Coarse-to-Fine Human Mesh Recovery with Transformers

Vatsal Agarwal[1*] Mara Levy[1] Max Ehrlich[1] Youbao Tang[2] Ning Zhang[2]
Abhinav Shrivastava[1]

[1]University of Maryland [2]PAII Inc.

Abstract

The introduction of Transformer networks in computer vision has resulted in rapid progress of deep models in a variety of vision tasks. Recently, there has been great success in utilizing such networks for the human mesh recovery task. While these works demonstrate remarkable performance, they suffer from high computational cost and slow speed due to the quadratic nature of the self-attention mechanism. In this work, we propose a coarse-to-fine modeling approach to improve the pipeline efficiency. We build upon previous approaches and adopt an encoder-decoder architecture to mine relationships between image, joint and vertex features. While previous works apply attention on the full set of vertex features, our key insight is that earlier model layers do not require such dense vertex representations and instead can rely on a sparser set of features. We evaluate our approach on the Human3.6M and 3DPW datasets and find that with our coarse-to-fine approach, we are able to achieve improved or competitive performance with up to 2x decrease in parameters as well as a 10x reduction in FLOPS and a 5x reduction in activation count.

1. Introduction

Estimating the 3D mesh of a human from a single image has become an increasingly popular task in computer vision. The ability to extract high-quality human models has tremendous applications for understanding human interactions as well as improving immersive technology such as augmented and virtual reality. To generate a human mesh, 3D coordinates are regressed for each vertex of the mesh representation. This process is time-consuming, however, as many practical settings require fast 3D mesh reconstruction to be done in real-time. Additionally, these settings often rely on using information from only a single view. Recently, deep learning methods have made considerable progress in improving this task. However, several challenges remain due to the complexity of different human body shapes and poses. These are further exacerbated in the monocular setting because of severe occlusions and ambiguities in scale and depth.

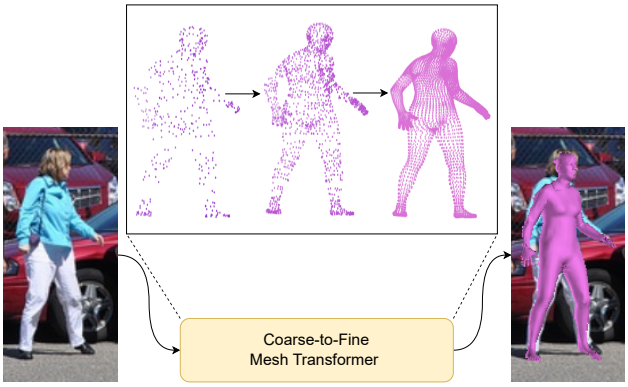

Figure 1. We propose a coarse-to-fine approach with Transformers to generate a robust mesh representation starting from a small set of initial vertex features and gradually increasing the mesh resolution.

There have been many strategies proposed to accurately and efficiently recover human shape and pose from an image [2, 3, 13, 16, 18, 19, 37]. These approaches can be broadly separated into two categories: model-based and model-free. Model-based methods rely on a parametric model of the human body such as SMPL [21] and SMPL-X [28] to generate the full mesh. Specifically, they train a neural network pipeline to estimate body and shape parameters that are then input to the parametric model to generate the final mesh. The primary benefit of such an approach is that the parametric model encodes a strong prior for plausible human shapes and positions. This comes at the cost of generalizability, however, as reconstructions in the wild are constrained by pre-defined human models. Furthermore, these model parameters are difficult for neural networks to regress. For instance, SMPL represents pose as a set of 3D rotations, which are difficult for neural networks to represent [16, 41].

In contrast, model-free methods aim to directly regress

[*]Work done during internship at PAII.

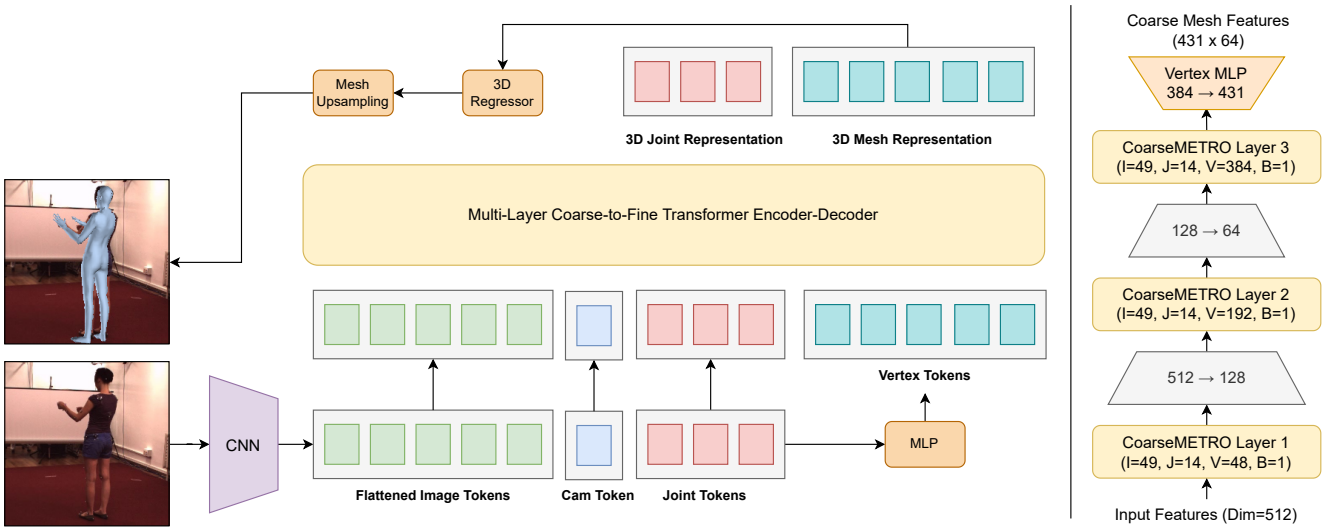

Figure 2. **Overview of the proposed framework.** Given an input image, we extract 7×7 grid features from a convolutional neural network (CNN) backbone. Given an initial set of J joint tokens, we apply MLP-based upsampling to obtain a set of coarse vertex features. We then feed the image and camera tokens along with the 3D joint and vertex tokens to our encoder-decoder network to obtain our mesh representation.

the 3D vertex coordinates. Works that use model-free methods have made significant improvements in performance. Recently works rely on the Transformer architecture to model non-local interactions between mesh vertices and body joints. Such approaches have two primary benefits. First, since each vertex has a corresponding feature, the model is better able to capture more precise articulations. Second, these models can easily be extended to different settings such as hand or full-body mesh reconstruction as they are not dependent on a parametric model. Despite obtaining impressive performance, these approaches incur a large computational cost concerning parameters and memory. Due to the one-to-one correspondence between mesh vertices and mesh features, it is not scalable to operate on the fine-grained mesh, which consists of 6890 vertices. To address this issue, most works apply self-attention on a coarse mesh with 431 vertices. However, this is still problematic for settings that require near real-time mesh recovery, as applying full self-attention over 400 vertices multiple times is cost-prohibitive.

