Learning Compositional Representation for 4D Captures with Neural ODE Supplementary Material

Boyan Jiang[1*] Yinda Zhang[2*] Xingkui Wei[1] Xiangyang Xue[1] Yanwei Fu [1]
[1] Fudan University [2] Google

In this supplementary material, we provide implementation and data processing details, additional ablation study, results on various tasks for Warping Cars, results of motion transfer with different initial poses, and more qualitative comparisons to OFlow.

1. Implementation Details

In this section, we provide network architectures used for the compositional encoder, latent pose transformer, and implicit occupancy decoder in our framework. Additionally, we discuss more details about training and 4D completion experiment.

1.1. Network Architecture

Compositional Encoder The input of our encoder network is a point cloud sequence of size $(B, L, N, 3)$, where B, L, N denote batch size, length of input sequence and the number of points in each point cloud, respectively. The first frame of the point clouds is consumed by the identity encoder and pose encoder. For motion encoder, we concatenate all the input point clouds along the last dimension and set the input dimension of our encoder network to $3L$. The encoder network is a variation of PointNet [7] which has five residual blocks as shown in Fig. 1a. Each of the first four blocks has an additional max-pooling operation to obtain aggregated feature of size $(B, 1, C)$ where C denotes the dimension of hidden layers, and an expansion operation (expand the pooled feature to the size (B, N, C)) to make it suitable for concatenation. The output of the fifth block is passed through a max-pooling layer and a fully connected layer to get the final latent vector of dimension 128.

Latent Pose Transformer Our latent pose transformer (LPT) is built as a latent ODE conditioned by the motion code, which contains a vector field network and the architecture is shown in Fig. 1b. The vector field network is fed with the motion code and the concatenation of a time value τ and its corresponding pose code $c_p^{(\tau)}$ as inputs, then outputs the differential of pose code at time τ. There are five residual blocks in the vector field network, and the input of each block is summed up with the feature encoded from the motion code.

Given the motion code c_m, initial pose code c_p and a queried time value t, our LPT evaluates the vector field network multiple times to obtain transformed pose code at time t, which has the same dimension as the initial pose code.

Implicit Occupancy Decoder We utilize Occupancy Network (ONet) [5] as our decoder (Fig. 1c), which has the similar architecture with the vector field network. The decoder gets a 3D query point from a set of sample points $\mathcal{S}$ and a conditioning code as input, and outputs a scalar value which indicates the probability that the queried point is inside object surface. In our framework, the conditioning code is the concatenation of the identity code $\mathbf{c}_i$ and the pose code $\mathbf{c}_p^{(\tau)}$ at time τ. Following ONet, we use the conditional batch normalization (CBN) scheme to insert guidance encoded from the concatenated conditioning code.

1.2. More Details and Hyper-parameters

Training Our framework is implemented in PyTorch. For training, we use the Adam optimizer[3] with the learning rate 10^{-4}. The threshold for the output occupancy probability is set to 0.4. We use the adaptive-step solver *dopri5* [2] with relative tolerance of 10^{-3} and absolute tolerance of 10^{-5}. During training, 2048 points are sampled both at the initial time step $t = 0$ and a randomly selected time step $t > 0$ for every sequence to compute loss.

