self-attentions.

In the image domain, Swin transformer [32] is probably the most famous one based on shifted window based self-attention. Deformable DETR [56] introduced an efficient attention mechanism using learnable attention patterns that only attend to a small set of key sampling locations around a reference. In the 3D domain, ShapeFormer [48] introduced a compact 3D representation called vector quantized deep implicit function (VQDIF) that utilizes spatial sparsity.

3. Background

We first revisit the attention modules which are the key ingredients of DeFormer.

Multi-head attention module Given a query element and a set of key elements, the multi-head attention module [43] aggregates the key contents with the attention weights that measure the compatibility of query-key pairs. To focus on contents from different feature subspaces and positions, the outputs of different attention heads are linearly aggregated using learnable weights. Let $q \in \Omega_q$ and $k \in \Omega_k$ be the index of a query element and key element with their features $\mathbf{z}_q \in \mathbb{R}^C$ and $\mathbf{x}_k \in \mathbb{R}^C$, respectively. Here, Ω_q and Ω_k are a set of query and key elements, respectively. C is the feature dimension. Then, the Multi-Head Attention (MHA) is calculated as:

$$\mathrm{MHA}(\mathbf{z}_q, \mathbf{x}) = \sum_{m=1}^{M} \mathbf{w}_m [\sum_{k=\Omega_k} A_{mqk} \cdot \mathbf{w'}_m \mathbf{x}_k] \quad (1)$$

where m indexes the attention head. $\mathbf{w}_m \in \mathbb{R}^{C_h \times C}$ and $\mathbf{w'}_m \in \mathbb{R}^{C \times C_h}$ are learnable weights with C_h being C/M. The attention weights $A_{mqk} = \exp\{\frac{\mathbf{z}_q^T \mathbf{u}_m^T \mathbf{v}_m \mathbf{x}_k}{\sqrt{C_h}}\}$ is normalized to $\sum_{k\in\Omega_k} A_{mqk} = 1$ and $\mathbf{u}_m, \mathbf{v}_m \in \mathbb{R}^{C_h \times C}$ are learnable weights. To distinguish different spatial locations, $\mathbf{z}_q$ and $\mathbf{x}_k$ are usually the concatenation/summation of element contents and positional embeddings.

Multi-scale deformable attention module In order to address the issues of applying transformer attention on higher resolution feature maps, which would look over all possible spatial locations and would result in high computational cost, Zhu et al. [56] proposed a multi-scale deformable attention module. The deformable attention only attends to a small set of key sampling points around a reference point, regardless of the spatial size of the feature maps—it assigns only a fixed number of keys for each query.

Let us denote multi-scale feature maps with L levels as $\{\mathbf{x}^l\}_{l=1}^L$. Let $\hat{\mathbf{p}}_q \in [0,1]^2$ and $\mathbf{z}_q$ be the normalized coordinates of the reference point for each query q and its content feature, respectively, where the top-left and the bottom-right corners of the normalized image space being $(0,0)$ and $(1,1)$. Given $\{\mathbf{x}^l\}_{l=1}^L$ as input, the Multi-Scale Deformable Attention (MSDA) is applied as:

$$\mathrm{MSDA}(\mathbf{z}_q, \hat{\mathbf{p}}_q, \{\mathbf{x}^l\}_{l=1}^L) \quad (2)$$
$$= \sum_{m=1}^{M} \mathbf{w}_m [\sum_{l=1}^{L}\sum_{k=1}^{K} A_{mlqk} \cdot \mathbf{w'}_m \, \mathbf{x}^l(\phi_l(\hat{\mathbf{p}}_q) + \Delta\mathbf{p}_{mlqk})]$$

where m indexes the attention head, l indexes the input feature level and k indexes the sampling point. $\Delta\mathbf{p}_{mlqk}$ and A_{mlqk} are the sampling offset and attention weight at the k-th sampling point in the l-th feature level and m-th attention head. Also, the function $\phi_l(\cdot)$ un-normalizes the normalized reference point coordinates. Again, $\mathbf{w}_m \in \mathbb{R}^{C_h \times C}$ and $\mathbf{w'}_m \in \mathbb{R}^{C \times C_h}$ are learnable weights as in Eq. 1.

Block sparse attention module In order to address quadratic complexity of transformer, researchers in the language field have attempted to sparsify attention [40]. However, it is well known that sparse multiplications cannot be efficiently implemented in GPUs. Hardware accelerators like GPUs and TPUs thus work better by loading blocks of contiguous bytes at once. Thus, the works such as Bigbird define attention on blocks by packing sets of query and keys together. In order to use such a block-sparse attention module, we need to specify a sparsity layout at the block-level by dividing the original attention patterns into blocks with block sizes of 8x8, 16x16, 32x32 etc. When the tokens are arranged in 1D or 2D regular grid and when attention is local, it reduces to sliding window attentions that can be accomplished by an efficient role operation without expensive gather operations of query and key vectors.

4. Method

We propose **De**formable mesh trans**Former** (dubbed as DeFormer) for 3D human mesh recovery. DeFormer is based on the transformer decoder with efficient attention mechanisms. The overview of DeFormer is depicted in Fig. 2. DeFormer consists of 1) a backbone feature extractor, 2) a transformer decoder with body sparse self-attention layers and deformable mesh cross-attention (DMA) layers and 3) a dense mesh reconstruction module with a global scale estimator (and a mesh upsampler). The transformer decoder produces contextualized features by aggregating multi-scale feature maps extracted from a backbone model and produces the 3D coordinates of human body mesh in image space corresponding to joint and vertex queries. The dense mesh reconstruction module then optionally upsamples the transformer output and scales it to physical size. We detail each component in below.

