AniFormer: Data-driven 3D Animation with Transformer

Haoyu Chen[1]
chen.haoyu@oulu.fi

Hao Tang[2]
hao.tang@vision.ee.ethz.ch

Nicu Sebe[3]
nicu.sebe@unitn.it

Guoying Zhao*[1]
guoying.zhao@oulu.fi

[1] CMVS
    University of Oulu

[2] Computer Vision Lab
    ETH Zurich

[3] DISI
    University of Trento

Abstract

We present a novel task, i.e., animating a target 3D object through the motion of a raw driving sequence. In previous works, extra auxiliary correlations between source and target meshes or intermedia factors are inevitable to capture the motions in the driving sequences. Instead, we introduce AniFormer, a novel Transformer-based architecture, that generates animated 3D sequences by directly taking the raw driving sequences and arbitrary same-type target meshes as inputs. Specifically, we customize the Transformer architecture for 3D animation that generates mesh sequences by integrating styles from target meshes and motions from the driving meshes. Besides, instead of the conventional single regression head in the vanilla Transformer, AniFormer generates multiple frames as outputs to preserve the sequential consistency of the generated meshes. To achieve this, we carefully design a pair of regression constraints, i.e., motion and appearance constraints, that can provide strong regularization on the generated mesh sequences. Our AniFormer achieves high-fidelity, realistic, temporally coherent animated results and outperforms compared start-of-the-art methods on benchmarks of diverse categories. Code is available: https://github.com/mikecheninoulu/AniFormer.

1   Introduction

Directly animating 3D objects via dynamics from raw driving sequences is a desirable task. Apart from purely scientific interests, learning to automatically synthesize high-fidelity 3D animation has a wide range of applications. For example, in the film industry, an automatic 3D animating model finds use indirectly transferring motions to various 3D characters without any need of manual designing the kinetic procedures. Using a learned 3D animating model in virtual reality (VR) scenarios, one can generate realistic animated 3D avatars without explicitly specifying the geometry properties and dynamics, which would be cumbersome but inevitable when using a standard graphics rendering engine.

* Indicates the corresponding author

Driving sequences
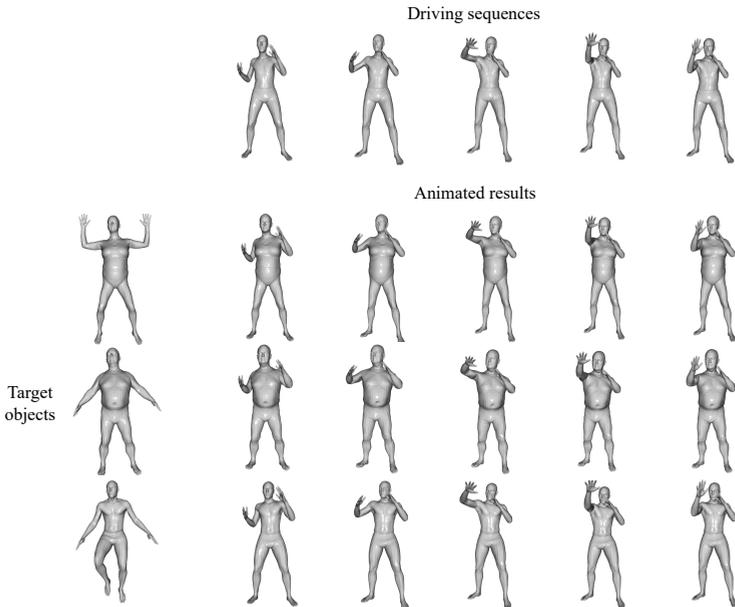
Animated results

Target
objects

Figure 1: Example produced animations from AniFormer: a driving sequence (top), different target objects (left), and animated resulting sequences (bottom). The samples are from DFAUST dataset [5]. The motion is directly learnt from raw meshes of the driving sequences without extra manual interventions.

There are various existing works related to 3D animation problem, including 3D mesh deformation [1, 7, 10, 12, 24, 26, 51, 56], 3D motion estimation [9, 17, 18], and 3D motion synthesis [21, 22]. In this paper, we study a new form, i.e., data-driven 3D animation. At the core, we aim to learn a mapping function that can animate a target 3D object driven by the motion from a raw mesh sequence. To the best of our knowledge, a general-purpose solution to raw-data driven 3D animation has not yet been explored by prior work, although its similar tasks, the 3D deformation and style transfer problems have been intensively researched that could also achieve 3D animation by processing single mesh frame by frame [24, 26, 51, 56]. Our method is inspired by previous 2D video synthesis works [25, 52].