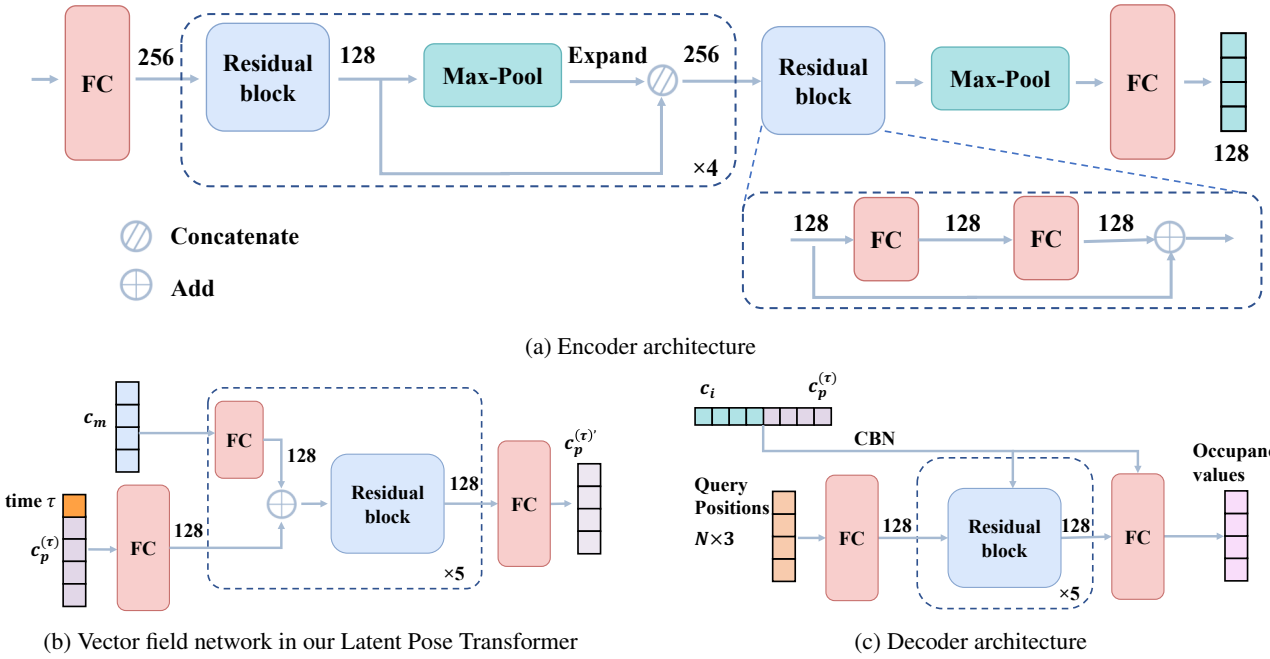
4D Completion For 4D completion, we use the same hyper-parameters for Occupancy Flow (OFlow) [6] and our method. We initialize the latent codes to be Gaussian noise with standard deviation 0.1 and use the Adam optimizer with learning rate 10^{-2} to perform back-propagation for 500 iterations. We reconstruct and compute BCE loss on all the observations in each iteration.

---

* indicates equal contributions.

Boyan Jiang Xingkui Wei and Xiangyang Xue are with the School of Computer Science, Fudan University.

Yanwei Fu is with the School of Data Science, MOE Frontiers Center for Brain Science, and Shanghai Key Lab of Intelligent Information Processing, Fudan University.



(a) Encoder architecture

(b) Vector field network in our Latent Pose Transformer

(c) Decoder architecture

Figure 1: **Detailed architectures of our framework.**

2. Data Processing

D-FAUST For our Identity Exchange Training (IET) strategy, all combinations of human identities and motions are required. Since all the mesh models in the original D-FAUST dataset have registered with the SMPL [4] model, we retrieve the SMPL identity and pose parameters for every mesh model by optimizing with back-propagation. The mean L2 distance between the predicted vertices and the ground truth vertices is used as the loss function.

We need point cloud sequences and query points for training purpose. When sampling the input point clouds, we do not perform a separate normalization for each model like OFlow. Instead, we keep the locations and scales of the original outputs of the SMPL model as they are already aligned. For sampling query points, we perform a global normalization for all the mesh models in our augmented dataset as described in Section 3.5 of the main paper.

Warping Cars We thank the authors of OFlow [6] for sharing the code, and follow their paper to generate the Warping car dataset. We choose 10 different car shape models in the watertight version of ShapeNet [1] "Car" category and generate 1000 warpings with the approach explained in Section 4.1 of the main paper. We adopt the same strategy as OFlow to obtain the input point clouds (normalized to a unit cube) and query points (sample uniformly in the bound volume), because the mesh models in the ShapeNet are consistently aligned and scaled.

3. Additional Ablation Study

Impact of the Identity Exchange Rate We train a set of models with the identity exchange rates set to $0\%, 25\%, 50\%, 75\%, 100\%$ respectively, and show the 4D reconstruction and motion transfer performance on D-FAUST dataset in Tab. 1. The overall performances for both tasks are in general stable w.r.t. the exchange rate. Though 4D reconstruction achieves the best accuracy at 0%, the model loses the shape/motion disentanglement and thus fails for motion transfer. In general, with 50%, the model achieves the best motion transfer performance and reasonably high reconstruction accuracy.

Exchange Rate	4D Reconstruction		Motion Transfer	
	IoU ↑	CD ↓	IoU ↑	CD ↓
0%	**83.3%**	**0.061**	65.3%	0.137
25%	81.8%	0.066	84.1%	0.057
50%*	81.8%	0.068	**85.0%**	**0.055**
75%	81.2%	0.068	83.7%	0.059
100%	81.0%	0.070	84.4%	0.058

Table 1: **Results about different choices of the identity exchange rate during training.** * denotes our choice in the main paper. CD is short for Chamfer Distance.

