2021 IEEE/CVF International Conference on Computer Vision (ICCV) | 978-1-6654-2812-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCV48922.2021.01270

Mesh Graphormer

Kevin Lin Lijuan Wang Zicheng Liu
Microsoft
{keli, lijuanw, zliu}@microsoft.com

Abstract

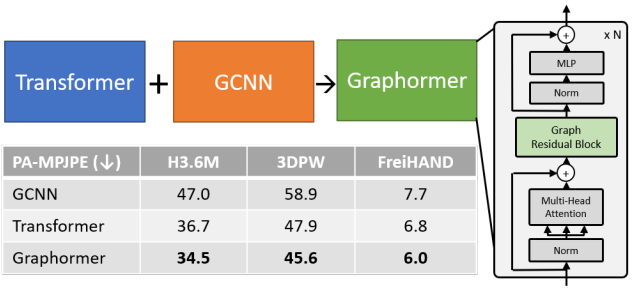
We present a graph-convolution-reinforced transformer, named Mesh Graphormer, for 3D human pose and mesh reconstruction from a single image. Recently both transformers and graph convolutional neural networks (GCNNs) have shown promising progress in human mesh reconstruction. Transformer-based approaches are effective in modeling non-local interactions among 3D mesh vertices and body joints, whereas GCNNs are good at exploiting neighborhood vertex interactions based on a pre-specified mesh topology. In this paper, we study how to combine graph convolutions and self-attentions in a transformer to model both local and global interactions. Experimental results show that our proposed method, Mesh Graphormer, significantly outperforms the previous state-of-the-art methods on multiple benchmarks, including Human3.6M, 3DPW, and FreiHAND datasets. Code and pre-trained models are available at https://github.com/microsoft/MeshGraphormer.

1. Introduction

3D human pose and mesh reconstruction from a single image is a popular research topic as it offers a wide range of applications for human-computer interactions. However, due to the complex body articulation, it is a challenging task.

Recently, Transformers and Graph Convolutional Neural Networks (GCNNs) have shown promising advances in human mesh reconstruction. For example, recent studies [20, 5] have suggested using GCNNs to directly regress 3D positions of mesh vertices by taking into account local interactions among neighbor vertices. In a recent study [22], a transformer encoder was used with self-attention to capture global interactions among body joints and mesh vertices, which lead to further improvement.

However, as discussed in the literature [41, 40, 8, 41, 3], Transformers and Convolution Neural Networks (CNN) each have their own limitations. Transformers are good at modeling long-range dependencies on the input tokens, but they are less efficient at capturing fine-grained local information. Convolution layers, on the other hand, are useful for extracting local features, but many layers are required to capture global context. In natural language processing and speech recognition, Conformer [8] is a recently proposed technique that leverages the complementarity of self-attention and convolutions to learn representations. This motivates us to combine self-attention and graph convolution for the 3D reconstruction of human mesh (Figure 1).



PA-MPJPE (↓)	H3.6M	3DPW	FreiHAND
GCNN	47.0	58.9	7.7
Transformer	36.7	47.9	6.8
Graphormer	**34.5**	**45.6**	**6.0**

Figure 1: Summary. We study how to combine self-attentions and graph convolutions in a transformer for human mesh reconstruction. The proposed Graphormer outperforms existing graph convolution networks [5, 20] and transformer-based method [22] by a clear margin. The numbers are the reconstruction error (PA-MPJPE) in the unit of millimeter. The lower the better.

We present a graph-convolution-reinforced transformer called Mesh Graphformer for reconstructing human pose and mesh from a single image. We inject graph convolutions into transformer blocks to improve the local interactions among neighboring vertices and joints. In order to leverage the power of graph convolutions, Graphormer is free to attend to all image grid features that contain more detailed local information and are helpful in refining the 3D coordinate prediction. Consequently, Graphormer and image grid features are mutually reinforced to achieve better performance in human pose and mesh reconstruction.