We cast the raw data-driven 3D animation as a continuous deformation problem, where the goal is to train a model such that the animated object has the desired dynamics from driving sequences while preserving the original appearance information. To this end, we utilize the strength of sequential attention mechanism from Transformers to learn given paired input and output 3D sequences. Specifically, we design a new Transformer structure that can jointly learn the style information from target meshes and dynamics from the driving sequences. To handle sequential mesh processing, we replace the classical regression head with various regression constraints for appearance and motion consistency over the whole generated sequences. Our method can learn to generate high-fidelity, realistic, and temporally coherent 3D human motions. Conditioning on the same driving sequences, our model can produce identical 3D motions for various seen and unseen target 3D human meshes. Moreover, we extend our method to different objects such as animals.

In summary, our main contributions are:
• To our knowledge, our work is the first that can achieve end-to-end 3D animation that learns the 3D motion directly from raw data sequences without any extra intervention.

- We propose the AniFormer, a Transformer-based architecture for 3D animation, which integrates both styles and motions from source meshes. It also discards the convention regression head with a single-vector representation and directly generates multiple frames as outputs to better learn the sequential consistency.

- Two regression constraints are carefully designed to synthesize realistic animated results, i.e., the natural appearance consistency in the driving sequences, and motion consistency between the driving sequence and synthesized sequence.

- Experimental results on four different datasets demonstrate that the proposed method achieves the state-of-the-art performance. Moreover, we further represent the generalization capability by extending our framework to diverse categories, such as various SMPL human meshes [4], non-SMPL human meshes, and animals, and showing it is a strong baseline for data-driven 3D animation.

2 Related Work

3D Mesh Deformation aims to generate a new 3D shape with a given pair of source and target shapes, which can be seen as a frame-level task of 3D animation. Towards 3D mesh deformation, previous works demand re-enforcing the correspondence between source and target meshes. For example, some disentanglement-based methods like [10, 36] use the shape correspondences between different pose meshes from the same body to decompose shape and pose factors. SNARF [7] uses differentiable forward skinning for animating non-rigid neural implicit shapes, which can be extended to unseen and complex shapes, but bone transformations between the target meshes and the desired pose are critical in their settings. SCANimate [24] learns a pose-aware parametric clothed human model by introducing an intermediate canonicalization procedure to convert the posed space into canonical space, however, its animation is achieved by manipulating SMPL pose parameters but not driven by raw data sources. As mentioned, all these existing deformation methods need extra intervened information which is non-trivial, and time-consuming to obtain. In contrast, our framework does not require any additional auxiliary information and can animate unseen target meshes with completely different source mesh sequences. Moreover, unlike existing works [6], we are the first to achieve the 3D animation with long raw sequential driven data.

3D Motion with Sequential Dependencies. Generating 3D mesh sequences via motion information has been intensively researched from different aspects. For 3D mesh pose and motion estimation, VIBE [17] and TCMR [9] extract static features from the sequential frames into a temporal feature by using a bidirectional gated recurrent unit (GRU) [8]. To synthesize 3D body meshes with high fidelity, they use an SMPL [4] parameter regressor to provide strong constraints on the synthesized meshes. For 3D motion synthesis, ACTOR [22] and Action2Motion [13] learn generators based on [16] that achieve action-conditioned generation of realistic and diverse 3D human mesh sequences. Again, an SMPL joint regressor is introduced to ACTOR to ensure the realistic generated motions and Action2Motion post-renders the skeleton joints to SMPL models to obtain better visualized results. Compared to the above works, we have two core differences: 1) both the appearance of target meshes and desired motions can be handled in the generated 3D sequences, and 2) our framework directly processes the raw mesh sequences without any manual intervention (e.g., SMPL parameters or skeleton joints) in an end-to-end manner.

Transformer for Temporal Modeling. Transformer architecture was first proposed by [30] and raised emerging research interests in vision tasks with promising performances [14].