4. Various Tasks for Warping Cars

Pose and Motion Transfer We evaluate the motion transfer performance of our method and OFlow on the Warping

Methods	Motion Transfer		Temporal Completion		Spatial Completion		Future Prediction	
	IoU ↑	CD ↓	IoU ↑	CD ↓	IoU ↑	CD ↓	IoU ↑	CD ↓
OFlow	30.8%	0.596	78.8%	0.138	80.2%	0.130	57.6%	0.293
Ours	68.9%	0.181	81.6%	0.117	81.3%	0.121	63.6%	0.227

Table 2: **Comparisons to OFlow on various tasks for our generated Warping Cars dataset.**

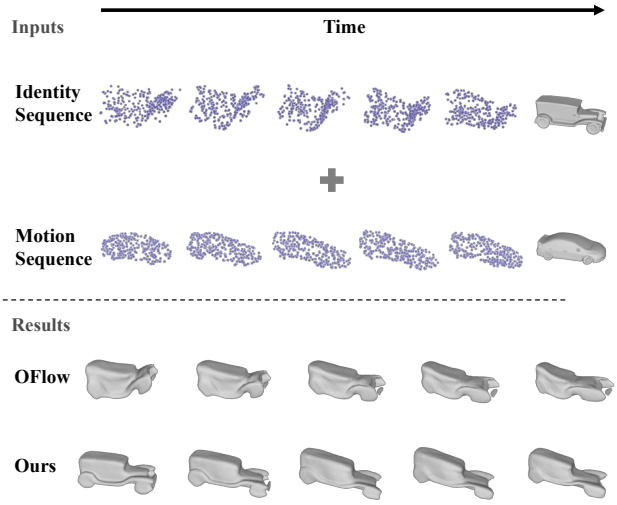


Figure 2: **Motion transfer (Warping Cars).**

Cars dataset. Similar to that on the D-FAUST dataset, we choose 20 car shape and warping pairs and generate mesh sequences of length $L = 17$ for evaluation. The quantitative results are shown in Tab. 2, and we shown a qualitative result in Fig. 2. Our method obtains significantly better performance. OFlow gets unsatisfactory transfer results due to the inconsistency between the initial pose of the identity sequence and the motion sequence. Thanks to the compositional representation which disentangles pose from shape properly, our model successfully transfers the motion to a new car shape. The results on non-human dataset also verify the potential of our model for motion transfer task on objects from various categories.

4D Completion We also conduct the 4D completion experiment for Warping Cars. Similar to the experiments on the D-FAUST dataset, we divide this task into two parts – temporal completion and spatial completion. We select 18 point cloud sequences in the testing set, each of length $L = 20$, and the strategies of removing frames and points are same as the previous experiments on the D-FAUST dataset, which are described in Section 4.4 of the main paper.

As the results shown in Tab. 2, Fig. 3 and 4, the proposed method achieves better results on both completion tasks than OFlow. We found that our method is more stable than OFlow during the completion experiments. The performance of OFlow heavily relies on the result of the first frame, because it only reconstruct mesh at $t = 0$, and then use a Neural ODE to transform the positions of the points on the reconstructed mesh. When the result of the first frame is unsatisfactory, it is difficult for OFlow to have good shapes for subsequent frames, as shown in Fig. 4. Our method applies the Neural ODE to update the latent pose code and reconstructs 3D model at each time step, which makes our results more stable.

Future Prediction In this experiment, we investigate the ability of our framework to predict future motion on our generated Warping Cars dataset. Same data for 4D completion task are taken, but we always remove the last 10 frames instead of randomly selected ones. We use the same hyper-parameters and optimization method based on back-propagation as the completion experiment. The quantitative results are shown in Tab. 2 and the qualitative results can be found in Fig. 5. As shown, our method is capable of tracking existing observations and predicting more accurate future motion than OFlow.

5. Motion Transfer with Different Initial Poses

In the previous motion transfer experiment, we transfer the motion code together with the initial pose code. To investigate if the motion code can be transferred without the initial pose code, we conduct an experiment that transfers a motion to different initial poses. The results are shown in Fig. 7. Specifically, first, we use our motion encoder to obtain the source motion code from the motion sequence shown in the first line of Fig. 7. Then five mesh models with different poses are selected, and we use our identity encoder and pose encoder to get the identity code and initial pose code for each model, which then are fed into our decoder together with the source motion code.

The sequences shown in the second to sixth rows are results after transferring. Applying a motion to a new pose is challenging and sometimes ill-defined (e.g. forcing a stand-up motion to start with a standing pose). Surprisingly, our model still produces reasonable results if the new pose is not too different from the original one, which shows some robustness of motion transfer against the initial pose.

6. More Qualitative Comparisons to OFlow

We show more qualitative results of 4D spatial completion on the D-FAUST dataset in Fig. 6, and 4D reconstruction on both the D-FAUST and Warping Cars dataset in Fig. 8, 9, 10 and 11.