In this work, we propose a simple but effective solution to address this issue by introducing a coarse-to-fine pipeline as shown in Fig. 1 that can be integrated easily to any existing Transformer-based approach, denoted as CoarseMETRO. Our approach is motivated by the observation that there is still considerable redundancy in the existing coarse mesh with 431 vertices. As such, we learn a set of coarse vertex tokens at different granularities to capture global and local mesh information. We utilize the extensive modeling capacity of the Transformer to learn these tokens without explicit guidance. With such a design, our network can allocate vertex tokens adaptively to focus on important body regions for a particular image. This is crucial in cases where the complexity of the pose is limited to one region such as the hands or legs and thus requires more attention.

Concretely, rather than using a fixed resolution for the vertex tokens, we instead initialize them as a function of the joint tokens and gradually upsample them across each stage of the network. We showcase the benefits of our approach by adopting the FastMETRO protocol and utilizing a Transformer-based encoder-decoder architecture. Specifically, the encoder operates on a set of image tokens obtained from a pre-trained feature extractor and a learnable camera token that is used to regress scale and translation parameters. The decoder input is a sequence of learnable joint and vertex tokens and applies self-attention to model non-local interactions between the joint and vertex features. Furthermore, it applies cross-attention to encourage better correspondences between the vertex and joint tokens and the image features.

We make a few key changes to the architecture to integrate our coarse-to-fine pipeline. First, we only construct one set of learnable tokens to represent the joint features. We then generate the initial vertex tokens by applying a set of linear layers on the aforementioned joint tokens. In doing so, we essentially initialize the vertex tokens as a function of the joint tokens and ensure both are the same resolution. The constructed tokens are then fed to the decoder and progressively upsampled by a scalar factor at each stage via a simple linear layer. To enforce consistency between ver-

Table 1. Comparison with transformers for monocular 3D human mesh recovery on Human3.6M [11]. Inference-time evaluations were done using a single V100 GPU with a batch size of 1. GFLOPs and Activation Counts were calculated using [30]. Best in **bold** second best underlined.

Model	Transformer				Overall	
	#Params	Time (ms)	GFLOPs	Activation Counts	#Params	PA-MPJPE ↓
FastMETRO–S–R50 [2]	**9.2M**	9.6	2.2G	7.6M	**32.7M**	39.4
FastMETRO–M–R50 [2]	17.1M	15.0	4.4G	15.3M	40.6M	38.6
FastMETRO–L–R50 [2]	24.9M	20.8	6.6G	22.9M	48.4M	37.3
TORE–FastMETRO-L–R50 [6]	23.5M	–	**0.6G**	–	50.2M	40.4
FastMETRO–L–H64 [2]	24.9M	20.8	6.6G	22.9M	153.0M	**33.7**
CoarseMETRO–S–R50 (Ours)	11.6M	**9.4**	**0.6G**	**4.5M**	35.1M	38.7
CoarseMETRO–S–H64 (Ours)	11.6M	**9.4**	**0.6G**	**4.5M**	139.7M	35.3

tex tokens at each granularity, we employ a single cross-attention layer between the upsampled and initial vertex tokens. Notably, each set of coarse vertex tokens are learned without requiring any pre-defined mesh information. The main contributions of this work are as follows:

- We propose constructing vertex tokens dynamically from a coarse resolution to finer resolutions via simple linear layers. This drastically reduces the memory cost of the Transformer block across the network.
- Our cross-granularity vertex cross-attention aids the model in encouraging consistency between vertex tokens of different resolutions thereby enabling accurate feature upsampling.

2. Related works

2.1. Human Mesh Recovery

Human Mesh Recovery (HMR) aims to recover 3D human shape and pose and has been an area of longstanding interest in the vision community. Given the challenge of extracting both a coherent and plausible human mesh from a single image, there have been a diverse set of approaches proposed for this task.

Many prior works propose using a neural network to estimate parameters for a parametric model that has been trained on a large number of human body models such as SMPL [21], SMPL-X [28], and MANO [31]. In order to further enhance performance, these works often generate intermediate representations that can capture crucial pose and shape information such as keypoints [38], part segmentations [13], IUV maps [37], and 3D keypoints [3]. These approaches can exploit the strong prior encoded in the parametric model to improve robustness to more complex scenes and difficult viewpoints. However, training these networks is difficult as the pose parameters they aim to predict are 3D rotations and are challenging to regress.

Recently, there has been a growing body of work that advances non-parametric modeling of human mesh and has demonstrated impressive results. GraphCMR [16] was one of the first papers to propose directly regressing the 3D coordinates of the human body mesh via graph convolutional neural networks (GCNNs). More specifically, they first used a pre-trained image-based network to extract image features and then concatenated them with 3D coordinates of a template SMPL mesh. These features were then fed to a GCNN to mine local vertex feature interactions and progressively deform the mesh to obtain an accurate human body pose and shape. Pose2Mesh [3] built upon this further by using a 2D skeleton as input and a coarse-to-fine approach that progressively upsamples the skeleton to generate the final mesh. It does the upsampling via a sequence of GCNNs. METRO [19] proposes using a Transformer encoder to model non-local interactions and uses a similar approach to [16]. MeshGraphormer [18] builds upon METRO by introducing a GCNN layer within their attention block to explicitly mine local interactions. FastMETRO [2] proposes a solution where the image features and 3D joint and vertex features are processed independently via an encoder-decoder design and learns a correspondence between the two sets of features through a cross-attention mechanism.

Besides, there have been several works that have explored techniques to improve model efficiency and performance simultaneously. TORE [6] improves upon FastMETRO by constraining the cross-attention to only focus on image-joint interactions and utilizes a token-pruning technique for reducing the number of attended image tokens. In a similar direction, DeFormer [36] eliminates the encoder entirely and instead proposes a decoder-only architecture that takes advantage of block-sparse attention and multi-scale features to further reduce memory costs. POTTER [39] explores an orthogonal direction for achieving low computational cost via a new efficient backbone architecture. Most recently, [23] investigates a similar direction to ours and proposes learning a set of virtual markers that can serve as an intermediate representation for generating accurate meshes.