To retain positional information of sequential dependencies, a trainable linear projection layer to embed each input patch to a high dimensional feature is always applied in, such as pose estimation [35], motion synthesis [22], facial analysis [34], and video analysis [11]. However, we argue that, the 3D animation task is different from the above tasks, i.e., its appearance consistency and motion continuity is crucial information that can be utilized as good regression constraints. Meanwhile, the regression head in the vanilla Transformer is in a single-vector representation which is hard to preserve the sequential consistency. Thus, we propose a novel architecture, the AniFormer, based solely on the existing generating flows, dispensing with the conventional regression head entirely for the 3D animation task.

3   Data-Driven 3D Animation

Similar to the work of [25, 32] for the 2D task, we are interested in animating a target 3D object based on the motion of a driving sequence. This section first presents a general introduction of the whole AniFormer framework. Then, we will demonstrate how to use AniFormer to achieve 3D animation learning with proposed constraints.

3.1   AniFormer: Transformer-Based Network for 3D Animation

In this proposed animation task, the latent representations of 3D models should be large enough to preserve the detailed surface geometry of the target mesh. However, this will make the number of learnable parameters of conventional recurrent temporal models like RNNs grow quadratically with the memory size, making them extremely large and almost non-trainable. Thus we propose to implement Transformer framework with simple temporal embedding for saving memory. An overview of the AniFormer is depicted in Fig. 2. Our AniFormer consists of three key components, i.e., a 3D mesh feature extractor, an AniFormer encoder, and regression constraints following the generated data flow.

3D Feature Extractor. Given a 3D mesh driving sequence of T frames $M_1, \cdots, M_T$, where $M \in \mathbb{R}^{V \times 3}$ and V stands for vertex number such as 6,890, a feature extractor based on the classical PointNet [23] extracts a static 3D feature per frame, resulting a sequence of latent embedding vector $Z_1, \cdots, Z_T$, where $Z \in \mathbb{R}^{C \times V}$ and $C = 1,024$ in practice. The network weights of the feature extractor are shared for all frames. From the extracted static features of all input 3D meshes, we retain the current frame's temporal information of the sequence via the classical positional embeddings used in Transformer networks. The procedure can be formulated as:

$$Z = \{Z_1; Z_2; \cdots; Z_T\} + E_t. \tag{1}$$

After summing with the temporal embedding $E_t \in \mathbb{R}^{T \times C \times V}$, the resulting feature vector Z is sent to the AniFormer encoder. In our case, original batch normalization layers are replaced with Instance Normalization [29] layers to preserve the instance style [15, 19, 27, 28].

AniFormer Encoder. The core problem of casting Transformer networks for animating target 3D meshes is how to jointly learn the correlations between the given target mesh and driving mesh sequences, since Transformer networks are all designed for self-attention mechanism with single-source inputs. To this end, inspired by style transfer works [15, 19, 31], we design the novel AniFormer encoder by integrating the style information from the target 3D mesh into the self-attention mechanism. Specifically, as shown in Fig. 2 right part, we feed the embedded latent embedding vector of the pose meshes as Z into different 1D convolution layers to generate the normalized representations $\mathbf{q}, \mathbf{k}, \mathbf{v}$ for the standard multihead
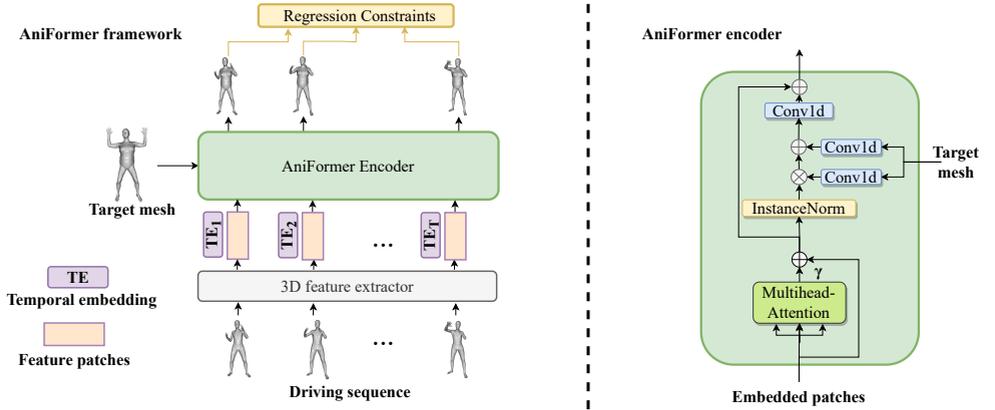
Figure 2: Model overview. We process the driving sequential meshes into patches, add temporal embeddings, and feed the resulting sequence of vectors to our AniFormer encoder. AniFormer encoder can adaptively generate deformed target meshes with given pose meshes via attention mechanism. The illustration was inspired by ViT [11].