References

[1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 2

[2] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980. 1

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[4] 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. 2

[5] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[6] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5379–5389, 2019. 1, 2

[7] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1

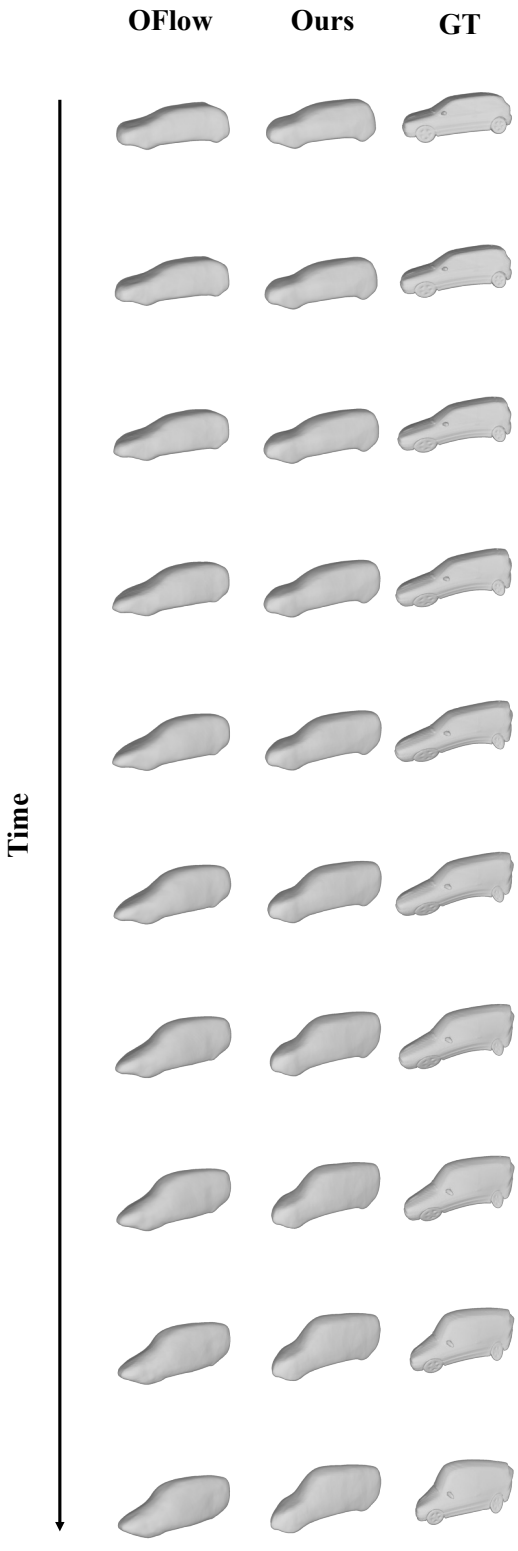


Figure 3: **4D temporal completion (Warping Cars).**

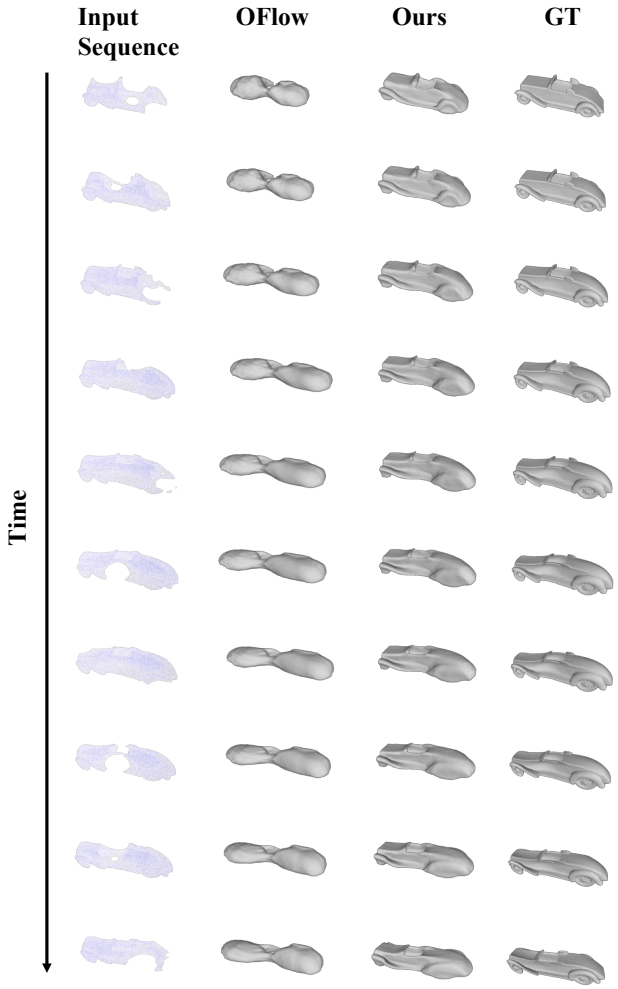


Figure 4: **4D spatial completion (Warping Cars).** Note that we randomly remove points in the occupancy grid for optimization and we show the corresponding partial point clouds here for the convenience of visualization.