4.1. Backbone feature extractor

Given a single RGB image, a backbone model extracts L levels of feature maps $\{\mathbf{f}^l\}_{l=1}^L$, where each level consists of $\mathbf{f}^l \in \mathbb{R}^{H^l \times W^l \times C^l}$ with $H^l \times W^l$ and C^l denoting the spatial resolution and channel dimension size of l-th feature map, respectively. Then, feature maps $\{\mathbf{f}^l\}_{l=1}^L$ are passed through 1×1 convolution and group normalization layers to obtain the input multi-scale feature maps $\{\mathbf{x}^l\}_{l=1}^L$ of the decoder. With this process, the channel dimension size of feature maps are changed from C^l to D.

4.2. Transformer decoder

The inputs to the decoder are a set of joint queries $Q_\text{J} = \{q_\text{J}^1 \dots q_\text{J}^J\}$ and vertex queries $Q_\text{V} = \{q_\text{V}^1 \dots q_\text{V}^N\}$ corresponding to the template human body mesh that consists of J joints and N vertices. These queries are fed into the decoder with three layers to produce contextualized visual features at the queries, which will be mapped to the 3D coordinates of joints $\mathbf{J} \in \mathbb{R}^{J\times 3}$ and vertices $\mathbf{V} \in \mathbb{R}^{N\times 3}$ where we denote their concatenation as $\mathbf{X} \in \mathbb{R}^{(J+N)\times 3}$. Each decoder layer consists of a body sparse self-attention and deformable mesh cross attention. Different from previous vertex-based approaches [26, 28], DeFormer removes camera projection model and regresses 3D joint/vertex coordinates in the normalized image space, $\mathbf{X}_\text{Im} \in \mathbb{R}^{(J+N)\times 3}$.

Initialization of joint and vertex queries Joint and vertex queries are encoded by two different components i.e., appearance (content) feature and position embedding. To obtain such initial features, we provide two options. The first approach learns both position embeddings and appearance features as in [56]. The second method obtains position embedding by mapping the 3D vertex coordinates of the template mesh to high dimensions with MLPs or sinusoidal functions [43]. In the second approach, the global feature vector from the backbone is repeatedly used at each query [29] to exploit image evidences from the backbone model to establish the initial appearance features.

Body Sparse Self-Attention Given query features, the self-attention layer processes them and produces contextualized features. To improve efficiency of transformer attention, we employ block sparse attention based on the sparse attention patterns built from the SMPL body model [33].

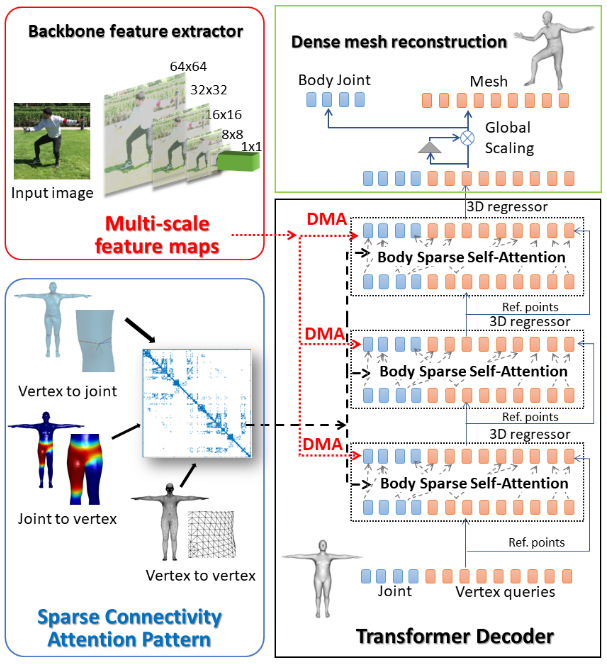


Figure 2. Overview of DeFormer. Given multi-scale visual feature maps extracted from a backbone model and joint/vertex queries, the transformer decoder that consists of body sparse self-attention and deformable mesh cross-attention (DMA) layers produce contextualized features for human body shape and pose. The body sparse self-attention layer exploits the sparsity patterns built from a human body mesh. At each DMA layer, the 3D coordinates of joint and mesh vertices in image space are reconstructed and used as reference points of deformable attention. Finally, the dense reconstruction module outputs a dense mesh with its global scale adjusted to physical scale.

