The Supplemental Material for "Decoupling Contact for Fine-grained Motion Style Transfer"

ACM Reference Format:
. 2024. The Supplemental Material for "Decoupling Contact for Fine-grained Motion Style Transfer". In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 3–6, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3680528.3687609

1 IMPLEMENTATION

1.1 Data Formatting and Dataset

We employ the approach suggested in [Holden et al. 2016] to represent poses with respect to the character's forward-facing direction. Specifically, the joint representation includes the position $p \in \mathcal{R}^3$, velocity $v \in \mathcal{R}^3$, and rotation $r \in \mathcal{R}^6$ for each frame. The joint rotation r is represented by the 2-axis matrix as proposed in [Zhang et al. 2018]. Consequently, a single frame in the CVAE has $12 \times J$ dimensions, where J represents the number of joints. The hip $h \in \mathcal{R}^3$ contains the translational velocities in the XZ plane and a root angular velocity around the Y axis.

All evaluations are based on human animations represented by joint rotations, rather than network output. Human animation reconstruction can be divided into three categories based on the output data used: rotation-based, position-based, and velocity-based, which primarily use rotations, positions, and velocities.

The rotation-based method solely uses rotation to recover the global position and rotation of joints via Forward Kinematics. The position-based method integrates both position and rotation data. We first recursively adjust the positions from the root to the end joints to ensure that bone lengths remain unaltered as follows:

$$p_i \leftarrow b_i \frac{p_i - p_j}{||p_i - p_j||_2} + p_j, \tag{1}$$

where b_i is the length of the i-th bone, and p_j represents the parent of p_i. Once the joints' positions are updated, we calculate the joints' rotation from the updated positions and leverage Forward Kinematics to attain the final animation similar to the rotation-based method. The velocity-based method firstly recovers the joints' position using the joints' velocity and the position at the initial frame, then processes similarly to the position-based method.

We utilize the CMU dataset and STYLE100 [Mason et al. 2022] for our experiments. STYLE100 is a labeled dataset suitable for evaluating the Fréchet Motion Distance (FMD) [Tang et al. 2023; Yin et al. 2023]. For all datasets, we divide the training set and testing set in a 9 : 1 ratio.

SA Conference Papers '24, December 3–6, 2024, Tokyo, Japan

ACM ISBN 979-8-4007-1131-2/24/12…$15.00
https://doi.org/10.1145/3680528.3687609

1.2 Training Details

1.2.1 $f_\delta(\cdot)$. The $f_\delta(\cdot)$ function takes the hip $\mathbf{h} \in \mathcal{R}^{T\times 3}$ as input and then generates contact states for two legs ($c_t \in \mathcal{R}^{T\times 2}$) for T frames. We acquire the ground-truth contact states using the following equation:

$$||p_j^{i+1} - p_j^i||_2^2 < \epsilon_v \quad \text{and} \quad he_j < \epsilon_H, \tag{2}$$

where p_j^i is the position of foot joint j at frame i, he_j is the joint height, $\epsilon_v = 0.02$ and $\epsilon_H = 10$ cm are thresholds.

We learned two $f_\delta(\cdot)$ on the CMU dataset and STYLE100 Dataset [Mason et al. 2022], respectively, using residual convolution networks. Each network comprises five convolution layers with 32 hidden units each. The first and the last layer are common layers while the three middle layers are residual layers. The temporal kernel sizes for these layers are 11, 5, 5, 5 and 1, respectively. ReLU serves as the activation function in the hidden layers, while the HardSigmoid function is employed in the output layer. We employ AdamW as our optimizer, setting the learning rate to 1e-4 and employing a weight decay of 1e-2.

1.2.2 Style transfer architecture. We utilized similar structures as defined in [Jang et al. 2022] to configure our style GCNs, contact GCNs, and AdaIN modules. Besides, we exclude the hip from M_s and M_c by setting the corresponding dimension to zero when input to the style GCNs and contact GCNs. The trajectory CNN is a residual convolution network comprising three convolution blocks, each followed by a pooling layer. Each convolution block includes a linear layer and three additional residual convolution layers with kernel sizes of 3, with hidden units set to 16, 32, and 32, respectively. Rather than considering all joints, the trajectory CNN only considers hip velocity as input.

The transformer module of the generator first leverages a linear layer to transform the output of the AdaINs into a 256-dimensional variable, which is then embedded with position encoding, and fed into three transformer layers. Following that, it is fed into a linear layer to produce the hip velocity and latent variable z. We employ the AdamW optimizer and set both the learning rate and weight decay to 1e-4.