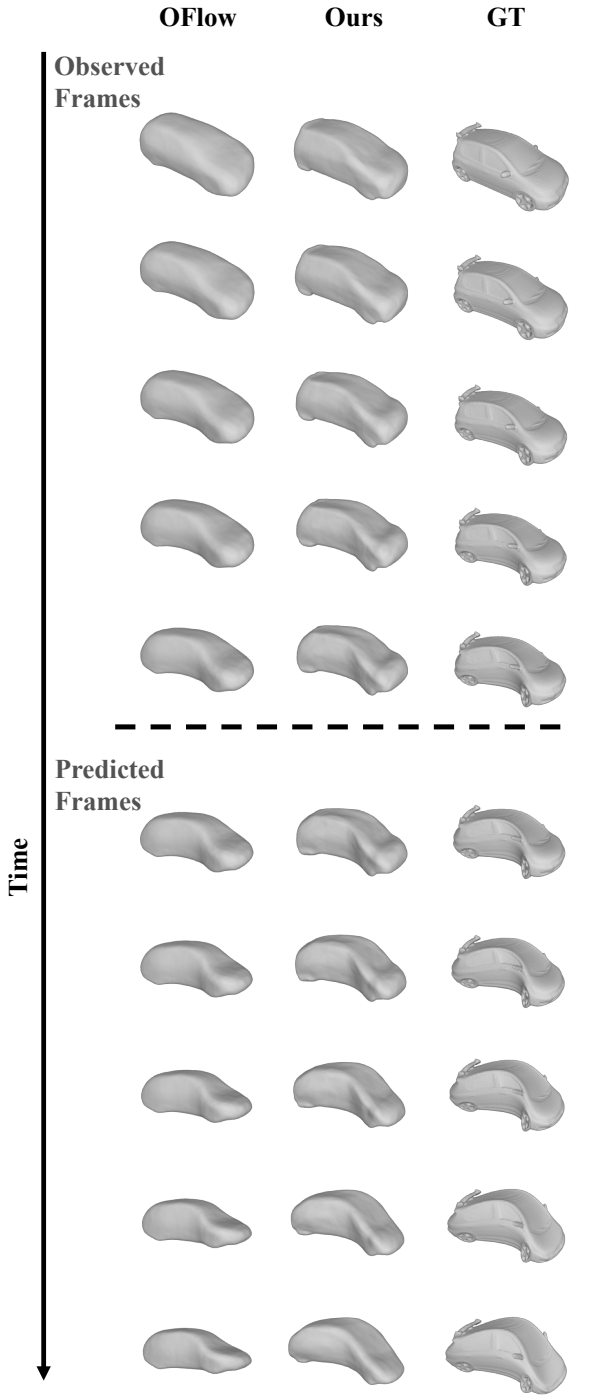


Figure 5: **Future prediction (Warping Cars).** We remove the last 10 frames of the test sequence to investigate the extrapolation ability of our method. The results above the dotted line are reconstructions for partial observation, and the results below are future predictions.

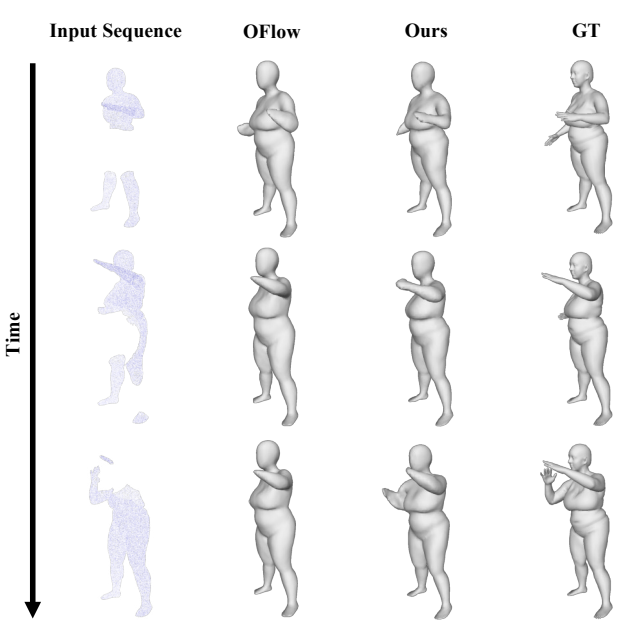


Figure 6: **4D spatial completion (D-FAUST).** Note that we randomly remove points in the occupancy grid for optimization and we show the corresponding partial point clouds here for the convenience of visualization.