We divide our sparse attention pattern into four pieces as illustrated in Fig. 3: i.e. the joint-to-joint (j2j), joint-to-vertex (j2v), vertex-to-joint (v2j) and vertex-to-vertex (v2v) pieces. To promote unrestricted non-local attentions between joints, the j2j piece have a dense pattern i.e. $\mathbf{1} \in \mathbb{R}^{J\times J}$. Attentions in the v2v piece are restricted to one-ring neighbors from each vertex. For this, an adjacency matrix $\mathcal{A} \in [0,1]^{N\times N}$ containing mesh connectivity can be simply used. For the v2j piece, we construct it from finding non-zero values of the joint regressor matrix $\mathcal{J} \in \mathbb{R}^{J\times N}$ which is used to estimate joint positions from vertex positions where we have $\bar{\mathcal{J}} \in [0,1]^{J\times N}$. For the j2v piece we construct it from a skinning weight matrix $\mathcal{W} \in \mathbb{R}^{N\times J}$. This is done by first extracting two weight matrices, one for the joints queries and the other for their parent joints. These weight matrices are summed so that the vertices will

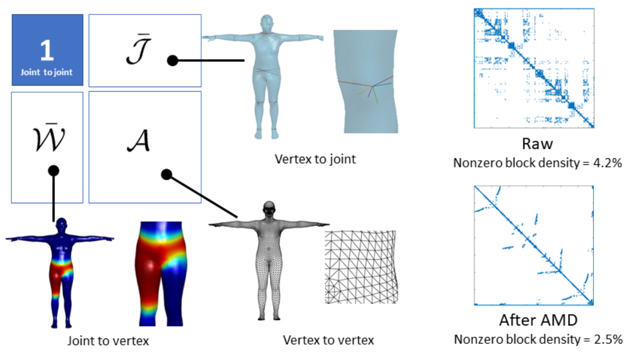


Figure 3. Sparse attention pattern built from the SMPL human body model [33], which consists of joint-to-joint (j2j), joint-to-vertex (j2v), vertex-to-joint (v2j) and vertex-to-vertex (v2v) pieces. It is established from a joint regressor matrix, skinning weight matrix and adjacency matrix. Using the approximate minimum degree (AMD) algorithm, nonzero blocks of the sparse attention pattern for 6.9K queries reduced from 4.2% to 2.5%.

be attended by the joints close to them. Then, similar to the v2j case, we find non-zero values within it and construct $\bar{\mathcal{W}} \in [0,1]^{N\times J}$. Finally, the sparse connectivity attention pattern, $\mathcal{S} \in [0,1]^{(J+N)\times(J+N)}$ is constructed by combining the pieces: $\mathcal{S} = [\,[\,\mathbf{1},\ \bar{\mathcal{J}}\,],\ [\,\bar{\mathcal{W}},\ \mathcal{A}\,]\,]$. Once $\mathcal{S}$ is constructed, we obtain a block-level sparse layout for 16×16 blocks and it is set to the block-sparse attention layer. $\mathcal{S}$ is also used for attention masks by inverting its values with logical NOT operations.

Since the template is an irregular mesh, $\mathcal{S}$ does not have an efficient structure to exploit it by GPU operations. We instead use a tool from sparse linear algebra i.e. pre-ordering techniques which are heuristic algorithms that find a permutation matrix to produce the least fill-in in the subsequent factorization. The approximate minimum degree (AMD) algorithm [7] is one of these methods that iteratively eliminates nodes with the smallest degree and is known to produce a fractal-like matrix structure with large blocks of zeros. In our case, we use AMD for making a block attention pattern further sparser. As shown in Fig. 3 right, after the AMD reordering, nonzero blocks of the sparse attention pattern for 6904 queries reduced from 4.2% to 2.5%. This means we can reduce memory consumption by $1/40$ from the full transformer self-attention. To make use of the re-ordered sparse pattern, the input query features are permuted according to the AMD permutation matrix and the output query features from the block-sparse attention layer are back permuted to the original ordering.

Deformable Mesh cross-Attention (DMA) Once contextualized features are produced by the self-attention layer, they are then passed through the DMA layer. DMA uses the current reconstructed mesh as a reference point to sample features around it to perform multi-scale deformable attention. The 3D coordinate regressor then reconstructs 3D coordinates of joints and mesh vertices corresponding to the given queries from the features produced. Our 3D regression module follows the feedback loop style approaches [22, 53] which regress corrections for pose and body shape parameters. We instead regress a 3D displacement vector for each query at each decoder layer.

Given the reconstructed joint position (or a mesh vertex position) as a reference point, MS deformable attention samples features at $K = 4$ offset points around it for $L = 4$ feature levels and $M = 8$ heads. These features are then aggregated by the MSDA function Eq. 2. The 3D coordinates for the next iteration $\mathbf{X}^{t+1}_{\text{Im}} \in \mathbb{R}^{(J+N)\times 3}$ are obtained by adding displacements $\Delta\mathbf{X}^{t}_{\text{Im}} \in \mathbb{R}^{(J+N)\times 3}$ predicted by giving the concatenation of joint locations and mesh vertex coordinates in image space at the t-th decoder layer $\mathbf{X}^{t}_{\text{Im}}$ as:

$$\begin{aligned} \mathbf{X}^{t+1}_{\text{Im}} &= \mathbf{X}^{t}_{\text{Im}} + \Delta\mathbf{X}^{t}_{\text{Im}},\ \text{for}\ t > 0. \\ \Delta\mathbf{X}^{t}_{\text{Im}} &= \quad \Psi(\ \Phi(\ \mathbf{X}^{t}_{\text{Im}}) \oplus \mathbf{x}^{t}_{\text{q}}) \end{aligned} \tag{3}$$

where $\mathbf{x}^{t}_{\text{q}} \in \mathbb{R}^{(J+N)\times D}$ is the features of joint and vertex queries at the t-th layer after self-attention. Φ maps the 3D coordinates to high-dimensional features, $\oplus$ concatenates 3D coordinate features with query features and Ψ regresses 3D displacement vectors from them. These 3D displacements are used to refine the mesh vertices/joints and as a result reference points that will be used in MS deformable attention are updated—the reference points are the x and y coordinates of the mesh vertices. As for the initial 3D mesh coordinates $\mathbf{X}^{0}_{\text{Im}}$, we can use the 3D vertex coordinates of a template, a single point $[0, 0, 0]$ or the learned position embeddings obtained using MLPs followed by a sigmoid function [56].