Our work is inspired by [3] and we adopt the architecture of [2] to showcase the potential of our approach. The proposed pipeline differs from the aforementioned works in a few crucial ways. First, unlike TORE, our approach enables the modeling of all vertex tokens across the entire network rather than limiting them until the last stage. Such a restriction prevents the model from properly learning shape information which is critical in constructing a robust mesh. With respect to DeFormer, we note that our pipeline improves performance without the need for multi-scale features which are costly due to increased spatial resolution. Additionally, our efficiency comes from using a compact set of vertex tokens, while theirs uses hardware-specific techniques that may not be able to be adopted across all hardware. Lastly, while [23] has a similar motivation to our work, they rely on learning these markers via a two-stage approach thereby increasing the complexity of training. Additionally, these markers are only a single resolution and therefore prevent the model from learning more fine-grained details explicitly. Another important distinction is that our pipeline is easy to adopt and can be integrated into any Transformer architecture with minimal changes required.

2.2. Transformers

[33] first introduced the Transformer architecture in the field of natural language processing and showcased impressive performance on a diverse set of language tasks. These successes were translated to the computer vision field with the development of ViT [5]. The seminal work proposed splitting an image into fixed-length patches and feeding them to a sequence of Transformer layers and demonstrated the efficacy of a pure Transformer architecture for vision tasks. One caveat, however, is that ViT requires much more training data compared to CNNs. Since then, there have been several works that have been proposed to apply Transformers to a variety of vision tasks, including image generation, object detection, image segmentation, pose estimation, and more recently human mesh recovery.

3. Method

Fig. 2 shows an overview of our approach. Specifically, we first feed an input image with size 224×224 and predict J body joint coordinates and V mesh vertex coordinates, where $J = 14$ and $V = 6890$. We use three modules to accomplish this task. First, a pretrained CNN extracts image features which capture global context. Three sets of learnable embedding tokens represent the joints, vertices, and camera information respectively following [2]. Finally, a Transformer encoder-decoder architecture progressively generates a coarse mesh. We explore the details of our architecture below.

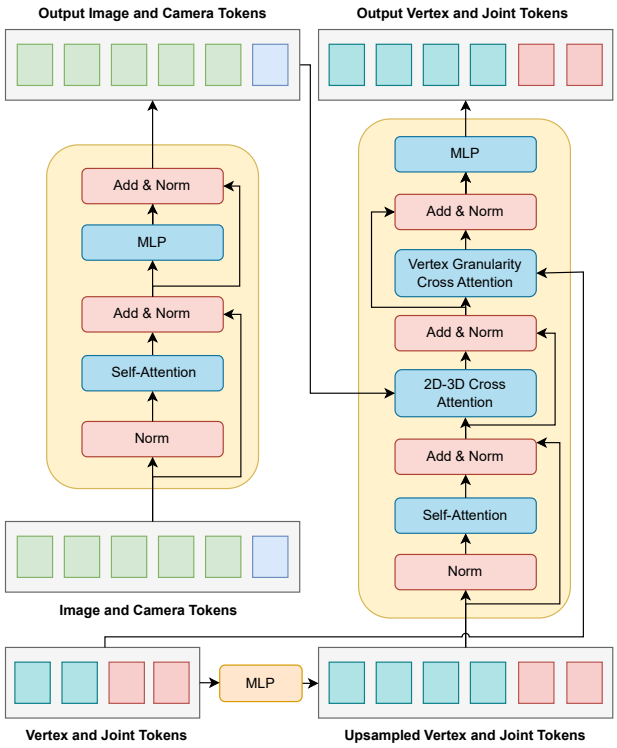

Figure 3. Architecture of our encoder-decoder module. We propose incorporating a cross-attention module to enforce feature consistency between low- and high-resolution vertex features at each stage of the network.

3.1. Feature Extractor

The first part of our pipeline utilizes a convolutional neural network (CNN) for extracting relevant features from the image. This network is pretrained for ImageNet classification [4]. To obtain image features for our Transformer encoder, we feed an input image to generate $7 \times 7 \times D$ coarse image features and then flatten them. Thus, we generate 49 image tokens which we denote as $\mathbf{X}_I \in \mathbb{R}^{49 \times D}$, where $D = 512$.

3.2. Coarse-to-Fine Transformer

The second part of our pipeline is the coarse mesh generation and is shown in Fig. 3. Our network learns to progressively generate a fine-grained mesh and predict the 3D human pose given the image features, $\mathbf{X}_I$ and an initial set of learnable joint tokens $\mathbf{X}_J \in \mathbb{R}^{J \times D}$, representing the number of modeled joints. Specifically, we employ an encoder-decoder architecture with three stages with a single encoder-decoder block in each stage. We condition our mesh representation on the initial joint tokens by applying an MLP layer on them to obtain the initial vertex tokens: $\mathbf{X}_V \in \mathbb{R}^{48 \times D}$. The joint and vertex tokens are used to predict the 3D coordinates of the joints and mesh vertices respectively. The details of our encoder-decoder architecture

Table 2. Ablation study of the Masked Joint Modeling objective using different percentages of masked joint tokens, evaluated on Human3.6M.

Max Percentage	0%	10%	20%	30%	40%	50%
PA-MPJPE	39.4	38.7	**37.8**	38.1	41.5	40.5
MPJPE	57.3	**54.6**	55.9	56.5	58.1	57.2

and upsampling mechanism are presented below.

3.2.1 Transformer Encoder

Our proposed Transformer encoder (as shown in Fig. 3 captures global-level features for the mesh generation. To this end, we propose using the encoder to mine non-local interactions between the image features. Additionally, following [2], we feed a learnable camera token $\mathbf{X}_C$ which captures crucial image and body information to predict the weak-perspective camera parameters. These camera parameters are used to fit the estimated human mesh to the 2D input image. Given these features, the transformer encoder produces three sets of refined features: $\mathbf{X}_C \in \mathbb{R}^{1\times D}$, $\mathbf{X}_I \in \mathbb{R}^{49\times D}$.