self-attention [30]. Then, the attention weights $A_{i,j}$ based on the pairwise geometry between $\mathbf{q}$ and $\mathbf{k}$ is given with the following formula:

$$\mathbf{A}_{i,j} = \frac{exp(\mathbf{q}_i \mathbf{k}_j)}{\sum_{i=1}^{n} exp(\mathbf{q}_i \mathbf{k}_j)}. \qquad (2)$$

After this, a matrix multiplication between v and the transpose of $\mathbf{A}$ is conducted. Finally, we weigh the result with a scale parameter γ and conduct an element-wise sum operation with the original latent embedding Z to obtain the refined latent embedding Z',

$$Z' = \gamma \sum_{i=1}^{n} (\mathbf{A}_{i,j} \mathbf{v}_i) + Z, \qquad (3)$$

where γ is a trainable parameter and initialized as 0. After obtaining the latent embedding vector Z', instead of stacking MLP blocks to the Z' as a vanilla Transformer does, the target mesh M_{target} as an external input is fed into two different 1D convolution layers to produce the modulation parameters which are multiplied and added to the Z', as shown in Fig. 2 right part. In such an explicit-imposing way, our AniFormer encoder can capture sequential motion information and producing a more robust animation of the target mesh. More detailed network architecture and implementation are illustrated in Supplementary Materials.

Note that, different than previous Transformer-based networks [11, 30, 33, 35], our Ani-Former could not only compute the attention weights on the embedded patches but also could fully preserve the local geometric details of the target mesh. Besides, by implementing the self-attention mechanism, the AniFormer can take pure latent feature vectors as inputs to drive the target meshes, while in the work of NPT [51], extra concatenation operators are needed for better deformation. Most importantly, our AniFormer is designed for sequential 3D mesh data processing which has never been attempted in those works.

From Regression Head to Regression Constraints. In almost all the existing Transformer-based networks, a regression head with an MLP block and one linear layer is used to take the average in the sequential dimension and produce the final output, such as image classification, pose estimation. However, we argue that this simple regression head might not

apply to the task of 3D animation (also see more experimental results in Section 4), because a regression head with single output cannot provide constraints of appearance and motion consistency over continuous frames, which is crucial for high fidelity 3D animation. To this end, we discard the conventional regression head and enable AniFormer to generate multiple frames by enforcing carefully designed regression constraints over those frames as introduced in the following section.

3.2   Imposing Regression Constraints

Our AniFormer is based on Transformer architectures and allows end-to-end training with raw mesh sequences as inputs. However, directly inheriting the regression head in the vanilla Transformer will lead to a degenerated performance since it aggregates all the sequential information on one target without consistency constraints along the temporal dimension. To this end, we discard the regression head and design two regression constraints to ensure that the generated sequences have both motion consistency to the driving sequences and appearance consistency to the target mesh.

Constraint on Motion Consistency. During training, we can constrain the generated meshes to have the identical kinetic motion using a simple $L1$ loss on the driving mesh sequence. However, due to the geometric gap between the target mesh and driving meshes, their motion trajectories are not identical. Thus, directly implementing the $L1$ loss would cause a geometric distortion problem and not lead to our expected results. To this end, we design a normalized regulation to preserve the motion consistency:

$$\mathcal{L}_m = \frac{1}{T-2} \sum_{t=3}^{T} \left(\frac{\|M_{t-1} - M_{t-2}\|_2^2}{\|G_{t-1} - G_{t-2}\|_2^2} - \frac{\|M_t - M_{t-1}\|_2^2}{\|G_t - G_{t-1}\|_2^2} \right), \tag{4}$$

where M stands for generated sequential meshes and G stands for the standard motion mesh from the driving sequences. As mentioned, the absolute motion trajectory of target meshes and driving meshes are not equal due to their different geometric proportions. But using our designed Eq. (4), we can proportionally compare the kinetic trajectories and achieve an efficient constraint on the motion of the generated 3D animations.