Extensive experiments show that the proposed Mesh Graphormer models both local and global interactions effectively and clearly outperforms previous state-of-the-art

978-1-6654-2812-5/21/$31.00 ©2021 IEEE
DOI 10.1109/ICCV48922.2021.01270

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

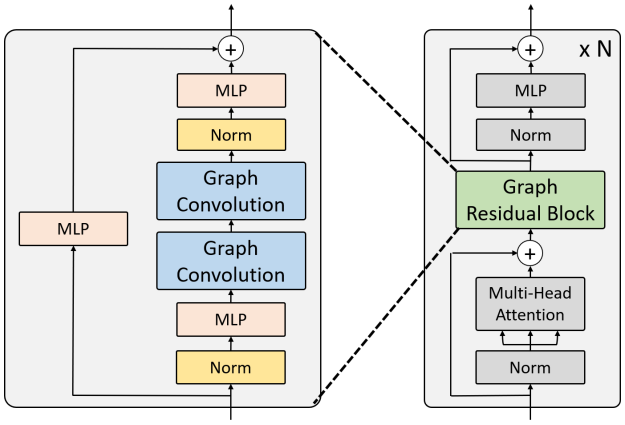


Figure 2: Architecture of a Graphormer Encoder. We propose a graph-convolution-reinforced transformer encoder to capture both global and local interactions for 3D human mesh reconstruction. The encoder consists of a stack of $N = 4$ identical blocks.

methods for reconstructing human mesh in several datasets. Additionally, we offer ablation studies on various model design options to incorporate graph convolutions and self-attention into a transformer encoder.

The main contributions of this paper include

- We present a graph-convolution-reinforced transformer called Mesh Graphormer to model both local and global interactions for the 3D reconstruction of human pose and mesh.
- Mesh Graphormer allows joints and mesh vertices to freely attend to image grid features to refine the 3D coordinate prediction.
- Mesh Graphormer outperforms previous state-of-the-art methods on Human3.6M, 3DPW, and FreiHAND datasets.

2. Related Works

Human mesh reconstruction can generally be divided into parametric and non-parametric approaches. The vast majority of previous work [19, 14, 34, 7, 18, 21, 31] uses a parametric human model such as SMPL [24] and focuses on using the SMPL parameter space as a regression target. Given the pose and shape coefficients, SMPL is robust and practical for creating human mesh. However, as discussed in the literature [20, 29, 5, 42], it remains difficult to estimate accurate coefficients from a single image, and researchers [4, 30, 18, 27] are trying to predict 3D posture, learning with more visual evidence, or adopting the dense correspondence maps [42, 9] to improve the reconstruction.

Instead of regressing the parametric coefficients, non-parametric approaches [5, 26, 20] regress vertices directly from an image. Among the previous studies, the Graph Convolutional Neural Network (GCNN) [17, 5, 20] is one of the most popular options as it is able to model the local interactions between neighboring vertices based on a given adjacency matrix [32, 5, 20, 38]. However, it is less efficient to capture global interactions between the vertices and body joints. To overcome this limitation, transformer-based methods [22] use a self-attention mechanism to freely attend vertices and body joints in the mesh and thereby encode non-local relationships of a human mesh. However, it is less convenient to model local interactions than GNN-based methods [5, 20].

Among the works mentioned above, METRO [22] is the most relevant study to our proposed method. The main difference between METRO and our proposed model is that we are designing a graph-convolution-reinforced transformer encoder for reconstructing human mesh. In addition, we add image grid features as input tokens to the transformer and allow the joints and mesh vertices to attend to grid features.

Transformer architecture is developing rapidly for different applications [6, 35, 33, 15, 10]. One important direction is to improve the expressiveness of a transformer network for better context modeling. Recent studies [40, 8] show that the combination of convolution and self-attention in a transformer encoder is helpful to improve representation learning. However, previous work has mainly focused on language modeling and speech recognition. When it comes to complex data structures such as 3D human mesh, this remains an open problem.

To address these challenges, we investigate how to inject graph convolutions into the transformer encoder blocks to better model both local and global interactions among 3D mesh vertices and body joints.

3. Graphormer Encoder

Figure 2 shows the architecture of our proposed Graphormer encoder. It consists of a stack of $N = 4$ identical blocks. Each block consists of five sub-modules, including a Layer Norm, a Multi-Head Self-Attention module, a Graph Residual Block, a second Layer Norm, and a Multi-Layer Perceptron (MLP) at the end. Our Graphormer encoder has a similar architecture to the traditional transformer encoder [35], but we introduce graph convolution into the network to model fine-grained local interactions. In the following sections, we describe Multi-Head Self-Attention (MHSA) module and Graph Residual Block.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

