Supplementary Material for
MPT: Mesh Pre-Training with Transformers for Human Pose and Mesh Reconstruction

Kevin Lin, Chung-Ching Lin, Lin Liang, Zicheng Liu, Lijuan Wang
Microsoft
{keli, chungching.lin, lliang, zliu, lijuanw}@microsoft.com

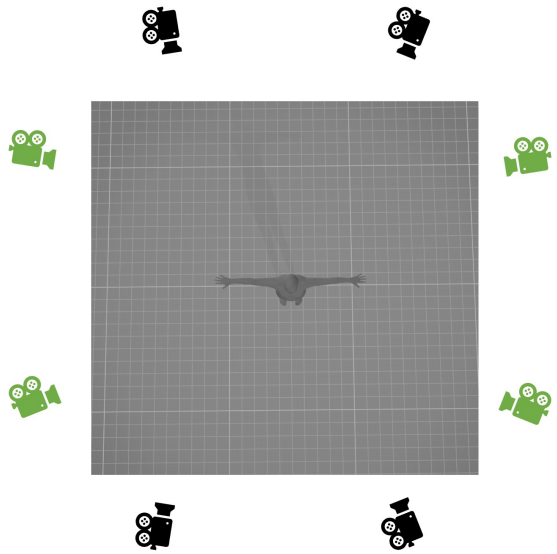

Figure 1. Illustration of our virtual cameras in a top-down view. In order to set up a virtual camera at a proper position, we follow the camera settings of Human3.6M to set up 4 virtual cameras (indicated in black color). In addition, we perform interpolation to obtain extra virtual camera positions (indicated in green color).

A. Model Capacity Comparison

Table 1 shows the model capacity comparison between our approach and METRO [4]. It shows that the 2D pose estimation backbone in our framework is approximately 4.5 times smaller than the backbone used in METRO. Despite this smaller model size, our method outperforms METRO in terms of reconstruction error.

B. Details of Virtual Cameras

During pre-training, we use virtual cameras to help synthesize heatmaps. In order to set up the virtual cameras at proper positions, we follow the camera settings of Human3.6M [2] in our experiments. To be specific, we set up 4 virtual cameras using the camera parameters from Subject 1 of Human3.6M training set.

Method	Backbone	# Bkbone Param	# Trans Param	MPJPE ↓	PA-MPJPE ↓
METRO	*img feat bkbone*	128M	102M	54.0	36.7
Ours	*2D pose net*	28M	102M	45.3	31.7

Table 1. Comparison of computational cost.

Number of Views	MPJPE ↓	PA-MPJPE ↓
1	92.5	61.7
2	89.5	61.4
4	89.0	58.4
8	88.3	58.2

Table 2. Adding more virtual camera views. We evaluate the pre-trained model on Human3.6M validation set without fine-tuning.

C. Adding More Virtual Cameras

We further investigate whether adding more camera views can improve the performance. Given the 4 virtual cameras defined by Human3.6M, we perform interpolation to obtain additional 4 virtual camera views. Figure 1 illustrates our virtual cameras using an example. The 4 virtual cameras defined by Human3.6M are denoted in black color. The interpolated virtual cameras are denoted in green color.

In Table 2, we observe that adding more camera views in pre-training improves the performance for both metrics on Human3.6M. Note that we evaluate the pre-trained model without fine-tuning.

D. Visualization of Pose Feature Maps

In Figure 2, we present additional visualization of the pose feature maps. By fine-tuning our pre-trained MPT model, we empirically observed that our model learns to extract pose feature maps where each map captures information on multiple body joints, and our model generates a human mesh with more reasonable pose and shape.