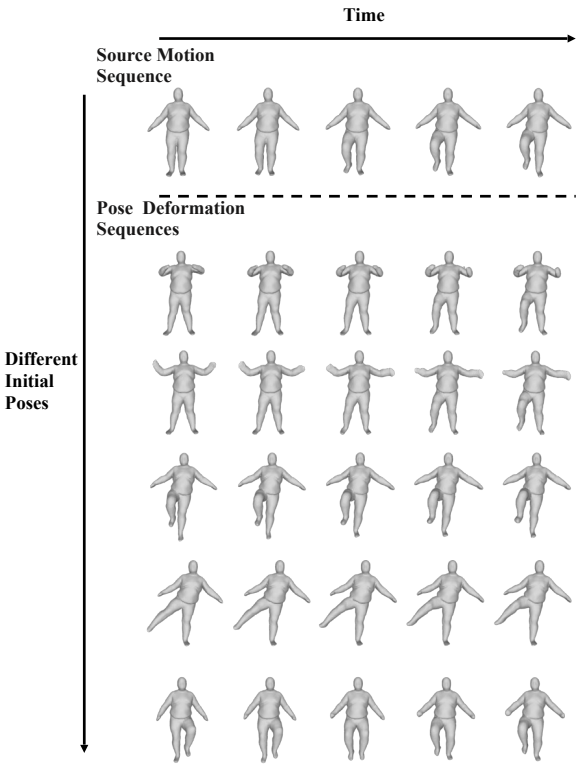


Figure 7: **Motion transfer with different initial poses.**

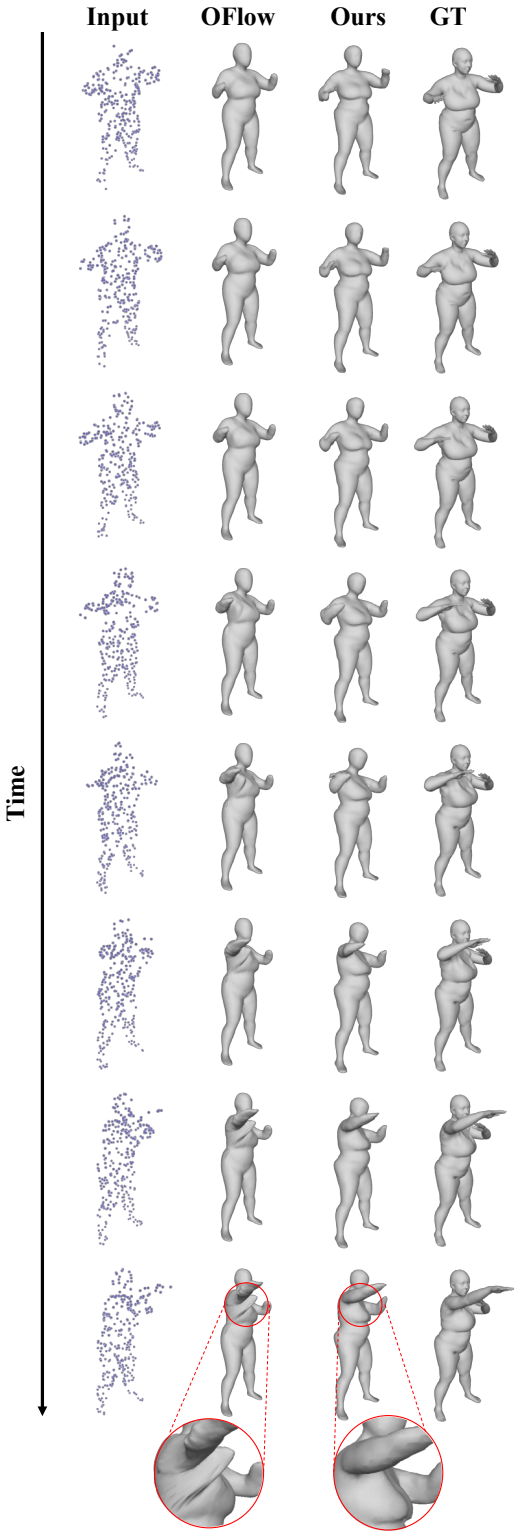


Figure 8: **4D reconstruction from point cloud sequence (D-FAUST).** We show the input, ground truth and outputs of OFlow and our method for 8 equally spaced time steps between 0 and 1.