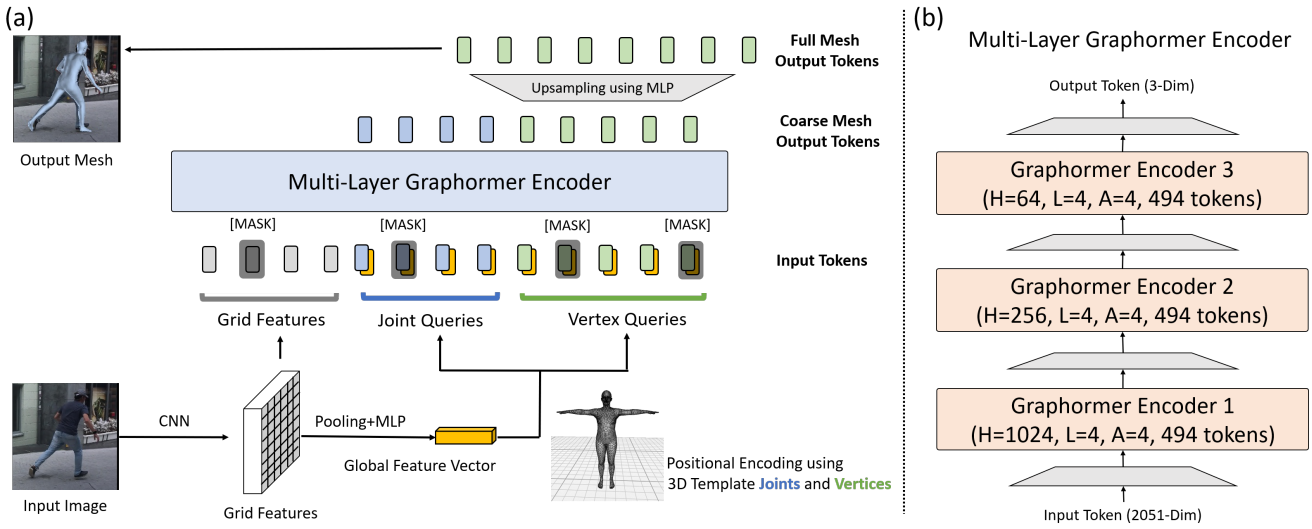


Figure 3: Graphormer for Human Mesh Reconstruction. (a) Our end-to-end mesh regression framework takes an image as input and predicts 3D joints and mesh vertices at the same time. We extract both the grid features and a global feature vector using a pre-trained CNN. The features are then tokenized and fed to a multi-layer Graphormer encoder. Grid features and multi-layer Graphormer encoder are mutually reinforced to reconstruct an accurate human mesh. (b) Our multi-layer Graphormer encoder consists of three Graphormer encoder blocks. All encoders have the same number of input tokens. We gradually reduce the hidden sizes, and output the 3D coordinates of body joints and mesh vertices.

3.1. Multi-Head Self-Attention

We employ the Multi-Head Self-Attention (MHSA) module proposed by Vaswani *et al.* [35], which uses several self-attention functions in parallel to learn contextual representation. Given an input sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n\times d}$, where d is the hidden size. It first projects the input sequence to queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$ by using trainable parameters $\{\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V\} \in \mathbb{R}^{d\times d}$. That is written as

$$\mathbf{Q}, \mathbf{K}, \mathbf{V} = \mathbf{X}\mathbf{W}_Q, \mathbf{X}\mathbf{W}_K, \mathbf{X}\mathbf{W}_V \quad \in \mathbb{R}^{n\times d}. \tag{1}$$

The three feature representations are split into h different subspaces, e.g., $\mathbf{Q} = [\mathbf{Q}^1, \mathbf{Q}^2, \ldots, \mathbf{Q}^h]$ where $\mathbf{Q}^i \in \mathbb{R}^{n\times \frac{d}{h}}$, so that we can perform self-attention for each subspace individually. For each subspace, the output $\mathbf{Y}^h = \{\mathbf{y}_1^h, \mathbf{y}_2^h, \ldots, \mathbf{y}_n^h\}$ are computed by:

$$\mathbf{y}_i^h = \mathbf{Att}(\mathbf{q}_i^h, \mathbf{K}^h)\mathbf{V}^h \quad \in \mathbb{R}^{\frac{d}{h}}, \tag{2}$$

where $\mathbf{Att}(\cdot)$ denotes the attention function [35] that quantifies how semantically relevant a query $\mathbf{q}_i^h$ is to keys $\mathbf{K}^h$ by scaled dot-product and softmax. The output $\mathbf{Y}^h \in \mathbb{R}^{n\times\frac{d}{h}}$ from each subsapce are later concatenated to form the final output $\mathbf{Y} \in \mathbb{R}^{n\times d}$.