Constraint on Appearance Consistency. We introduce an appearance regularisation term as a constraint, which penalises the local distortion between the generated meshes and given target mesh. This regularization is inspired by the edge loss from [12] that can enforce the generated mesh surface to be tight, resulting in a smooth surface with local triangles identical to target meshes. Specifically, we generalize the edge loss into sequential settings, let $\mathcal{N}(\mathbf{p})$ denotes the neighbor vertices of vertex $\mathbf{p}$ and V is the total vertex number of the mesh, the appearance regularization can be defined as below:

$$\mathcal{L}_a = \frac{1}{T} \frac{1}{V} \sum_{t=1}^{T} \sum_{\mathbf{p}} \sum_{\mathbf{u} \in \mathcal{N}(\mathbf{p})} \left(\|\mathbf{p}_M^t - \mathbf{u}_M^t\|_2^2 - \|\mathbf{p}_N^t - \mathbf{u}_N^t\|_2^2 \right), \tag{5}$$

where M and N stand for the generated sequential meshes and the target mesh, respectively.
Full Objective Function. At last, we define the full objective function as below:

$$\mathcal{L}_{full} = \lambda_r \mathcal{L}_r + \lambda_m \mathcal{L}_m + \lambda_a \mathcal{L}_a, \tag{6}$$

where $\mathcal{L}_r$, $\mathcal{L}_m$, and $\mathcal{L}_a$ are the three used losses, including reconstruction loss, motion loss, and appearance loss. λ is the corresponding weight of each loss. In Eq. (6), reconstruction loss $\mathcal{L}_r$ is a simple point-wise $L2$ distance commonly used in previous works [10, 51, 56].
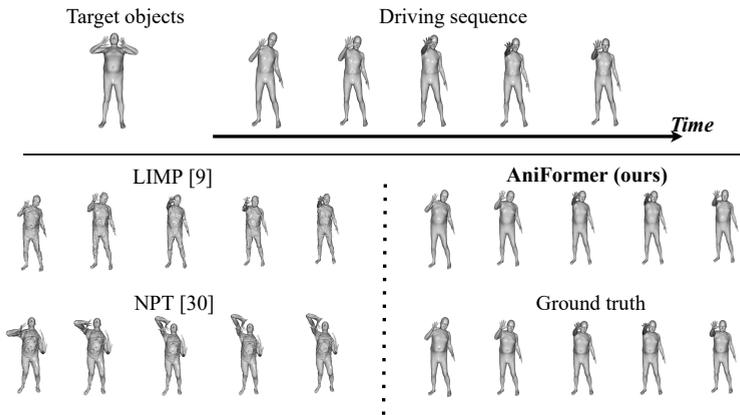
Figure 3: Qualitatively comparison with the state-of-the-art methods on unseen motions from the DFAUST dataset. Given the long driving sequence, compared methods process the deformation frame by frame while AniFormer takes several frames as inputs.
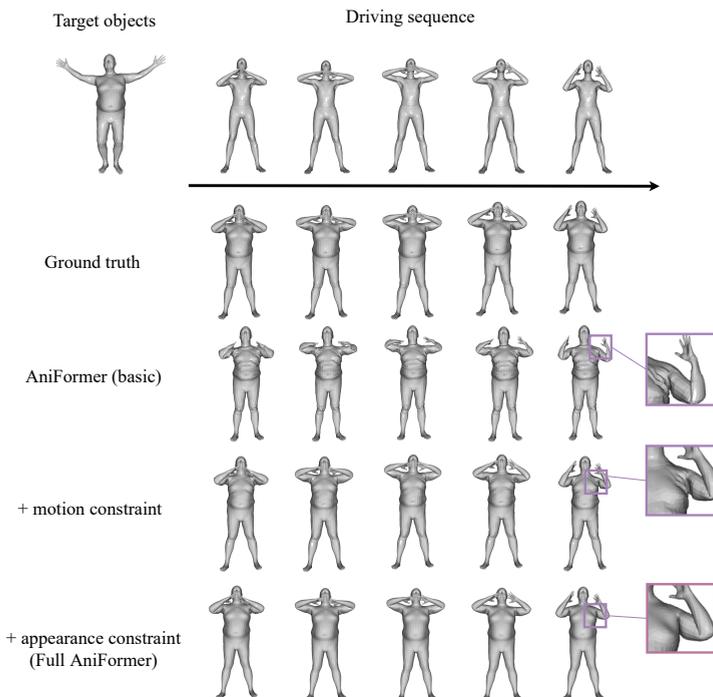

Figure 4: Visualized ablation study by progressively enabling each component of our model with last line as the full AniFormer. As shown, the basic AniFormer can already enable the animation while the constraints makes it more plausible with less distortions.