3.2.2 Transformer Decoder

The proposed Transformer decoder shown in Fig. 3 refines the joint and vertex features via a series of self-attention and cross-attention modules. Specifically, our module takes in a set of three learnable embeddings: $\mathbf{X}_J \in \mathbb{R}^{J\times D}, \mathbf{X}_{V'} \in \mathbb{R}^{V'\times D}$, and $\mathbf{X}_V \in \mathbb{R}^{V\times D}$. $\mathbf{X}_J$ corresponds to the set of joint tokens, while $\mathbf{X}_{V'}$ and $\mathbf{X}_V$ correspond to the set of vertex tokens at the original resolution and upsampled resolution respectively. We first model the global interactions between the joint and upsampled vertex features via a self-attention layer. Next, we apply cross-attention between the refined joint and vertex features and the output image and camera features from the Transformer encoder to ensure the spatial coherence of the 3D features.

Finally, we enforce feature consistency between the updated joint and vertex tokens and the original resolution vertex tokens by applying another cross-attention layer. Specifically, we feed the updated joint and vertex tokens as the queries and the original resolution vertex tokens as the keys and values. Similar to [19], we apply random masking on the joint features as an augmentation method to enable our network to be robust to occlusions.

3.2.3 Coarse to Fine Upsampling Layer

In this section, we describe the specific implementation of our coarse-to-fine pipeline. Following each encoder-decoder stage, we use a coarse-to-fine upsampling layer to gradually increase the feature resolution starting with an initial set of 48 vertex tokens. Previous works [2, 18, 19] utilized a progressive downsampling approach to gradually reduce the embedding dimension D across each stage before predicting the 3D coordinates. To accomplish this, we use an MLP layer to increase the number of vertex tokens at the end of each stage. Our latent vertex features consist of 48, 192, and 384 tokens at each stage respectively. We then apply another MLP layer on the last set of vertex features to generate the coarse mesh representation $V_{3D} \in \mathbb{R}^{431\times 3}$. To predict the 3D coordinate locations of 14 key joints, we similarly apply an MLP layer on the joint tokens and obtain $J_{3D} \in \mathbb{R}^{14\times 3}$. We follow the same protocol as [2] to generate our fine-grained mesh representation. Specifically, we first apply an MLP layer to upsample from 431 to 1723 vertices. We then use the pre-computed matrix from [29] to obtain our final 6890×3 mesh.

Table 3. Ablation study of vertex-vertex cross attention.

Vertex-Vertex Cross Attention	PA-MPJPE	MPJPE
None	40.7	58.5
Vertex Cross-Attention at Beginning	40.3	55.3
Vertex Cross-Attention at End	**38.7**	**54.6**

3.3. Training Details

3.3.1 3D Vertex Regression Loss

Given the output mesh coordinates $\hat{V}_{3D}$ and the ground truth mesh coordinates $\bar{V}_{3D}$, the 3D regression loss is computed as follows:

$$L_V = \frac{1}{M}||\hat{V}_{3D} - \bar{V}_{3D}||_1$$

3.3.2 3D Joint Regression Loss

For the joint regression loss, we use two sets of predicted joint coordinates. The first set is obtained from our model directly, which predicts 3D joint coordinates, $\hat{J}_{3D}$ in the coarse-to-fine pipeline. The second set of joint coordinates are obtained from our predicted mesh coordinates $\hat{V}_{3D}$. Specifically, we use a predefined joint regression matrix $G \in \mathbb{R}^{J\times V}$ as is common practice in the literature. Thus, our mesh generated 3D coordinates are $\hat{J'}_{3D} = G\hat{V}_{3D}$. Our 3D joint regression loss is as follows:

$$L_J = \frac{1}{M}(||\hat{J}_{3D} - \bar{J}_{3D}||_1 + ||\hat{J'}_{3D} - \bar{J}_{3D}||_1)$$

3.3.3 2D Joint Re-Projection Loss

Given the sparse amount of 3D mesh annotations available, previous works have also relied on 2D re-projection losses on the joint predictions to ensure alignment between the predicted 3D joints and the input image. We use our camera token to predict weak-perspective scale and translation

Table 4. Performance comparison with the previous state-of-the-art methods on 3DPW and Human3.6M datasets. $*$ indicates our training results for the model following the specified protocol.

1*	3DPW			Human3.6M	
Method	MPVE ↓	MPJPE ↓	PA-MPJPE ↓	MPJPE ↓	PA-MPJPE ↓
HMR–R50 [12]	–	130.0	76.7	88.0	56.8
GraphCMR–R50 [16]	–	–	70.2	–	50.1
SPIN–R50 [15]	116.4	96.9	59.2	62.5	41.1
I2LMeshNet–R50 [25]	–	93.2	57.7	55.7	41.1
PyMAF–R50 [37]	110.1	92.8	58.9	57.7	40.5
ROMP–R50 [32]	105.6	89.3	53.5	–	–
PARE–R50 [13]	99.7	82.9	52.3	–	–
METRO–R50 [19]	–	–	–	56.5	40.6
DSR–R50 [7]	99.5	85.7	51.7	60.9	40.3
FastMETRO-S–R50 [2]	**91.9**	**79.6**	**49.3**	55.7	39.4
FastMETRO-S–R50 [2]*	94.5	82.0	49.5	57.4	39.1
TORE-FastMETRO-L–R50 [6]	96.8	80.3	50.6	59.5	40.4
CoarseMETRO-S-R50 (Ours)	93.4	80.9	50.4	**54.6**	**38.7**
METRO–H64 [19]	88.2	77.1	47.9	54.0	36.7
MeshGraphormer–H64 [18]	87.7	74.7	45.6	**51.2**	34.5
FastMETRO-L–H64 [2]	**84.1**	**73.5**	**44.6**	52.2	**33.7**
FastMETRO-L–H64 [2]*	85.6	75.8	45.1	55.9	35.8
TORE-FastMETRO–H64 [6]	88.8	74.2	44.7	57.4	35.1
CoarseMETRO-S-H64 (Ours)	93.9	82.8	50.4	52.3	35.4

parameters, $s, \mathbf{t}$ where $s \in \mathbb{R}$ and $\mathbf{t} \in \mathbb{R}^2$. We obtain the predicted 2D joint locations by using the scale and translation parameters to form an orthographic projection matrix. We apply this re-projection on both the directly predicted 3D joint coordinates and the 3D joint coordinates obtained from the mesh. Thus the 2D joint re-projection loss is as follows:

$$L_J^{2D} = \frac{1}{M}(||\hat{J}_{2D} - \bar{J}_{2D}||_1 + ||\hat{J'}_{2D} - \bar{J}_{2D}||_1)$$

3.3.4 Total Loss

Following existing literature, we train our model with a combination of 3D and 2D training datasets and use a series of loss coefficients for each loss as well as two binary flags, α and β depending on whether 3D or 2D annotations are available or not. Thus the total loss is as follows:

$$L_{\text{total}} = \alpha \cdot (L_V + L_J) + \beta \cdot L_J^{2D}$$

4. Experiments

Our training paradigm matches that of [2]. Specifically, we use the AdamW [22] optimizer with a learning rate of 10^{-4}. We set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and use a weight decay of 10^{-4}. Furthermore, we also apply gradient-clipping based on the gradient norm, and set the clipping norm value to 0.3. Our network is trained with a batch-size of 16 for 60 epochs. All networks with a ResNet-50 backbone were trained on 4 A100 GPUs for approximately 2 days, while networks with a HRNet-W64 backbone were trained on 8 A100 GPUs for approximately 4 days. At training time, we apply the standard data augmentation protocol following existing works [2, 14, 18, 19].