3.2. Graph Residual Block

While MHSA is useful for extracting long-range dependencies, it is less efficient at capturing fine-grained local information in complex data structures such as 3D mesh. The Graph Residual Block in our proposed network aims to address this challenge.

Given the contextualized features $\mathbf{Y} \in \mathbb{R}^{n\times d}$ generated by MHSA, we improve the local interactions with the help of graph convolution:

$$\mathbf{Y}' = \mathbf{GraphConv}(\bar{\mathbf{A}}, \mathbf{Y}; \mathbf{W}_G) = \sigma(\bar{\mathbf{A}}\mathbf{Y}\mathbf{W}_G). \tag{3}$$

$\bar{\mathbf{A}} \in \mathbb{R}^{n\times n}$ denotes the adjacency matrix of a graph and $\mathbf{W}_G$ the trainable parameters. $\sigma(\cdot)$ is the activation function that gives the network non-linearity. According to BERT [6], we use the Gaussian Error Linear Unit (GeLU) [11] in this work.

To train with multiple graph convolution layers, we follow the design principle in GraphCMR [20] to build our Graph Residual Block. The network architecture is spiritually similar to [20], but we replace the group normalization [39] with the layer normalization [2] and replace ReLU [28] with GeLU [11]. This is to bring the type of layers in line with the transformer.

Our Graph Residual Block makes it possible to explicitly encode the graph structure within the network and thereby improve spatial locality in the features.

4. Graphormer for Mesh Reconstruction

In this section, we describe how the Graphormer encoder is applied to human pose and mesh reconstruction from a

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

single image.

Figure 3(a) gives an overview of our end-to-end mesh regression framework. An image with a size of 224x224 is used as input and image grid features are extracted. The image features are tokenized as input for a multi-layer Graphormer encoder. In the end, our end-to-end framework predicts 3D coordinates of the mesh vertices and body joints at the same time.

In the following, we describe the image-based architecture with which we extract image grid features. Next, we describe the Multi-Layer Graphormer Encoder for regression of the 3D vertices and body joints. Finally, we discuss important training details.

4.1. CNN and Image Grid Features

In the first part of our model, we use a pre-trained image-based CNN for feature extraction. Previous work [20, 14, 22] extracts a global 2048-Dim image feature vector as a model input. The disadvantage is that a global feature vector does not contain fine-grained local details. This motivates us to add grid features [13] as input tokens and allow joints and mesh vertices to freely attend to all the grid features. As we will show in our experiments, the local information provided by the grid features are effectively leveraged by the graph convolutions in the Graphormer to refine 3D positions of mesh vertices and body joints.

As shown in Figure 3(a), we extract the grid features from the last convolution block in CNN. The grid features are typically $7 \times 7 \times 1024$ in size. They are tokenized to 49 tokens, and each token is a 1024-Dim vector. Similar to [22], we also extract the 2048-Dim image feature vector from the last hidden layer of the CNN and perform positional encoding using the 3D coordinates of each vertex and body joint in a human template mesh. Finally, we apply MLP to make the size of all input tokens consistent. After that, all input tokens have 2051-Dim.

4.2. Multi-Layer Graphormer Encoder

Given the grid features, joint queries, and vertex queries, our multi-layer Graphormer encoder sequentially reduces the dimensions to map the inputs to 3D body joints and mesh vertices at the same time.

As shown in Figure 3(b), our multi-layer Graphormer encoder consists of three encoder blocks. The three encoder blocks have the same number of tokens including 49 grid feature tokens, 14 joint queries, and 431 vertex queries. However, the three encoders have different hidden dimensions. In this work, the hidden dimensions of the three encoders are 1024, 256 and 64, respectively.

Similar to [22], our multi-layer Graphormer encoder processes a coarse mesh with 431 vertices. We use a coarse template mesh for positional encoding, and our Graphormer encoder outputs a coarse mesh. Then, we use a linear projection to sample the coarse mesh up to the original resolution (with 6K vertices in SMPL mesh topology). As explained in the literature [20, 22], learning a coarse mesh followed by upsampling is helpful to avoid redundancies in the original mesh and makes training more efficient.