4 Experiments

In this section, we perform an extensive evaluation of proposed AniFormer. In addition to comparing with previous works for evaluating the animating effects, we also analyze the effect of each component in our model.

Datasets. For training the model, we use the motion sequences in the DFAUST [5] dataset.

DFAUST is a large human motion sequence dataset that captures the 4D motion of 10 human subjects with 129 different body motions, such as "punching", "chicken wings pose", etc. Each motion consists of hundreds of frames. To create training data, we evenly sample 30 frames from each motion as a complete sequence. Then, one motion (driving sequence) and one appearance (target mesh) will be randomly combined as a pair for training. The ground truth is obtained by using SMPL model [4] to synthesize the target animated sequence with the shape and pose parameters provided by the dataset. The mesh vertices are shuffled randomly to ensure the network is vertex-order invariant.

In testing, we use our model to animate the unseen target meshes with motions from given driving sequences. To do so, we create another 8 new target meshes with the SMPL model by sampled from pose and shape parameter spaces. Then we use the driving sequences to animate these 8 new target meshes. The driving sequences are split into two settings, i.e., the seen driving sequences (80 motions) and the unseen driving sequences (20 motions). Unseen driving sequences are not available in the training procedure. Furthermore, to evaluate the generalization ability of our AniFormer, we employ the model to drive the target meshes from other datasets, e.g., FAUST [3] and MG-dataset [2], for animation.

Lastly, we verify AniFormer on animal meshes from the SMAL dataset [57] via domain-specific training. SMAL is a 3D articulated model that can represent various animals including related animal meshes [57]. More details of data processing in Supplementary Materials.

Evaluation Metrics. We introduce the evaluation metrics based on the point-wise mesh Euclidean distance (PMD) which is commonly used as a metric in the 3D pose transfer task. PMD compares the output mesh with the ground truth in a point-wise aligned way. Specifically, we first uniformly sample points from our results and ground truths, then we compute the average PMD over all the frames:

$$PMD = \frac{1}{T}\frac{1}{V}\sum_{t}\sum_{v}\left\|M_{v}^{t} - G_{v}^{t}\right\|_{2}^{2}, \tag{7}$$

where M_{v} and G_{v} are the point pairs from the generated mesh M and ground truth mesh G.

Baselines. We compare the proposed approach to the most recent single mesh deformation approaches. Specifically, we compare with two state-of-the-art methods, i.e., NPT [51] and LIMP [10]. Since both methods could only process a single mesh, we use them to generate long sequences frame by frame. We also compare to vanilla Transformer [1] and its variants. Since so far there is no existing Transformer-based framework for the 3D sequential mesh generation task, we implement them by modifying our AniFormer for a fair comparison. For instance, the regression head version is obtained by replacing the regression constraints of AniFormer back to a classic regression head. Note that, for the compared methods NPT and LIMP, we use the same protocols provided to train the models. For the models in the Ablation study, we trained the compared models within same amount of time to make it fair (also making sure they are converged).