4.1. Datasets

Following the prior Transformer based works, we train our model with the Human3.6M [11], UP-3D [17], MuCO-3DHP [24], COCO [20], and MPII [1] training sets. For evaluation, we use the P2 protocol in Human3.6M. We evaluate our model's ability to generalize to more difficult poses and scenes via the 3DPW dataset [34]. Following previous works, we first finetune our model with the 3DPW training set. Additionally, given that ground-truth is not provided for the Human3.6M dataset, we follow prior Transformer-based works and use pseudo 3D human mesh obtained by SMPLify-X [28] to train our models. To ensure a fair comparison, we use the ground-truth 3D human joints in the testing set for evaluation. For our feature extractor, we use ResNet-50 [10]. We also investigate the use of more powerful backbones such as HRNet-W64 [35].

Figure 4. Qualitative results of CoarseMETRO on Human3.6M and 3DPW. We visualize the results from the CoarseMETRO–S–H64 model. We show several different categories of poses including occlusions to show that our method is robust to many types on images.

4.1.1 Evaluation Metrics

We use three evaluation metrics: MPJPE [11], PA-MPJPE [40], and MPVPE [27]. The units for these metrics is millimeters. MPJPE (mean-per-joint-position-error) measures the Euclidean distance between the predicted and ground-truth joint 3D coordinates. PA-MPJPE uses Procrustes Analysis [9] to perform 3D alignment and then measures the MPJPE. Lastly, MPVPE (mean-per-vertex-position-error) measures the Euclidean distances between the predicted and ground-truth vertex coordinates.

4.2. Experimental Results

In order to demonstrate the efficacy of our approach, we evaluate our model on the Human3.6M and 3DPW datasets. Each dataset has their own unique set of challenges. For instance, the Human3.6M dataset consists of simpler scenes where the difficulty lies in accurately modeling human shape. Contrastingly, the 3DPW dataset captures more in-the-wild scenes and requires a model to generalize well to a variety of settings and human poses. We find that our model achieves competitive performance on both benchmarks. We primarily perform comparisons against other Transformer-based methods including METRO [19], Mesh Graphormer [18], FastMETRO [2], and TORE [6]. For fair comparisons, we evaluate our model with two backbone settings, namely ResNet-50 and HRNet-W64. The results are shown in Table 4. We also rerun FastMETRO's experiments and report the results we obtain after training the models with the provided code.

We first observe that our model with the ResNet-50 backbone improves upon FastMETRO by 0.8mm for PA-MPJPE and 0.9mm for MPJPE on the Human3.6M dataset. This highlights that our coarse-to-fine approach is able to in-fact leverage redundancy in the vertex token representation to generate a more robust mesh representation with respect to pose and shape. Furthermore, compared to TORE, we find that our model has remarkable improvements on both metrics for the Human3.6M dataset despite using only a single Transformer encoder-decoder layer. When applying the HRNet-W64 backbone, we note that our trained version of FastMETRO performs quite worse than reported results on the Human3.6M dataset (by about 2mm for both metrics). Moreover, we observe that our model is able to closely match the FastMETRO-L configuration while only using a tenth of the FLOPS. When comparing against TORE, we can see a 5.1mm improvement in MPJPE while using the same number of FLOPS and fewer parameters.

When examining our results on the 3DPW dataset, we note our approach is fairly competitive with FastMETRO and TORE for the ResNet-50 backbone, however there is a relative performance drop when using the higher-resolution backbone. We hypothesize that this could be a limitation of using only a single Transformer block in each layer.

4.2.1 Computational Efficiency

In this section, we analyze the computational complexity of our approach and previous Transformer methods. We show these results in Table 1. We find that our coarse-to-fine approach is able to reduce the number of Transformer parameters by 51% while still maintaining performance. Furthermore, our approach leads to a 10x decrease in FLOPS and 5x reduction in activation counts. It is crucial to note that our network still models vertex features across all stages of the network and is thus able to still learn shape information explicitly rather than through just joint token as in [6].

4.3. Ablation Results

We conduct an extensive array of experiments on the Human3.6M dataset to demonstrate the effectiveness of each of our components. First, we investigate how incorporating cross-attention between vertex features of different granularities improves performance as shown in Table 3. To this