4.3. Training Details

Following METRO [22], we train our model by applying a loss of L_1 in addition to the Graphormer outputs. In our training, we also use Masked Vertex Modeling [22] to improve the robustness of our model. To be specific, we apply L_1 loss to 3D mesh vertices and body joints. We also apply L_1 loss to 2D projected body joints to improve the alignment between the image and the reconstructed mesh. In addition, we apply intermediate supervision on the coarse mesh to accelerate convergence.

We train Graphormers with the Adam optimizer [16]. We use an initial learning rate of 1×10^{-4} for both Graphormer and CNN backbones. We train our model for 200 epochs and lower the learning rate by a factor of 10 after 100 epochs. All Graphormer weights are randomly initialized, and the CNN backbone is initialized with ImageNet pre-trained weights. Following [22], we report on results with the HRNet backbone [37].

5. Experiments

In this section, we first discuss the datasets we use in our training. We then compare our method with previous state-of-the-art methods. Finally, we provide in-depth ablation studies on our model architectures.

5.1. Datasets

We conduct extensive training using publicly available datasets, including Human3.6M [12], MuCo-3DHP [25], UP-3D [21], COCO [23], MPII [1]. Please note that the 3D mesh training data from Human3.6M [12] is not available due to licensing issues. Therefore, we use the pseudo 3D mesh training data from [5, 26]. We follow the general setting where subjects S1, S5, S6, S7 and S8 are used in training, and test subjects S9 and S11. We present all results using the P2 protocol [14, 19]. To get a fair comparison with earlier state-of-the-art [18, 22], we also use 3DPW [36] training data for 3DPW experiments.

We also apply our method to reconstructing 3D hand. We conduct training on FreiHAND [44] and evaluate using its online server. Please note that we use test-time augmentation for FreiHAND experiment.

5.2. Main Results

We compare our method with previous approaches to reconstruct human mesh on Human3.6M and 3DPW datasets. In Table 1, we can see that our method outperforms the previous state-of-the-art for both datasets. This includes both

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

Figure 4: Qualitative results of our method on different datasets. The top two rows are the results on Human3.6M, and the bottom two rows are the results on 3DPW.

Method	3DPW			Human3.6M	
	MPVE ↓	MPJPE ↓	PA-MPJPE ↓	MPJPE ↓	PA-MPJPE ↓
HMR [14]	–	–	81.3	88.0	56.8
GraphCMR [20]	–	–	70.2	–	50.1
SPIN [19]	116.4	–	59.2	–	41.1
Pose2Mesh [5]	–	89.2	58.9	64.9	47.0
I2LMeshNet [26]	–	93.2	57.7	55.7	41.1
VIBE [18]	99.1	82.0	51.9	65.6	41.4
METRO [22]	88.2	77.1	47.9	54.0	36.7
Mesh Graphormer	**87.7**	**74.7**	**45.6**	**51.2**	**34.5**

Table 1: Performance comparison with the previous state-of-the-art methods on 3DPW and Human3.6M datasets.

GCNN-based models [20, 5] and transformer-based methods [22]. It suggests that Graphormer models both local and global interactions better and achieves more accurate mesh reconstruction.

Figure 4 shows the qualitative results of our method on Human3.6M and 3DPW datasets. We see that our method is robust to challenging poses and noisy backgrounds.

We also apply our method to 3D hand reconstruction and compare it with other modern approaches [22, 5, 26, 44] on FreiHAND dataset. In Table 2, we see that our method works much better than the prior art. This demonstrates the generalization capability of our method for other objects.

Figure 5 shows the qualitative results of our method on FreiHAND test set. We can see that Graphormer can be

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

Figure 5: Qualitative results of our method on FrieHAND test set. Graphormer generalizes to 3D hand reconstruction even if the gestures are novel and complicated.

Method	PA-MPVPE ↓	PA-MPJPE ↓	F@5 mm ↑	F@15 mm ↑
Hasson et al [14]	13.2	–	0.436	0.908
Boukhayma et al. [20]	13.0	–	0.435	0.898
FreiHAND [19]	10.7	–	0.529	0.935
Pose2Mesh [5]	7.8	7.7	0.674	0.969
I2LMeshNet [26]	7.6	7.4	0.681	0.973
METRO [22]	6.7	6.8	0.717	0.981
Mesh Graphormer	**5.9**	**6.0**	**0.764**	**0.986**

Table 2: Performance comparison with the previous state-of-the-art methods on FreiHAND dataset.

used to reconstruct a 3D hand even if the target gesture is novel and complicated.