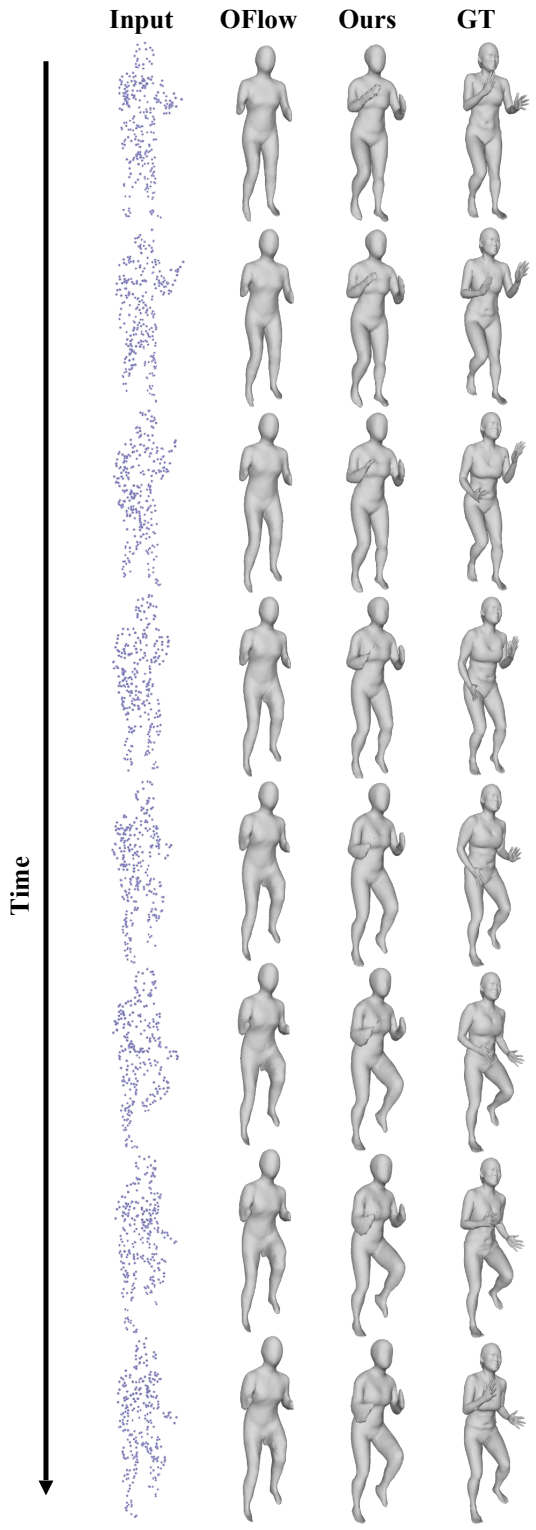


Figure 9: **4D reconstruction from point cloud sequence (D-FAUST).** We show the input, ground truth and outputs of OFlow and our method for 8 equally spaced time steps between 0 and 1.