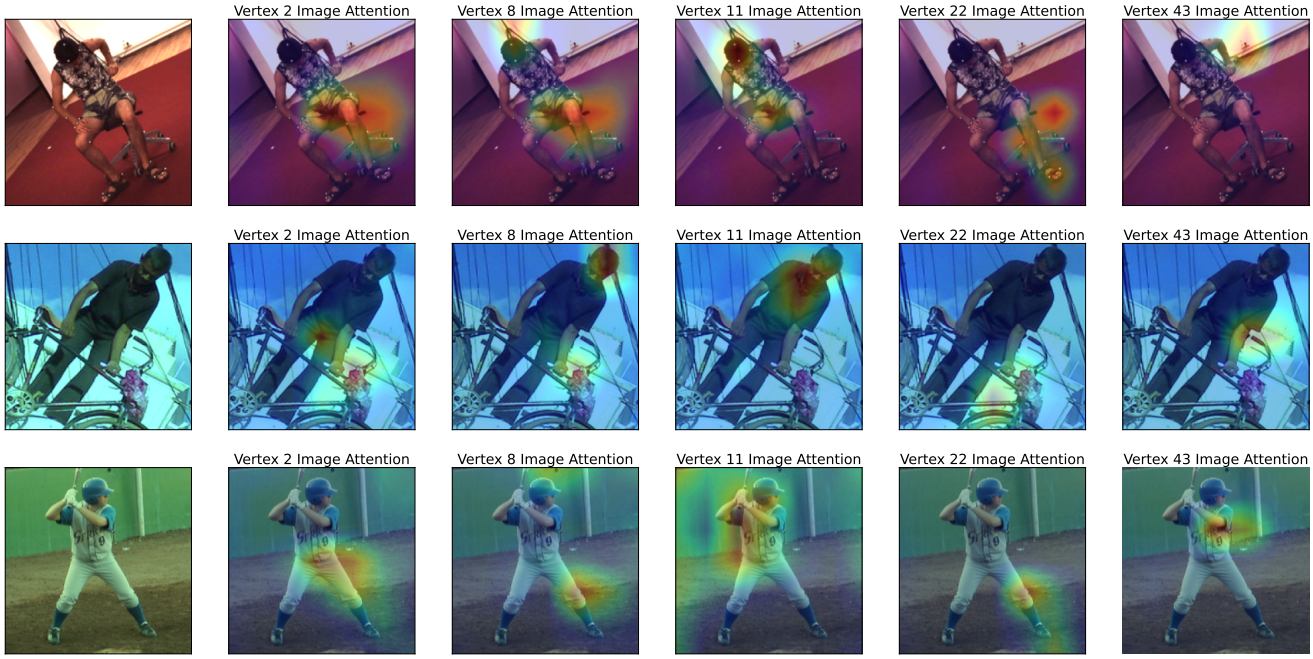

Figure 5. **Decoder Vertex Cross-Attention Maps** These attention maps visualize which image regions our vertex tokens are attending to at the first stage of our network. Note that our vertex tokens consistently attend to meaningful body regions across different images.

end, we first train a baseline model with no cross-attention and observe a noticeable decrease in performance by about 2mm in PA-MPJPE and 4mm for MPJPE. This indicates that cross-attention is crucial for the network to more reliably upsample the coarse set of vertex features into the full mesh representation. Furthermore, we explore different placement strategies, specifically either applying cross-attention at the beginning or end of the decoder network. We observe that while adding the cross-attention at the start does improve upon the baseline, the 2D-3D cross-attention applied subsequently could reduce its effects.

Many previous works incorporating Transformers for human mesh recovery have relied on some form of masking to increase the robustness of the model. METRO [19] and Mesh Graphormer [18] specifically utilized random masking on the joint and vertex features, while FastMETRO [2] applies masking on non-adjacent joints to force the Transformer to focus on local relationships. Since our initial vertex representation is formed from an initial set of 14 joint tokens, we only apply random masking on this set of features. The results are shown in a Table 2. We observe that applying 10-30% masking achieves good results while masking too many of the input tokens (i.e. 40-50%) prevents the network from modeling important inter-joint and joint-image relationships.

4.4. Attention Map Analysis

In order to better understand what relationships are being captured by our attention maps, we visualize the intermediate features in our Transformer block and examine which regions our latent vertex features are attending to.

As shown in Fig. 5, we inspect the cross-attention maps between specific vertex tokens and all of the image tokens from the first decoder layer in our network. We average the attention scores from all heads and reshape the attention maps to match the original image size. We observe that the specific tokens accurately focus on the relevant regions in the image. Despite our latent vertex tokens not being defined by a pre-existing mesh topology, we find that they are able to learn to meaningful body parts in the image consistently across different images. For example, vertex 2 can be seen to focus mostly on the left-hip area of the person, while vertex 43 seems to focus more on the left hand/wrist.

5. Conclusion

In this work, we introduce a coarse-to-fine approach for human mesh recovery using an encoder-decoder Transformer architecture. Specifically, we introduce a strategy to initialize a mesh representation with a set of joint tokens and then gradually upsample and refine these features through a novel cross-attention module. We evaluate this model on the Human3.6M and 3DPW dataset and showcase competitive results with a remarkable reduction in parameters, FLOPs, and activation counts. Future work could build upon this and apply non-parametric models to the task of human mesh recovery from videos. Other avenues would be to incorporate human pose and shape priors to even further boost model efficiency.

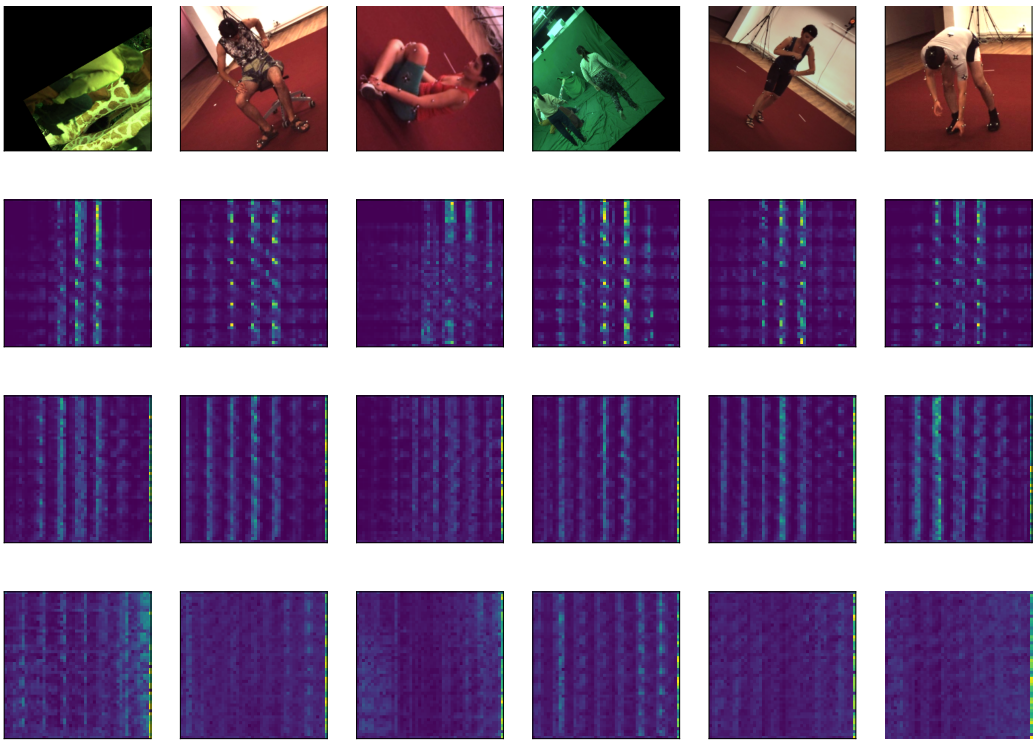

Figure 6. **Encoder Self-Attention Maps** This shows the self-attention maps between the image features and the camera token in the encoder block. Each row displays the attention map from the corresponding layer (e.g. the first row shows the maps from the first encoder layer). It can be seen that the attention maps initially focus on specific image tokens and then gradually attend more towards the camera token features.