5.3. Ablation Study

We conduct a large-scale ablation study on Human3.6M [12] to investigate our model capability. We evaluate our model using the 3D pose accuracy, and report the accuracy using PA-MPJPE metric [43, 14, 19, 20].

Effectiveness of Grid Features: In our first ablation study, we are interested in the effect of adding image grid features to the transformer encoder. We have implemented a baseline model [22] that uses the image grid features as inputs. Table 3 shows that the addition of image grid features improves the reconstruction performance by a clear margin. This indicates that using a single global feature vector is one of the performance bottlenecks in existing techniques.

The next question one may ask is what feature maps are important for training a transformer encoder. We perform

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

Method	PA-MPJPE
Transformer [22]	36.7
Transformer + Grid Features	**35.9**

Table 3: Analysis of the effective of using the image grid features in a transformer framework.

Grid Features	A	B	C	D	E	F
PA-MPJPE	37.1	37.7	36.3	38.1	36.0	**35.9**

Table 4: Performance comparison of the use of different image grid features from the HRNet.

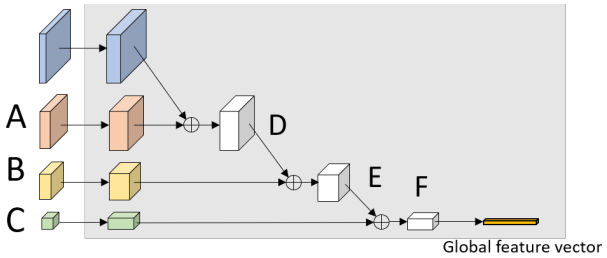


Figure 6: Illustration of different grid features we extracted from HRNet [37] for ablation study.

Encoder1	Encoder2	Encoder3	PA-MPJPE
✗	✗	✗	35.9
✓	✗	✗	36.7
✗	✓	✗	36.2
✗	✗	✓	**35.1**
✓	✓	✓	36.0

Table 5: Ablation study of Graphormer Block for three encoders used for human mesh regression. ✓: Add graph convolution into the specified transformer encoder; ✗: No graph convolution in the specified transformer encoder.

analysis for a variety of feature maps in an HRNet [37]. Figure 6 shows the six feature maps we have selected for experiments, and Table 4 shows the performance comparison. Our finding is that the latest feature map (labeled F in Figure 6) works better than others. The results suggest that the feature pyramid is helpful for improving the performance of a transformer encoder.

Effectiveness of Adding Graph Convolution: An important question is whether adding a graph convolution to the transformer is useful. In this experiment, we investigate the effect of graph convolution and how graph convolution and transformer can be combined.

Since our proposed framework contains three encoder blocks, we examine the effect of graph convolution by gradually adding a graph convolution layer to each encoder and comparing the performance. In Table 5, the first row corresponds to the baseline that does not use any graph convolution in the network. The rest of the rows show the results of adding a graph convolution layer to different encoder blocks. The results show some interesting observations: (*i*) Adding a graph convolution to Encoder1 or Encoder2 does not improve performance. (*ii*) Adding the graph convolution to Encoder3 improves 0.9 PA-MPJPE. The results suggest that the lower layers focus on the global interactions of the mesh vertices to model the posture of the human body, while the upper layers pay more attention to the local interactions for better shape reconstruction.

Model Architecture	PA-MPJPE
Graph Conv. and MHSA in parallel	36.4
Graph Conv. before MHSA	35.6
Graph Conv. after MHSA	**35.1**

Table 6: Performance comparison between different design options of the architecture of Graphormer Block.

Design Choices	PA-MPJPE
Grid Features	35.9
Grid Features + Graph Conv.	35.1
Grid Features + Graph Res. Block	**34.5**

Table 7: Ablation study of the basic graph convolution and Graph Residual Block.

Analysis of Encoder Architecture: We further examine the network architecture in an encoder block. That is, we investigate the relationship between the MHSA module (Multi-Head Self-Attention) and the Graph Convolution module by using three different designs: (*i*) We use a graph convolution layer and MHSA in parallel, similar to [40]. (*ii*) We first use a graph convolution layer and then MHSA. (*iii*) We first use MHSA and then a graph convolution layer. Please refer to the supplementary material for a graphical illustration of the three architecture designs. In Table 6, we can see that adding a graph convolution layer directly after MHSA works much better than other design options.