4.3. Dense mesh reconstruction modules

Mesh upsampling and smoothing Depending on the resolution of the reconstructed mesh obtained with the 3D coordinate regressor, we apply upsampling to obtain a dense mesh. When the transformer produces a level-1 (1723 vertices) or level-2 (431 vertices) down-sampled coarse mesh, we upsample the mesh using MLPs [28]. When the transformer produces the original resolution mesh (6890 vertices) we do not perform upsampling. The process is followed by Taubin smoothing [39] to remove undesirable noise and spikes on a surface. To obtain a down-sampled sparse pattern matrix, the down-sampling and upsampling matrices from [37] are used to construct the down-sampled v2j and j2v patterns. For example let $\mathbf{D}_{0\to 1}$ and $\mathbf{U}_{1\to 0}$ be the down-sampling matrix from level-0 to level-1 and the upsampling matrix from level-1 to level-0, respectively. Then, the level-l v2j and j2v matrices are obtained by finding the nonzero values of $\bar{\mathcal{J}}\mathbf{U}_{1\to 0}$ and $\mathbf{D}_{0\to 1}\bar{\mathcal{W}}$.

3D global scale estimator Once the 3D coordinates of joint and mesh vertex are predicted by the transformer, we apply a global scaling to the output 3D coordinates to transform them from the normalized image space to the model space to obtain a mesh in physical size. Specifically, the scale value is regressed from the joint and mesh vertex coordinates in image space using MLPs. We then apply a scaling to $\mathbf{X}_{\mathrm{Im}}$ and transform them to physical model space to obtain $\mathbf{X}_{\mathrm{M}} \in \mathbb{R}^{(J+N)\times 3}$. This allows us to simplify the structure of the decoder layer and its learning process, such that reference points can be obtained without learning a camera projection model within a decoder layer. Note that the above scaling process proceeds in the other way around from previous approaches that project the reconstructed mesh in physical size onto image space [28, 29].

4.4. Training

Following the literature in the field [15, 28, 29, 35], we use a mixed-training strategy using several different training datasets that contain samples with and without image-mesh annotations. To train our model, we minimize the errors between the transformer predictions and the ground truths. In contrast to previous approaches, 3D loss functions are applied on the results reconstructed in the image space and model space. We apply L_1 loss [28] to 3D mesh vertices $\mathbf{V}_{\mathrm{Im}}, \mathbf{V}_{\mathrm{M}} \in \mathbb{R}^{N\times 3}$, 3D body joints $\mathbf{J}_{\mathrm{Im}}, \mathbf{J}_{\mathrm{M}} \in \mathbb{R}^{J\times 3}$ and 2D body joints $\mathbf{J}_{\mathrm{2D}} \in \mathbb{R}^{J\times 2}$. In addition, a loss is calculated from the 3D joint locations regressed from 3D vertex coordinates using the pre-computed joint regressor, $\mathbf{J}^{\mathrm{reg}}_{\mathrm{Im}}$ $\mathbf{J}^{\mathrm{reg}}_{\mathrm{M}} \in \mathbb{R}^{J\times 3}$. We also enforce regularizations on the predictions using the Laplacian operator similarly to [18, 23].

Position loss The positional losses consist of four terms:

$$L^V_{3D} = (||\mathbf{V}_{\mathrm{Im}} - \bar{\mathbf{V}}_{\mathrm{Im}}||_1 + ||\mathbf{V}_{\mathrm{M}} - \bar{\mathbf{V}}_{\mathrm{M}}||_1)/N$$
$$L^J_{3D} = (||\mathbf{J}_{\mathrm{Im}} - \bar{\mathbf{J}}_{\mathrm{Im}}||_1 + ||\mathbf{J}_{\mathrm{M}} - \bar{\mathbf{J}}_{\mathrm{M}}||_1)/J$$
$$L^J_{reg3D} = (||\mathbf{J}^{\mathrm{reg}}_{\mathrm{Im}} - \bar{\mathbf{J}}_{\mathrm{Im}}||_1 + ||\mathbf{J}^{\mathrm{reg}}_{\mathrm{M}} - \bar{\mathbf{J}}_{\mathrm{M}}||_1)/J$$
$$L^J_{2D} = (||\mathbf{J}_{\mathrm{2D}} - \bar{\mathbf{J}}_{\mathrm{2D}}||_1 + ||\mathbf{J}^{\mathrm{reg}}_{\mathrm{2D}} - \bar{\mathbf{J}}_{\mathrm{2D}}||_1)/J$$

where $\bar{\mathbf{V}}_{\mathrm{Im}}, \bar{\mathbf{V}}_{\mathrm{M}} \in \mathbb{R}^{N\times 3}$, $\bar{\mathbf{J}}_{\mathrm{Im}}, \bar{\mathbf{J}}_{\mathrm{M}} \in \mathbb{R}^{J\times 3}$ and $\bar{\mathbf{J}}_{\mathrm{2D}} \in \mathbb{R}^{J\times 2}$ indicate the ground truths for 3D vertex positions in image space/model space, 3D joint positions in image space/model space and 2D joint locations, respectively.