6. Implementation Details

In this section, we describe in further detail the codes used to develop CoarseMETRO and our choice of hyperparameters for training. We implemented our CoarseMETRO model using Pytorch [26] and experimented with two pre-trained feature extractors: ResNet-50 [10] and HRNet-W64 [35]. Each backbone network was initialized with pre-trained ImageNet [4] weights and further fine-tuned during training. We use the backbone network to generate an output feature map of size $7 \times 7 \times 2048$ and apply a single MLP layer to reduce the channel dimension to size 512. Our Transformer model consists of three encoder-decoder layers with a single block per layer and is randomly initialized using Xavier [8] initialization. Each attention module in our network uses 8 heads. Similar to [2], we apply sinusoidal positional encodings on our flattened image features before passing them to the Transformer network. Additionally, we also rely on a sparse pre-computed mesh upsampling matrix from [29] to upsample our coarse mesh representation with 1723 vertices to obtain the fine-grained 6890-vertex mesh. This matches the number of vertices in the SMPL [21] model.

7. Extended Attention Map Analysis

In this section, we extend our visualization of the attention maps to examine the intermediate features in the encoder blocks of the Transformer as well as validate that our joint tokens are corresponding to the correct regions of the image.

We first examine the attention maps captured by our Transformer encoder layers as shown in Fig. 6. Each layer's attention maps are shown row-wise. It can be observed that the initial attention maps focus more on specific image tokens corresponding to regions of interest. The last two stages of attention maps highlight how the encoder gradually focuses more on the camera token. This demonstrates that our model is effectively distilling image-level information into the camera token over multiple Transformer blocks while still placing some focus on the relevant image regions.

Next, we demonstrate that our joint tokens meaningfully attend to the image tokens. In Fig. 7, we inspect the cross-attention maps between specific joint tokens and all of the image tokens from the first decoder layer in our network.

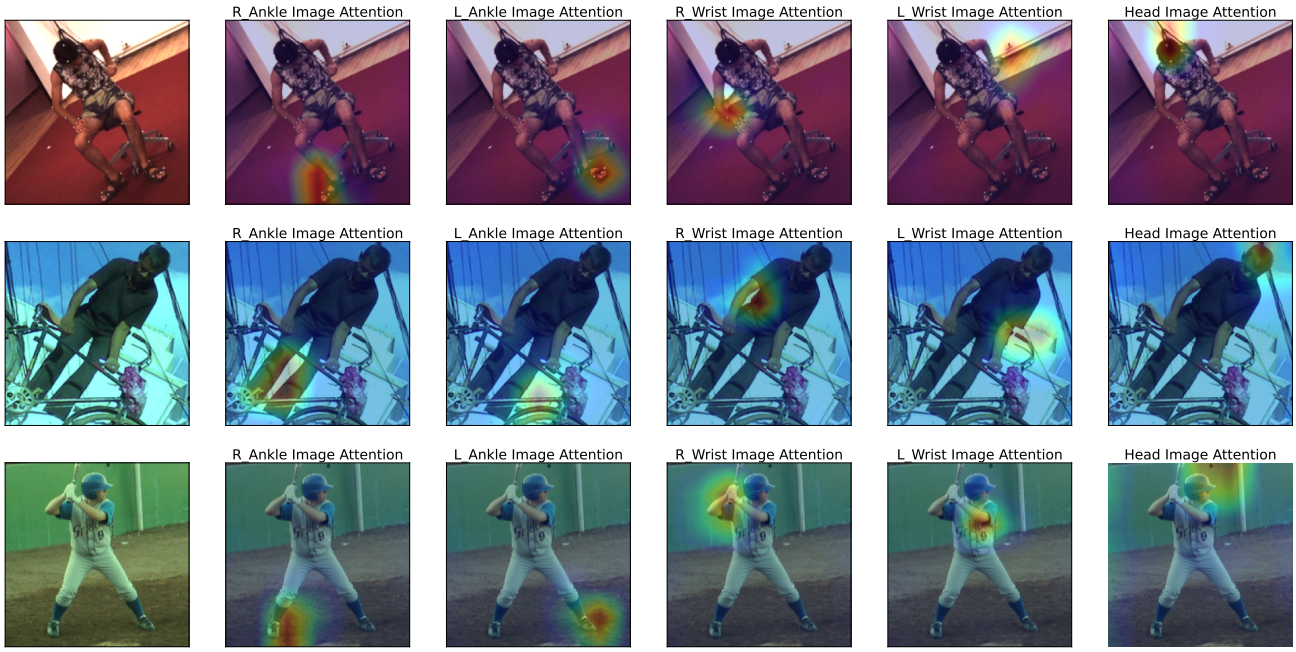

Figure 7. **Decoder Joint Cross-Attention Maps** These attention maps visualize which image regions our joint tokens are attending to at the first stage of our Transformer network. We observe that our joint tokens consistently attend to the key corresponding areas in the image.

As previously mentioned, we average the attention scores from all heads and reshape the attention maps to match the original image size. We observe that the specific tokens accurately focus on the relevant regions in the image. For instance, on the third row it can be seen that the joint token corresponding to the left ankle is attending most to the baseball player's left ankle.

8. Additional Results

We display additional results of our proposed approach on the Human3.6M [11] and 3DPW [34] test sets in Fig. 8. As can be seen, our model can robustly capture the human shape and pose despite challenging occlusions and background settings.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 6

[2] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh. Cross-attention of disentangled modalities for 3d human mesh recovery with transformers. In *European Conference on Computer Vision*, pages 342–359. Springer, 2022. 1, 3, 4, 5, 6, 7, 8, 9

[3] 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 *European Conference on Computer Vision*, pages 769–787. Springer, 2020. 1, 3, 4

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 4, 9

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[6] Zhiyang Dou, Qingxuan Wu, Cheng Lin, Zeyu Cao, Qiangqiang Wu, Weilin Wan, Taku Komura, and Wenping Wang. Tore: Token reduction for efficient human mesh recovery with transformer, 2022. 3, 6, 7

[7] Sai Kumar Dwivedi, Nikos Athanasiou, Muhammed Kocabas, and Michael J. Black. Learning to regress bodies from images using differentiable semantic rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11250–11259, 2021. 6

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010. 9

[9] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975. 7

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.