Effectiveness of Adding Graph Residual Block: In the previous sections, we explored the use of a basic graph convolution layer in a transformer encoder. Here we extend our experiments to Graph Residual Block. In particular, we replace a graph convolution layer with a Graph Residual Block to improve model capacity. In Table 7, we see that a

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

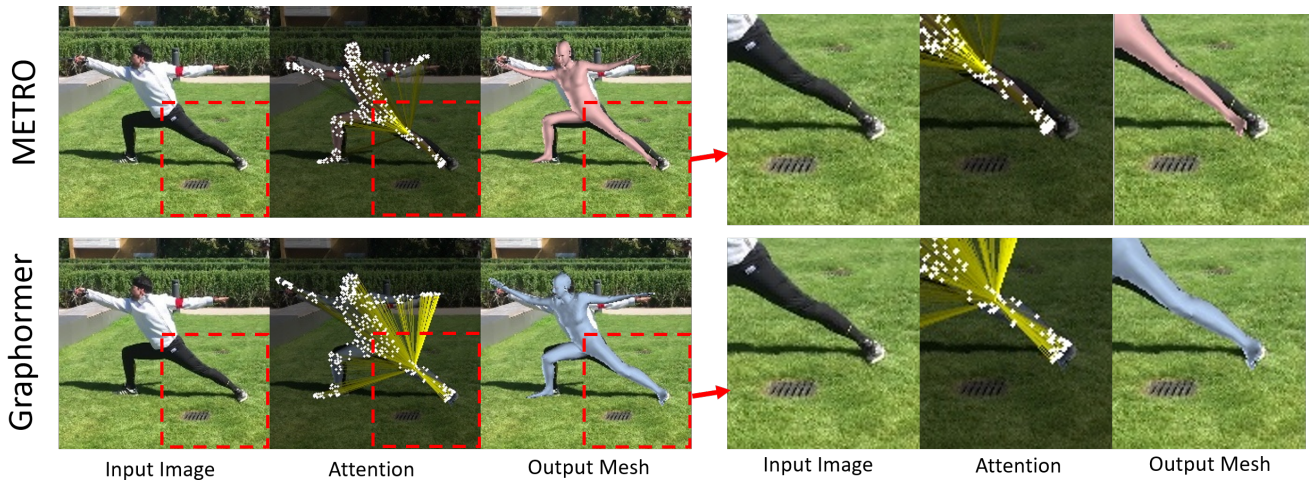


Figure 7: Qualitative comparison. The top row (with pink mesh) shows the results of a transformer baseline [22] which has no graph convolutions. The bottom row (with blue mesh) shows our results with Graphormer. The attention visualization shows the interactions between the left knee and all other vertices, and the brighter color indicates stronger interactions. As can be seen, Graphormer is helpful to model both global and local interactions among vertices and body joints, and achieves a more accurate reconstruction. Please note the color of each attention map is normalized with respect to the maximum attention. Figure 6 in the supplementary material shows the attention maps without color normalization.

Grid Features	Graph Conv.	Graph Res.	PA-MPJPE
✗	✗	✗	36.7
✗	✓	✗	36.7
✗	✓	✓	36.6
✓	✗	✗	35.9
✓	✓	✗	35.1
✓	✓	✓	**34.5**

Table 8: Ablation study of different combinations of image grid features and graph convolutions.

Graph Residual Block works better than a basic graph convolution layer.

Relationship between Grid Features and Graph Convolution: An interesting question is what happens when we use Graphormer Encoder, but not grid features. To answer the question, Table 8 shows the comparisons.

First of all, the first row of Table 8 corresponds to the baseline which has no grid features or graph convolution in the transformer. Compared to the fourth row of Table 8, we see that we can achieve an improvement of 0.8 PA-MPJPE if we only enable grid features. Next, if we enable graph convolution alone, as shown in the third row of Table 8, only 0.1 PA-MPJPE is improved. Finally, as shown in the bottom row of Table 8, when we enable both grid features and graph convolution, it eventually improves PA-MPJPE by 2.2 which is much greater than the sum of the two individual improvements $(0.1 + 0.8)$. It shows that grid features and graph convolutions mutually reinforce each other, which leads to an further increase in performance.