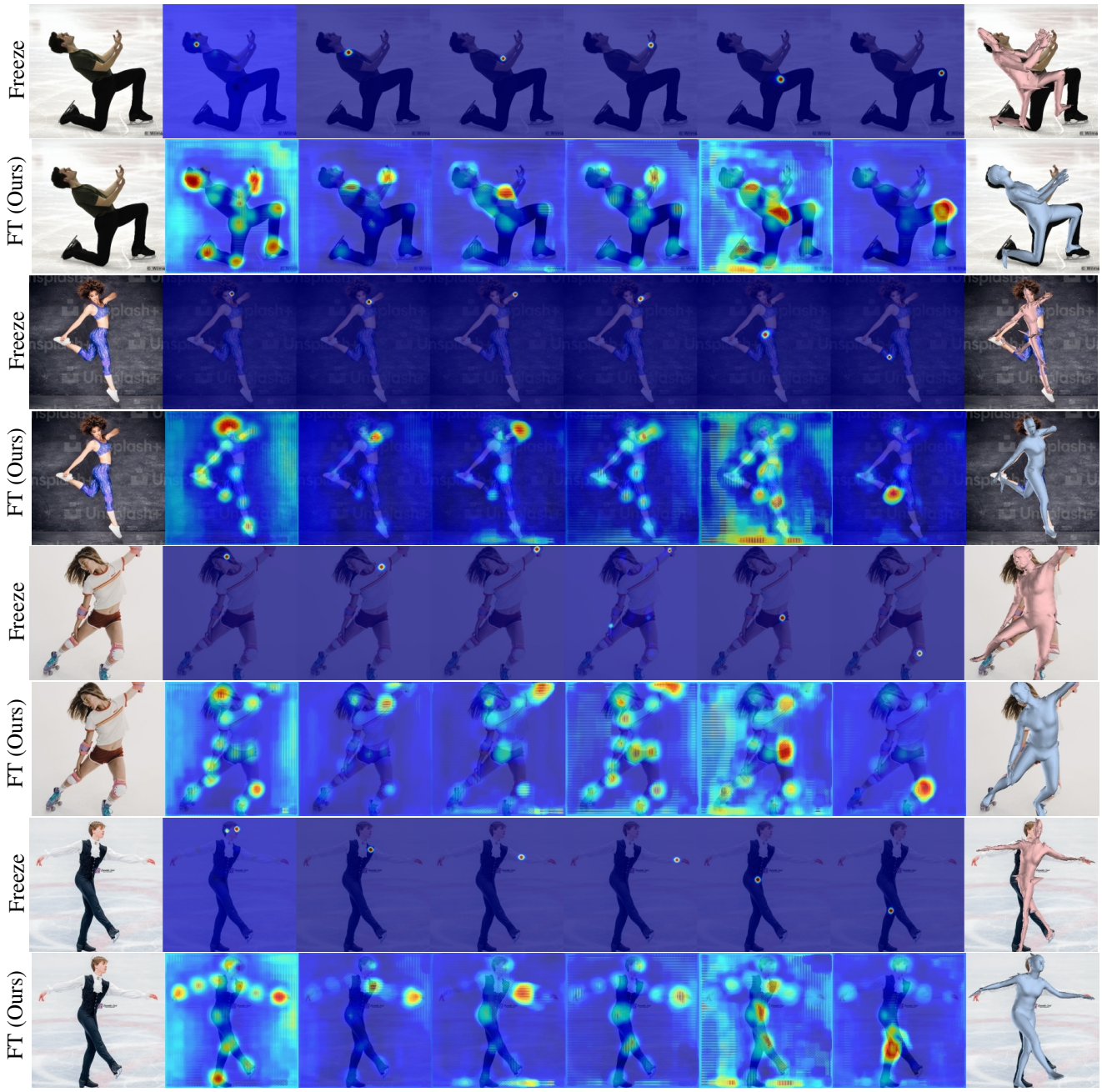

Figure 2. Visualization of pose feature maps. Our model learns to extract pose feature maps where each map captures information on multiple body joints, leading to improved reconstruction.

E. Resolution of Pose Feature Maps

As we use HigherHRNet [1] to generate pose feature maps, one may wonder what resolution is needed. Table 3 shows the performance comparison with different resolutions. The results suggest that using larger resolution (*i.e.,* 224×224) can slightly improve the results. We note that HigherHRNet was pre-trained using the non-cropped images [1], but we use 224×224 cropped images instead.