Training and Runtime. Our network receives a sequence of driving meshes of size $V_1 \times 3 \times T$, and a target mesh of size $V_2 \times 3$ where V and T stand for the vertex number and frame number. Note that V_1 might not equal to V_2 because our AniFormer can handle meshes of different sizes. The network is implemented in PyTorch [20] and optimized using the Adam. The batch size is 2. The total number of training epoch is 200. The learning rate is initialized as 5e-5 and reduced at milestones of 80, 120, and 160 with gamma as 0.1. Due to the memory consumption, the input length of the driving sequences is fixed as 3 frames for both training and testing. The total training time is around 50 hours on an Nvidia GPU V100.
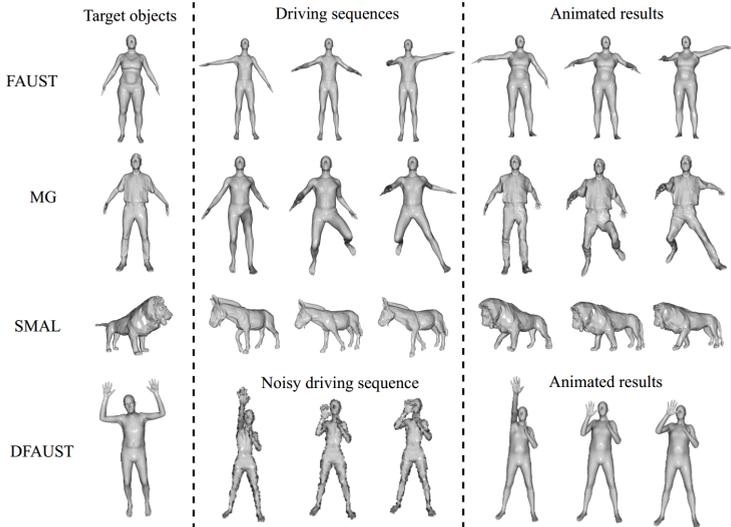
Figure 5: Top three rows: generalization ability of AniFormer on non-SMPL meshes. Last row: Robustness of AniFormer to noisy inputs.

During inference, we conduct the animation of the target mesh by using a sliding window with the size of T (3 frames) with the driving sequence as the driving motion. Inference runs at $\sim$170ms per frame with generated meshes of 6,890 vertices.

Ablation Study. We start the experiments from verifying how much improvement was brought by discarding the regression head. The quantitative and qualitative evaluations are shown in Table 1 and Fig. 4. We start from a vanilla version with a regression head, then remove the regression head, and lastly add regression constraints. As can be seen, the regression head performs even worse than without head version. One intuitive explanation for this is that, the flattening operator in the MLP blocks will damage the mesh vertex order and the single output lack of temporal consistency with neighboring frames. Particularly, bringing the two regression constraints is very helpful in learning, which means enforcing constraints on multiple outputs could be a better solution for 3D sequential Transformers.

State-of-the-Art Comparisons. We provide experimental results by comparing AniFormer with state-of-the-art methods both quantitatively in Table 2 and qualitatively in Fig. 3. For both settings: "Seen motions" and "Unseen motions" from the DFAUST dataset, our Ani-Former significantly outperforms compared state-of-the-art methods with frame-level average PMD ($\times 10^{-4}$) of: 0.124 and 3.95 (AniFormer) vs. 1.37 and 11.31 (NPT). Note that LIMP downsamples the meshes from 6,890 to 2,100 vertices to avoid heavy computation while AniFormer processes the original ones. Thus, to make the comparison, we downsample the ground truth meshes to 2,100 vertices when evaluating LIMP to compute the PMD. Fig. 3 shows the qualitative performances of the compared methods on unseen targets on the DFAUST dataset. As shown, given unseen target meshes, both NPT and LIMP suffer from degeneracy, while AniFormer still can provide promising results close to the ground truth.

Non-SMPL Mesh Animation. AniFormer can be generalized to the animation of non-SMPL meshes. As shown in Fig. 5 rows 1-2, unseen meshes from FAUST [3] and MG [2] datasets can be directly animated by AniFormer without any further finetune. Note that those meshes are not in line with the SMPL model and more challenging (more fine-grained geometry details). Furthermore, we show the results of our model on animating a lion with

Table 1: Ablation study on regression head of the Transformer in our AniFormer.

Method	PMD $\downarrow$ ($\times 10^{-4}$)
Regression Head (Vanilla)	15.31
Without Head	3.84
Motion Constraint	1.37
Appearance Constraint	0.31
Full Constraints (AniFormer)	**0.12**

Table 2: Comparison with the state-of-the-art methods on the DFAUST datasets. We compute frame-level average point-wise mesh Euclidean distance (PMD) as the metric. Our AniFormer outperforms other compared methods for more than one order of magnitude.

Method	PMD$\downarrow$ ($\times 10^{-4}$)	
	Seen Motions	Unseen Motions
LIMP [10]	5.44	7.11
NPT-MP [51]	4.51	6.69
NPT [51]	4.37	5.31
AniFormer (Ours)	**0.12**	**3.95**

the motion of a horse on row 3 of Fig. 5, by training on the SMAL dataset, showing the capacity of our AniFormer in different domains.

Robustness to Noise. Furthermore, we evaluate the robustness of AniFormer by adding uniform noise to the vertices of driving meshes. In the last row of Fig. 5, we can see that AniFormer is insensitive to the noise and can robustly extract the information from driving sequences even with the challenging poses.