Visualization of Local Interactions: We further study the effect of Graphormer in learning both global and local interactions among body joints and mesh vertices. We extract the attention maps from the last layer of our last encoder (i.e., Encoder3), and compute the average attentions of all the attention heads. We compare the self-attentions of Graphormer with that of the existing approach [22]. Figure 7 shows the qualitative comparison. We can see in the top row of Figure 7, previous approach [22] fails to model the interactions between the left knee and left toes. In contrast, as shown in the bottom row of Figure 7, Graphormer is able to model both global and local interactions, especially those between the left knee and the left toes. As a result, Graphormer reconstructs a more favorable shape compared to prior works.

6. Conclusion

We introduced Mesh Graphormer, a new transformer architecture that incorporates graph convolutions and self-attentions for reconstructing human pose and mesh from a single image. We explored various model design options and demonstrated that both graph convolutions and grid features are helpful for improving the performance of the transformer. Experimental results show that our method generates new state-of-the-art performance on Human3.6M, 3DPW, and FreiHAND datasets.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, 2014. 4

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[3] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A2-nets: Double attention networks. *arXiv preprint arXiv:1810.11579*, 2018. 1

[4] Hongsuk Choi, Gyeongsik Moon, and Kyoung Mu Lee. Beyond static features for temporally consistent 3d human pose and shape from a video. *arXiv e-prints*, pages arXiv–2011, 2020. 2

[5] 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, 4, 5, 6

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 2, 3

[7] Peng Guan, Alexander Weiss, Alexandru O Balan, and Michael J Black. Estimating human shape and pose from a single image. In *ICCV*, 2009. 2

[8] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. In *INTERSPEECH*, 2020. 1, 2

[9] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. In *CVPR*, 2018. 2

[10] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020. 2

[11] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 3

[12] 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. 4, 6

[13] Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *CVPR*, 2020. 4

[14] Angjoo Kanazawa, Michael J Black, David W Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 2, 4, 5, 6

[15] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021. 2

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[17] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 2

[18] Muhammed Kocabas, Nikos Athanasiou, and Michael J. Black. Vibe: Video inference for human body pose and shape estimation. In *CVPR*, 2020. 2, 4, 5

[19] 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, 4, 5, 6

[20] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 1, 2, 3, 4, 5, 6

[21] 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, 4

[22] 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

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 4

[24] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6):248, 2015. 2

[25] 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*, 2018. 4

[26] 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. 2, 4, 5, 6

[27] Gyeongsik Moon and Kyoung Mu Lee. Pose2pose: 3d positional pose-guided 3d rotational pose prediction for expressive 3d human pose and mesh estimation. *arXiv preprint arXiv:2011.11534*, 2020. 2

[28] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 3

[29] Mohamed Omran, Christoph Lassner, Gerard Pons-Moll, Peter Gehler, and Bernt Schiele. Neural body fitting: Unifying deep learning and model based human pose and shape estimation. In *3DV*, 2018. 2

[30] Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Human mesh recovery from multiple shots. *arXiv preprint arXiv:2012.09843*, 2020. 2

[31] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3d human pose and shape from a single color image. In *CVPR*, 2018. 2

[32] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3d faces using convolutional mesh autoencoders. In *ECCV*, 2018. 2

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.

[33] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020. 2

[34] Hsiao-Yu Tung, Hsiao-Wei Tung, Ersin Yumer, and Katerina Fragkiadaki. Self-supervised learning of motion capture. In *NeurIPS*, 2017. 2

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 3

[36] 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. 4

[37] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 4, 7

[38] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 2

[39] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018. 3

[40] Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. Lite transformer with long-short range attention. In *ICLR*, 2020. 1, 2, 7

[41] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. In *ICLR*, 2018. 1

[42] Hongwen Zhang, Jie Cao, Guo Lu, Wanli Ouyang, and Zhenan Sun. Learning 3d human shape and pose from dense body parts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 2

[43] 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*, 2018. 6

[44] Christian Zimmermann, Duygu Ceylan, Jimei Yang, Bryan Russell, Max Argus, and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *ICCV*, 2019. 4, 5

Authorized licensed use limited to: ShanghaiTech University. Downloaded on October 19,2026 at 14:26:06 UTC from IEEE Xplore. Restrictions apply.