Laplacian loss Let $\mathcal{L}_c$ be the cotangent Laplace operator [18], then the Laplacian loss L_{lap} is written as:

$$L_{lap} = \frac{1}{N}(||\mathcal{L}_c(\mathbf{V}_{\mathrm{Im}} - \bar{\mathbf{V}}_{\mathrm{Im}})||_1) \qquad (4)$$

Total loss The total loss we minimize is as follows:

$$L_{\mathrm{total}} = \alpha(\lambda^V_{3D}(L^V_{3D} + L^J_{reg3D}) + \lambda^J_{3D}L^J_{3D} + \lambda_{lap}L_{lap}) + \beta\lambda^J_{2D}L^J_{2D}$$

where α and β are binary flags for each training sample, which indicate the availability of 3D and 2D ground truths, respectively. λ^V_{3D}, λ^J_{2D}, λ^J_{3D} and λ_{lap} are the weights for controlling the relative strengths of respective terms.

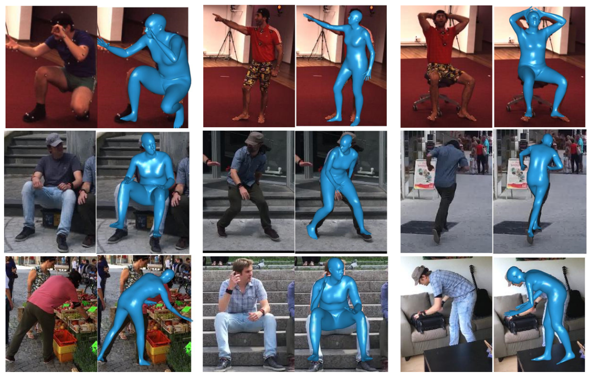


Figure 4. Qualitative results on Human3.6M and 3DPW.

5. Experimental results

This section provides the main results of DeFormer. Additional results are shown in supplementary materials.

5.1. Implementation Details

We implemented our models in Pytorch and used the block sparse attention module from deepspeed [16]. We use ResNet50 [20, 46] and HRNet [38] as our CNNs backbones. We also tested visual transformer backbone models [11, 17, 47, 50]. The backbone models are initialized with the weights pre-trained on ImageNet, MPII or COCO dataset. We found that initializing a backbone model by pre-training on 2D human pose tasks (MPII or COCO) improves the overall performance and training convergence, instead of using ImageNet pre-trained weights. This is in line with previous approaches [24, 52, 55]. The weights in the transformer decoder are randomly initialized. We train our models using the AdamW optimizer for 50 epochs with an initial learning rate of 1×10^{-4} and lower it by a factor of 10 after 25 epochs, following [28, 29, 56]. All of our models were trained on 8 NVIDIA A100 GPUs, which takes about 1 day for most of our models and finishes within 2 days.

5.2. Dataset and metrics

We train our model using publicly available datasets which include Human3.6M [21], MuCo-3DHP [34], UP-3D [27], COCO [30], MPII [8], following [15, 28, 29, 35]. For Human3.6M, we use the pseudo 3D mesh training data from [15, 28, 29, 35]. We also use 3DPW [45] training data to fine-tune on 3DPW following [14, 28, 29]. For evaluations on Human3.6M, we follow the P2 protocol setting, where subjects S1, S5, S6, S7 and S8 are used in training and subjects S9 and S11 are used in testing.

We use the following three metrics for evaluation.
MPJPE: Mean-Per-Joint-Position-Error (MPJPE) measures the Euclidean distances between the ground truth joints and the predicted joints. The unit is millimeter.
PA-MPJPE: PA-MPJPE, or the reconstruction error, is another metric for 3D human pose estimation. Procrustes

Table 1. Comparison of mesh transformer approaches using different backbones. Num. of vertex/joint queries are 431 and 14, respectively.

Method	Epochs	Input size	Feat. resolution	Backbone	Backbone Pre-train dataset	H36M MPJPE ↓ (PA-MPJPE ↓)	#Params Total / Backbone	Inference FPS
METRO [28]	200	224×224	1	ResNet50	ImageNet	56.5 (40.6)	125.8M / 23.5M	32
	200	224×224	1	HRNet-w64	ImageNet	54.0 (36.7)	230.4M / 128.1M	14
	50	256×192	1	HRNet-w48	COCO	48.9 (35.8)	165.2M / 63.6M	16
Graphormer [29]	200	224×224	$\{7, 1\}$	HRNet-w64	ImageNet	51.2 (34.5)	226.5M / 128.1M	14
FastMETRO [14]	60	224×224	7	ResNet50	ImageNet	53.9 (37.3)	48.4M / 23.5M	35
	60	224×224	7	HRNet-w64	ImageNet	52.2 (33.7)	153.0M / 128.1M	14
DeFormer	50	224×224	{56,28,14,7,1}	ResNet50	COCO	50.7 (36.3)	54.5M / 23.5M	41
	50	256×192	{64,32,16,8,1}	HRNet-w48	COCO	**44.8 (31.6)**	93.5M / 63.6M	18
	50	384×288	{64,48,24,12,1}	HRNet-w48	COCO	43.7 (30.7)	93.5M / 63.6M	15

Table 2. Comparisons with other SOTA methods on 3DPW and Human3.6M. ♮ indicates the methods fine-tuned on 3DPW.