Limitations. Lastly, we discuss the limitations of AniFormer. Due to the need to preserve the geometric details of the target models as well as model the temporal information, the computational complexity becomes a main issue of AniFormer. Besides, since AniFormer directly uses the motion from the driving sequence to animate the target objects and the driving data from DFAUST is relatively simple, thus the non-linear local deformation of the target objects (plausible subtle muscle moments) has not been explored yet. However, the ability of models to generalize the plausible motions with non-linear local deformation can efficiently improve the animation visual effects which could be future research direction.

5 Conclusion

We propose the AniFormer framework, i.e., a novel Transformer network customized for raw-data driven 3D animation with carefully-designed regression constraints. The AniFormer for the first time reveals the potential of animating 3D target meshes via raw driving mesh sequences and qualitative results on various categories further show that our model provides powerful animation ability by maintaining motions depicted in the driving sequences and preserving appearance from target meshes. Future directions could be unsupervisedly learning the motion, improving the AniFormer into a more compact architecture and automatically generalizing the non-linear motion for the target objects.

References

[1] Tristan Aumentado-Armstrong, Stavros Tsogkas, Allan Jepson, and Sven Dickinson. Geometric disentanglement for generative latent shape models. In *ICCV*, 2019.

[2] Bharat Lal Bhatnagar, Garvita Tiwari, Christian Theobalt, and Gerard Pons-Moll. Multi-garment net: Learning to dress 3d people from images. In *ICCV*, 2019.

[3] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *CVPR*, 2014.

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

[5] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J Black. Dynamic faust: Registering human bodies in motion. In *CVPR*, 2017.

[6] Haoyu Chen, Hao Tang, Henglin Shi, Wei Peng, Sebe Nicu, and Guoying Zhao. Intrinsic-extrinsic preserved gans for unsupervised 3d pose transfer. In *ICCV*, 2021.

[7] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. *ICCV*, 2021.

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[9] Hongsuk Choi, Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Beyond static features for temporally consistent 3d human pose and shape from a video. *CVPR*, 2021.

[10] Luca Cosmo, Antonio Norelli, Oshri Halimi, Ron Kimmel, and Emanuele Rodolà. Limp: Learning latent shape representations with metric preservation priors. *ECCV*, 2020.

[11] 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. *ICLR*, 2021.

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

[13] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *ACM MM*, 2020.

[14] 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.

[15] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, 2013.

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

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

[19] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019.

[20] 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. In *NeurIPS*, 2019.

[21] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *ACM TOG*, 2021.

[22] Mathis Petrovich, Michael J Black, and Gül Varol. Action-conditioned 3d human motion synthesis with transformer vae. *ICCV*, 2021.

[23] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[24] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. SCANimate: Weakly supervised learning of skinned clothed avatar networks. In *CVPR*, 2021.

[25] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *NeurIPS*, 2019.

[26] Robert W Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM TOG*, 2004.

[27] Hao Tang, Song Bai, and Nicu Sebe. Dual attention gans for semantic image synthesis. In *ACM MM*, 2020.

[28] Hao Tang, Dan Xu, Yan Yan, Philip HS Torr, and Nicu Sebe. Local class-specific and global image-level generative adversarial networks for semantic-guided scene generation. In *CVPR*, 2020.

[29] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

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

[31] Jiashun Wang, Chao Wen, Yanwei Fu, Haitao Lin, Tianyun Zou, Xiangyang Xue, and Yinda Zhang. Neural pose transfer by spatially adaptive instance normalization. In *CVPR*, 2020.

[32] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *NeurIPS*, 2018.

[33] Guanglei Yang, Hao Tang, Mingli Ding, Nicu Sebe, and Elisa Ricci. Transformer-based attention networks for continuous pixel-wise prediction. In *ICCV*, 2021.

[34] Zitong Yu, Xiaobai Li, Pichao Wang, and Guoying Zhao. Transrppg: Remote photoplethysmography transformer for 3d mask face presentation attack detection. *IEEE Signal Processing Letters*, 2021.

[35] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3d human pose estimation with spatial and temporal transformers. *ICCV*, 2021.

[36] Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Pons-Moll. Unsupervised shape and pose disentanglement for 3d meshes. In *ECCV*, 2020.

[37] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017.