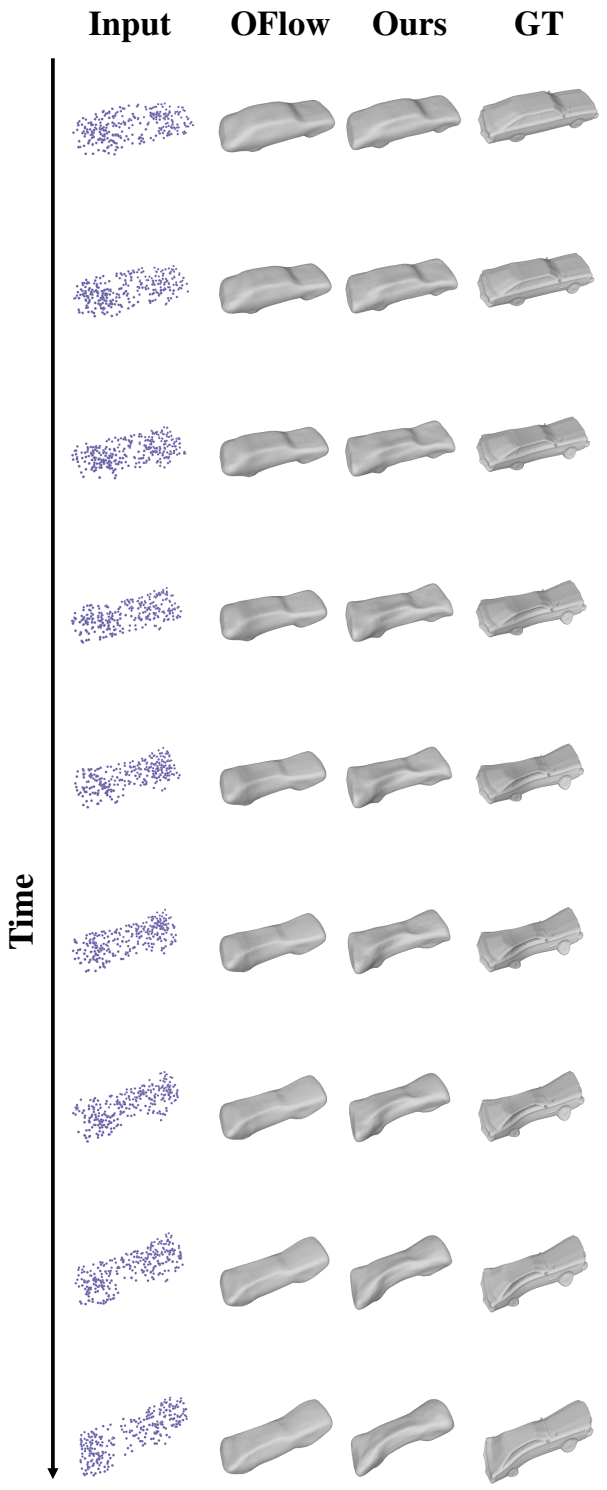


Figure 10: **4D reconstruction from point cloud sequence (Warping Cars).**

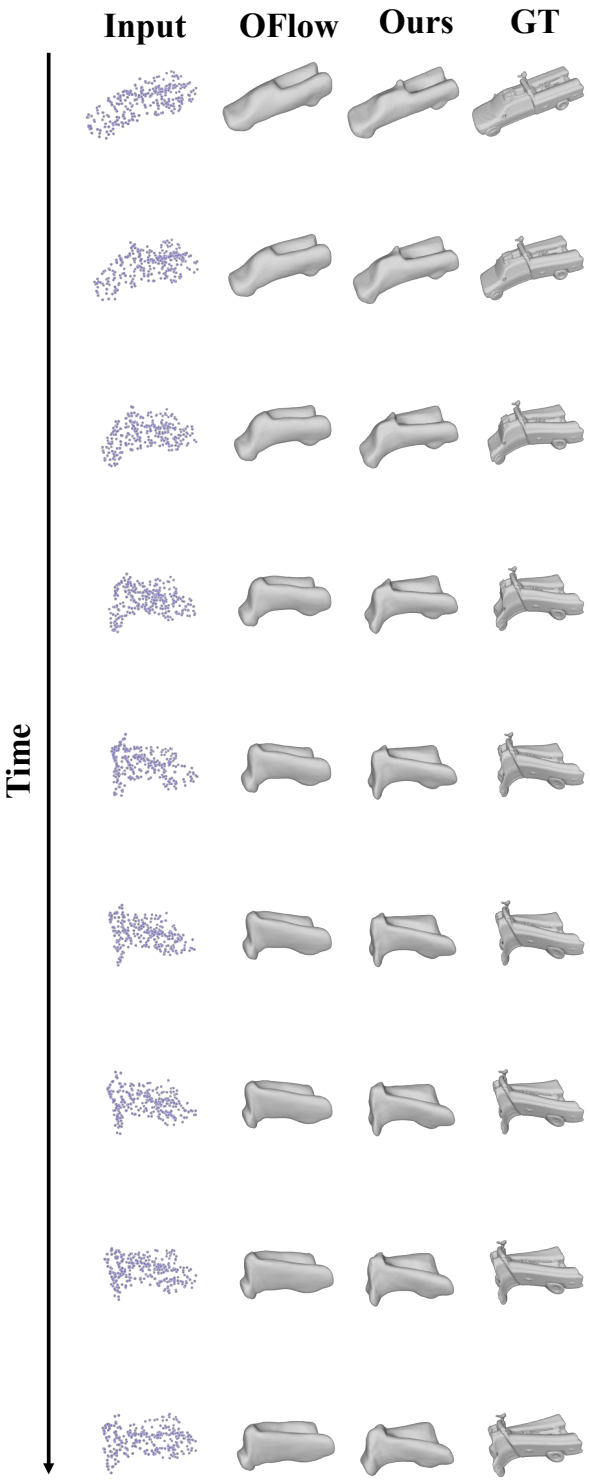


Figure 11: **4D reconstruction from point cloud sequence (Warping Cars).**