Method	3DPW			Human 3.6M	
	MPVE ↓	MPJPE ↓	PA-MPJPE ↓	MPJPE ↓	PA-MPJPE ↓
HMR [22]	—	—	81.3	88.0	56.8
GraphCMR [26]	—	—	70.2	—	50.1
SPIN [25]	116.4	—	59.2	—	41.1
Pose2Mesh [15]	—	89.2	58.9	64.9	47.0
METRO [28]♮	88.2	77.1	47.9	54.0	36.7
Graphormer [29]♮	87.7	74.7	45.6	51.2	34.5
PyMAF [52]♮	87.0	74.2	45.3	54.2	37.2
FastMETRO [14]♮	84.1	73.5	44.6	52.2	33.7
DeFormer♮	**82.6**	**72.9**	**44.3**	**44.8**	**31.6**

Analysis (PA) is first used to align the ground truth and predicted result. Then, MPJPE is calculated. PA-MPJPE measures the error of the reconstructed structure with the effect of scale and rotation removed.

MPVE: Mean-Per-Vertex-Error (MPVE) measures the Euclidean distances between the ground truth vertices and the predicted vertices.

5.3. Comparisons to previous approaches

Table 1 shows the comparison results against previous transformer-based human mesh recovery approaches [14,28,29]. Among the transformer-based approaches using CNNs backbone models, DeFormer with the HRNet-w48 backbone achieves the best performance.

We compare DeFormer with other state-of-the-art techniques on 3DPW and Human3.6M benchmarks. The results are shown in Table 2. On both datasets, DeFormer outperforms previous works including parametric methods that use multi-scale feature maps [52, 53] and transformer-based approaches [14, 28, 29]. This indicates that DeFormer can utilize multi-scale visual features more effectively and map them to 3D vertex coordinates more accurately based on its transformer decoder.

5.4. Ablation study

Multi-scale feature maps. Table 3 investigates the effectiveness of multi-scale feature maps. It compares the performance obtained by varying the number of feature maps to be used. With the HRNet backbones, leveraging multi-scale

Table 3. MPJPE and PA-MPJPE by varying input feature map resolutions. Here, $W^1 = W^0/4$, $W^2 = W^0/8$, $W^3 = W^0/4$ and $W^4 = W^0/32$ represent feature map widths where W^0 is the input image width. Feature maps are visualized in Fig. 5.

Backbone-Pre-train data	Feature map resolutions			
	W^4	$W^{3,4}$	$W^{2,3,4}$	$W^{1,2,3,4}$
R50-MPII	54.1 (39.6)	53.7 (38.6)	53.6 (38.4)	54.5 (39.1)
R50-COCO	50.5 (36.8)	51.2 (36.5)	50.6 (36.2)	50.7 (36.3)
HR32-MPII	56.9 (41.7)	52.2 (37.4)	49.9 (35.7)	49.5 (34.8)
HR48-COCO	50.9 (36.5)	47.5 (34.2)	46.6 (33.1)	**44.8 (31.6)**

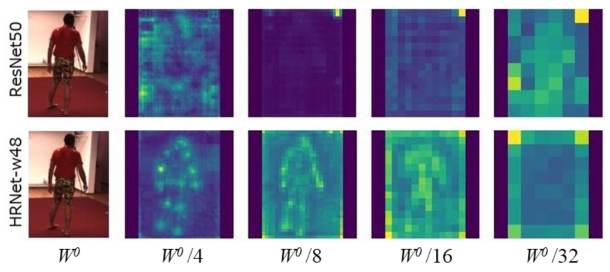


Figure 5. Visualizations of multi-scale feature maps from ResNet50 (top) and HRNet-w48 (bottom) backbones pre-trained on COCO 2D keypoints and then trained for the human mesh recovery task for 50 epochs. Left to right: Input image and feature maps from backbone layers $l = 1, 2, 3, 4$. Feature maps are summed along the channel dimension. HRNet produces meaningful spatial feature maps at high resolutions.

feature maps improves the performance. Table 3 clearly shows a decrease in the MPJPE and PA-MPJPE errors with the introduction of high-resolution feature maps in the case of HRNet. However, it is not the case with the ResNet50 backbone model. To see why it shows such a different behavior depending on the backbone model, we visualize the feature maps extracted from them (Fig. 5). It is found that the feature maps of ResNet50 have less meaningful spatial patterns at higher resolutions, which is why its multi-scale feature maps do not contribute much to the performance improvements. In contrast, HRNet produces meaningful spatial feature maps for all resolutions. This is probably because its network architecture has multi-scale fusion layers that promote interactions between feature maps at different levels throughout the network, allowing high-resolution feature maps to participate in.

Table 4. DeFormer MPJPE ↓ (PA-MPJPE↓) scores using different visual transformer backbones. All backbone models are pre-trained on COCO. Input image size is 256×192.

Backbone (#Params)	MPJPE (PA-MPJPE)	Total #Params
ViTPose-L (299.8M)	56.9 (39.0)	329.9M
Swin-L-FPN (186.6M)	49.7 (33.8)	216.7M
AggPose-L (87.4M)	43.6 (31.3)	117.5M
HRFormer-B (43.2M)	**43.3 (30.0)**	64.0M

Table 5. Ablation study on decoder cross attention (CA).