Resolution	MPJPE ↓	PA-MPJPE ↓
112×112	46.5	32.7
224×224	45.3	31.7

Table 3. Different resolutions of the pose feature maps. We conduct fine-tuning on a mixture of 2D and 3D training sets, and evaluate the performance on Human3.6M validation set.

Input features	MPJPE ↓	PA-MPJPE ↓
Binary Human seg. mask	55.6	35.4
2D pose heatmaps (Ours)	45.3	31.7

Table 4. Pre-training with different input features.

We think that increasing the input resolution to the original resolution on which HighHRNet was trained could further enhance model performance.

F. Additional Implementation Details

We implement our models based on PyTorch [7] and Graphormer codebase [4, 5]. We additionally adopt DeepSpeed [8] which empirically leads to faster and more stable training. We use 16 NVIDIA V100 GPUs for most of our training experiments. We use the Adam optimizer for training. We set the initial learning rate as 2×10^{-4}, and then use learning rate warmup over the first 10% training steps followed by linear decay to 0. We perform mesh pre-training on the 2 million meshes (sparsely sampled from AMASS dataset [6]) for 10 epochs. When fine-tuning on mixed datasets, we empirically fine-tune for 80 epochs.

We use the same transformer architecture as in the literature [4, 5]. Specifically, our transformer model has 3 transformer blocks. Each block has 4 transformer layers and 4 attention heads. For the 3 transformer blocks, the hidden sizes are 1024, 256, 64, respectively. To reduce the computational cost, the transformer model outputs a coarse mesh. We then use MLPs to upsample the predicted mesh to the original resolution, similar to [4, 5]. For simplicity, we do not add graph convolutions to the transformer layers.

Regarding the details of 2D pose estimation model, we use HigherHRNet [1] based on `HRNet-w48` architecture. In our ablation study, we replace HigherHRNet with SimpleBaseline [9], which is based on `ResNet101` architecture. Both models are initialized with COCO Keypoint pre-trained weights.

Following the literature [3, 5], we use a weak perspective camera model for calculating 2D re-projection loss. Our model predicts camera parameters, including a scaling factor s and a 2D translation vector t. Our method does not leverage ground truth camera parameters. The camera parameters are learned by optimizing 2D re-projection.

G. Discussion on Segmentation and 2D Pose Backbones

We empirically found that a semantic segmentation backbone is not suitable for our framework. There are several reasons: First, acquiring segmentation feature maps during pre-training is challenging as we do not have RGB images available in pre-training. While one might consider using a rendering engine to generate photo-realistic human images for feature extraction, such rendering process is a lot more expensive than heatmap synthesis and existing rendering engines may not completely bridge the domain gap. Secondly, although we can synthesize a binary human segmentation mask by re-projecting the 3D mesh onto the 2D image plane, this binary mask does not accurately represent human pose in cases of (self-)occlusion. Table 4 shows a comparison between using 2D pose heatmaps and binary human masks. It shows that 2D pose heatmaps yield better results in our framework.

H. Limitations and Societal Impact

3D pose estimation models can be potentially applied to human activity analysis applications, such as detecting whether the senior subjects are falling. However, there is a risk associated with directly applying the models for mission-critical decision making, especially in the health care field. Real-world applications may require an auxiliary supervision model or task-specific fine-tuning.

Our models have a dependency on existing MoCap training data. Some of the public released MoCap data may be licensed, meaning they can only be used for scientific research purposes. Users must strictly adhere to the data usage agreement to utilize MoCap datasets for their intended purposes.

References

[1] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. In *CVPR*, 2020. 2, 3

[2] 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, 2014. 1

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

[4] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 1, 3

[5] Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *ICCV*, 2021. 3

[6] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *ICCV*, 2019. 3

[7] 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. *NeurIPS*, 2019. 3

[8] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training

deep learning models with over 100 billion parameters. In *KDD*, 2020. 3

[9] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *ECCV*, 2018. 3