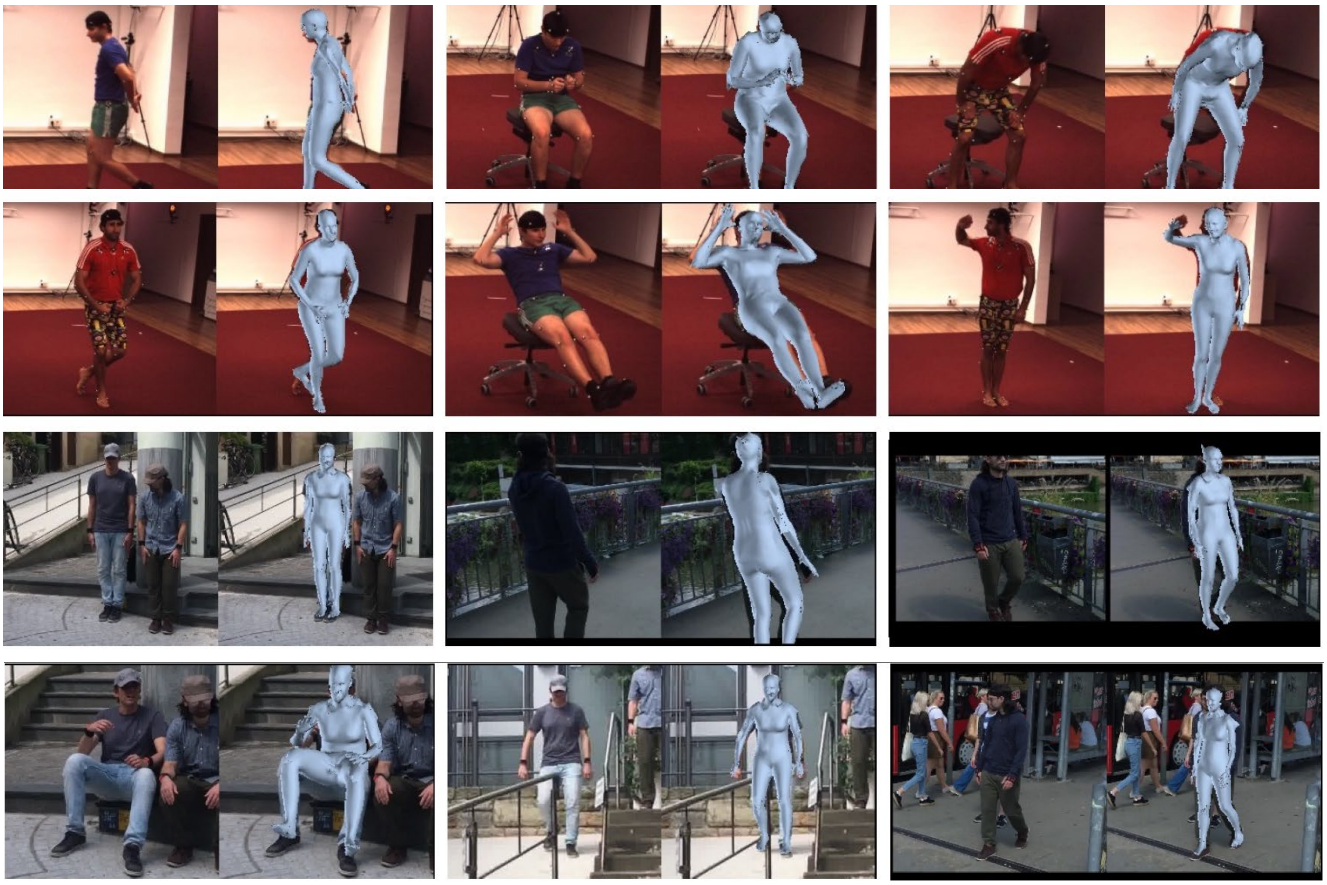

Figure 8. **Extended Qualitative Results on Human3.6M and 3DPW**. We visualize more results from the CoarseMETRO–S–H64 model. We again see that our model is robust to variations in different scenes with respect to pose, background, and camera viewpoint.

Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778. IEEE, 2016. 6, 9

[11] 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 transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013. 3, 6, 7, 10

[12] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 6

[13] Muhammed Kocabas, Chun-Hao P Huang, Otmar Hilliges, and Michael J Black. Pare: Part attention regressor for 3d human body estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11127–11137, 2021. 1, 3, 6

[14] 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. 6

[15] Nikos Kolotouros, Georgios Pavlakos, Michael J. Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. *CoRR*, abs/1909.12828, 2019. 6

[16] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 1, 3, 6

[17] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6050–6059, 2017. 6

[18] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12939–12948, 2021. 1, 3, 5, 6, 7, 8

[19] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1954–1963, 2021. 1, 3, 5, 6, 7, 8

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Mi-

crosoft coco: Common objects in context, 2014. cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list. 6

[21] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 1, 3, 9

[22] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[23] Xiaoxuan Ma, Jiajun Su, Chunyu Wang, Wentao Zhu, and Yizhou Wang. 3d human mesh estimation from virtual markers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 534–543, 2023. 3, 4

[24] 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 *2018 International Conference on 3D Vision (3DV)*, pages 120–130. IEEE, 2018. 6

[25] 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. 6

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 9

[27] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3d human pose and shape from a single color image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 459–468, 2018. 7

[28] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019. 1, 3, 6

[29] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. In *Proceedings of the European conference on computer vision (ECCV)*, pages 704–720, 2018. 5, 9

[30] Meta AI Research. fvcore. https://github.com/facebookresearch/fvcore, 2023. 3

[31] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), 2017. 3

[32] Yu Sun, Qian Bao, Wu Liu, Yili Fu, Michael J Black, and Tao Mei. Monocular, one-stage, regression of multiple 3d people. In *ICCV*, 2021. 6

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[34] 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 *European Conference on Computer Vision (ECCV)*, 2018. 6, 10

[35] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020. 6, 9

[36] Yusuke Yoshiyasu. Deformable mesh transformer for 3d human mesh recovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17006–17015, 2023. 3

[37] 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 *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1, 3, 6

[38] Ce Zheng, Matias Mendieta, Taojiannan Yang, Guo-Jun Qi, and Chen Chen. Feater: An efficient network for human reconstruction feature map-based transformer. 3

[39] Ce Zheng, Xianpeng Liu, Guo-Jun Qi, and Chen Chen. Potter: Pooling attention transformer for efficient human mesh recovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1620, 2023. 3

[40] Xiaowei Zhou, Menglong Zhu, Georgios Pavlakos, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Monocap: Monocular human motion capture using a cnn coupled with a geometric prior. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):901–914, 2018. 7

[41] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 1