CA type	MPJPE ↓	PA-MPJPE↓
Bilinear interp. ×3	48.3	34.3
Transformer CA ×3	50.7	37.1
MS deformable CA ×1	47.7	34.8
MS deformable CA ×3	**44.8**	**31.6**

Table 6. Ablation on self-attention (SA). Peak mem. usage indicates per device memory usage for each GPU device. Benchmarks are run on 8 Nvidia A100 GPUs. Training batch size is 16. Note FPS has improved from Table 1 with some code optimizations.

SA type	#Query	MPJPE ↓ (PA-MPJPE)	Peak mem. usage	Steps per sec	FPS
Transformer	0.4K	44.8 (31.6)	7.2GB	4.33	**19.8**
	1.7K	44.5 (**30.3**)	25.2GB	2.68	18.5
	6.9K	—	—	—	9.6
Body Sparse	0.4K	45.2 (31.0)	**6.1GB**	**4.43**	19.6
	1.7K	44.3 (30.5)	**8.4GB**	**3.40**	**19.2**
	6.9K	**43.9** (30.7)	**17.8GB**	**1.80**	**14.7**

Backbone models. In Table 1, we show the comparison results against METRO [28] with the HRNet-w48 backbone pre-trained on COCO. With this change, METRO improves PA-MPJPE by 1.4mm and MPJPE by 5.1mm from the original version [28] that employs the HRNet-w64 model pre-trained on ImageNet. DeFormer with the same HRNet-w48 backbone model performs better than METRO by 3.9mm in PA-MPJPE and 4.3mm in MPJPE by exploiting multi-scale feature maps via MS deformable attention.

Table 4 shows the results using four different visual transformer backbone models. Using visual transformer backbone models, DeFormer further gains improvements in MPJPE and PA-MPJPE. Surprisingly, however, we found that the improvements are dependent on their model architectures, irrespective of the model size. Even though ViTPose [47] achieves SOTA performances in 2D human pose detection tasks, it does not perform well with DeFormer, because ViT does not produce multi-scale feature maps and is known to work poorly in dense prediction tasks. Swin-L-FPN [17] has an aggregation mechanism of multi-scale feature maps based on feature pyramids [31]. It works better than the ViT backbone but it is not as effective as HRNet variants. AggPose [11] and HRFormer-B [50] possess multi-scale fusion layers similar to HRNet. Thus, DeFormer with them shows strong performances.

Decoder cross attention. Table 5 shows the comparison between the decoder cross attention with different feature sampling and aggregation approaches. Specifically, we compare bilinear interpolation [53], standard cross attention [12] and MS deformable cross attention [56]. To conduct a fair comparison experiment, the above decoders are comprised of three layers and we use the same backbone model (HRNet-w48). We also tested a one-layer setting for MS deformable attention. For bilinear interpolation, the decoder layers take feature maps similar resolutions with the ones used in [53] i.e. $\{64, 32, 16\}$. Similarly, for the decoder with standard cross attention, a feature map with resolution of 8 is fed into the decoder following [14]. The deformable attention decoder uses all feature maps from $\{64, 32, 16, 8, 1\}$ resolutions at each layer. As we can see from Table 5, the deformable attention performs the best among the three approaches by leveraging all level of multi-scale feature maps at each decoder layer. By increasing the number of decoder layers from one to three, it gains 3pt improvements in both MPJPE and PA-MPJPE.

Decoder self-attention. Table 6 shows the ablation on the self-attention modules. Overall, the MPJPE and PA-MPJPE scores are almost equivalent when using the full self-attention and the body sparse attention. Due to high memory consumption, it is not possible to train DeFormer with the full transformer self-attention for the 6.9K query setting using batch size of 16 on A100 GPUs with 40GB memory. With the use of body sparse self-attention, DeFormer can be trained in less than 2 days even with 6.9K queries. An increase in the number of vertex queries i.e. the mesh resolution of the template provides approximately +1pt improvements in the MPJPE and PA-MPJPE scores from $44.8(31.6)$ to $43.9(30.7)$. In addition, inference FPS can be evaluated for the 6.9K query setting. DeFormer with BodySparse-SA is $1.5\times$ faster than that with the standard transformer self-attention. An increase in the speed is relatively limited as we do not use an efficient local window attention commonly used in language and image domain, due to the irregular connectivity of a triangle mesh. Our current implementation is also not optimized to fully exploit the GPU block sparse computing of CUDA.

6. Conclusion

We presented DeFormer, a decoder-only transformer for 3D human mesh recovery, and addressed memory issues of transformer attention by introducing the efficient mesh-driven attention mechanisms: deformable mesh attention and body sparse attention. DeFormer can thus leverage multi-scale image feature maps and a dense human mesh body model, achieving a SOTA performance in human mesh recovery. We are interested in extending DeFormer to learn expressive human body models that can capture fine-grained details, such as facial expressions and finger poses.

Acknowledgment This work was supported by JSPS KAKENHI Grant Number 20K19836 in Japan. Y.Y. is supported by JSPS KAKENHI Grant Number 22H00545.

References

[1] Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 2

[2] Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2

[3] End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 2

[4] Mesh graphormer. In *ICCV*, 2021. 2

[5] Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *ECCV*, 2022. 2

[6] Joshua Ainslie, Santiago Ontañón, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. Etc: Encoding long and structured data in transformers. In *EMNLP*, 2020. 3

[7] Patrick R. Amestoy, Timothy A. Davis, and Iain S. Duff. Algorithm 837: Amd, an approximate minimum degree ordering algorithm. *ACM Trans. Math. Softw.*, 30(3):381–388, 2004. 5

[8] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Schiele Bernt. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 6

[9] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020. 3

[10] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *ECCV*, pages 561–578. Springer, 2016. 2

[11] Xu Cao, Xiaoye Li, Liya Ma, Yi Huang, Xuan Feng, Zening Chen, Hongwu Zeng, and Jianguo Cao. AggPose: Deep aggregation vision transformer for infant pose estimation. In *IJCAI*, 2022. 6, 8

[12] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, Cham, 2020. 8

[13] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509, 2019. 3

[14] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *ECCV*, 2022. 1, 2, 6, 7, 8

[15] Hongsuk Choi, Gyeongsik Moon, and Kyoung Mu Lee. Pose2mesh: Graph convolutional network for 3d human pose and mesh recovery from a 2d human pose. In *ECCV*, 2020. 1, 2, 6, 7

[16] DeepSpeed Contributors. Deepspeed. https://github.com/microsoft/DeepSpeed, 2022. 1, 6

[17] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020. 6, 8

[18] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. 3d-coded : 3d correspondences by deep deformation. In *ECCV*, 2018. 6

[19] P. Guan, A. Weiss, A. Balan, and M. J. Black. Estimating human shape and pose from a single image. In *ICCV*, pages 1381–1388, 2009. 2

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[21] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE TPAMI*, 36(7):1325–1339, 2014. 6

[22] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 2, 5, 7

[23] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 6

[24] Muhammed Kocabas, Chun-Hao P. Huang, Otmar Hilliges, and Michael J. Black. PARE: Part attention regressor for 3D human body estimation. In *ICCV*, pages 11127–11137, 2021. 6

[25] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *ICCV*, 2019. 2, 7

[26] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 1, 2, 4, 7

[27] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J. Black, and Peter V. Gehler. Unite the people: Closing the loop between 3d and 2d human representations. In *CVPR*, 2017. 2, 6

[28] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 1, 2, 4, 5, 6, 7, 8

[29] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *ICCV*, 2021. 1, 2, 3, 4, 6, 7

[30] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 6

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017. 8

[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 3

[33] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, 2015. 1, 2, 4, 5

[34] Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll, and Christian Theobalt. Single-shot multi-person 3d pose estimation from monocular rgb. In *3DV*. IEEE, 2018. 6

[35] Gyeongsik Moon and Kyoung Mu Lee. I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single rgb image. In *ECCV*, 2020. 1, 6

[36] Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sinong Wang, and Jie Tang. Blockwise self-attention for long document understanding. In *EMNLP*, 2020. 3

[37] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. In *ECCV*, pages 704–720, 2018. 5

[38] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 2, 6

[39] Gabriel Taubin, Tong Zhang, and Gene Golub. *Optimal surface smoothing as filter design*, volume 1064 of *LNCS*, pages 283–292. 1996. 5

[40] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Comput. Surv.*, 2022. 1, 3

[41] Yating Tian, Hongwen Zhang, Yebin Liu, and Limin Wang. Recovering 3d human mesh from monocular images: A survey. *arXiv preprint arXiv:2203.01923*, 2022. 1, 2

[42] Gül Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. BodyNet: Volumetric inference of 3D human body shapes. In *ECCV*, 2018. 2

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, volume 30, 2017. 3, 4

[44] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *ICLR*, 2017. 3

[45] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *ECCV*, 2018. 6

[46] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018. 6

[47] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. ViTPose: Simple vision transformer baselines for human pose estimation. In *NeurIPS*, 2022. 6, 8

[48] Xingguang Yan, Liqiang Lin, Niloy J. Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *CVPR*, pages 6239–6249, 2022. 3

[49] Yang Ye and Shihao Ji. Sparse graph attention networks. *IEEE TKDE*, 2021. 3

[50] Yuhui Yuan, Rao Fu, Lang Huang, Weihong Lin, Chao Zhang, Xilin Chen, and Jingdong Wang. Hrformer: High-resolution transformer for dense prediction. In *NeurIPS*, 2021. 6, 8

[51] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *NeurIPS*, 33, 2020. 1, 3

[52] Hongwen Zhang, Yating Tian, Yuxiang Zhang, Mengcheng Li, Liang An, Zhenan Sun, and Yebin Liu. Pymaf-x: Towards well-aligned full-body model regression from monocular images. *arXiv preprint arXiv:2207.06400*, 2022. 1, 6, 7

[53] Hongwen Zhang, Yating Tian, Xinchi Zhou, Wanli Ouyang, Yebin Liu, Limin Wang, and Zhenan Sun. Pymaf: 3d human pose and shape regression with pyramidal mesh alignment feedback loop. In *ICCV*, 2021. 1, 2, 5, 7, 8

[54] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu. Deephuman: 3d human reconstruction from a single image. In *ICCV*, 2019. 2

[55] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: A weakly-supervised approach. In *ICCV*, pages 398–407, 2017. 6

[56] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 1, 2, 3, 4, 5, 6, 8