We use the encoder of Transformer-based Variational AutoEncoder (VAE) from [Chen et al. 2023] as our CVAE encoder. We begin by embedding the sequence with global position tokens. Subsequently, the embedded sequence is fed into multiple transformer layers, producing Gaussian distribution parameters μ and Σ. Finally, we sample the latent variable $\mathbf{z}$ using the reparameterization trick in [Kingma and Welling 2013].

The CVAE decoder employs a transformer for the decoding process, in which the attention blocks use the hip as the query vector and derive the key and value from the latent variable z. Before feeding the hip into the transformer, we perform a linear transformation on it to ensure that its dimension corresponds to that of z and then apply the global positional embedding. To generate the final motion, the transformer's output is fed into a linear layer.

The CVAE architecture consists of 9 transformer layers for both the encoder and decoder, with each layer having hidden units set to a size of 256. We use the GELU activation function, and the dimension of the latent variable z is set to 8×256, consistent with the configuration reported in [Chen et al. 2023]. We utilize the AdamW optimizer with a learning rate of 1e-4 and a weight decay of 1e-2.

2 MANIFOLD VARIANT

2.1 Frame-Level Methods

We present a variant of our method that encodes frame-level randomness rather than sequence-level randomness. Specifically, we model $\mathbf{z}$ as a vector of temporal latent variables: $\mathbf{z} = z^{1\sim T}$. Each $z^i \in \mathcal{R}^8$ is represented by eight coefficients that blend different phases, inspired by phase-based synthesis methods [Holden et al. 2017; Starke et al. 2022]. Our experiments demonstrate that this representation is sufficient for reconstruction, as shown in Tab. 3, where the reconstruction metric measures the L2 global position distance between the reconstructed motion and the ground truth. We apply a similar architecture to the frame-level method as in our manifold method, with some modifications. During encoding, we include a temporal dimension in the output. During decoding, we create the query vector $\mathbf{q}^{1\sim T}$ by concatenating the latent $\mathbf{z}^{1\sim T}$ and hip $\mathbf{h}^{1\sim T}$ sequences. The key $\mathbf{k} \in \mathrm{R}^{k\times d}$ and value $\mathbf{v} \in \mathrm{R}^{k\times d}$ of the attention layer are learned parameters that model the motion phases in the latent space, where $k = 8$ denotes the number of motion phases and d is the phase dimensions.

3 EXPERIMENTS

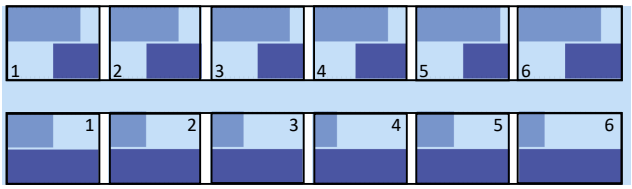

Fig. 1. The diagram illustrates the contact patterns of some short clips from the STYLE100 Dataset. Each row shows 6 contact timing patterns of 6 short clips with similar normalized hip velocities, with the top rectangle of each pattern representing the left foot and the bottom rectangle representing the right. The clips in different rows have different normalized hip velocities.

3.1 Hip velocity-contact timing relationship

To demonstrate that hip velocity patterns and contact timing are related, we conduct an experiment. First, we divide a walking sequence into multiple 20-frame short clips. We then apply min-max normalization to the hip velocities of each clip to exclude the influence of the speed magnitude since contact timing is more closely associated with the speed change than the absolute magnitude. Following that, we cluster these normalized clips based on their normalized hip velocities' L2 distance using the K-means method into 6 clusters. During the clustering process, similar contact patterns are assembled into the same cluster, demonstrating that one can deduce a sequence's contact timing pattern from its hip velocity change without knowing the hip's average speed magnitude.

3.2 Performance of the learned $f_\delta(\cdot)$

We evaluate the performance of the learned $f_\delta(\cdot)$ by examining precision and recall rate, which we refer to as *Contact Precision-Recall*. We first identify the frames in which any foot touches the ground and the frames in which it lifts off as *state change* frames. Precision is the percentage of the intersection of predicted and ground-truth state change frames in the predicted step change frames. The recall rate is the percentage of the intersection in ground-truth state change frames. Two state change frames in the intersection have a temporal difference of less than 2 frames (30 fps). It is noteworthy that the contact precision-recall is stringent since the 2 frames difference is short. Results are displayed in Tab. 1.

To examine whether the learned pattern is irrelevant to the absolute magnitude of the hip velocity, we use a factor in the range $[0.5, 1.5]$ to randomly scale the magnitude before feeding the hip velocity into the model. The experiment's results are presented in Tab. 1, which indicates that the absolute magnitude does not affect the contact timing significantly.

The metrics derived from evaluating the STYLE100 are higher than those obtained from the CMU. This difference may be due to the higher complexity of the motions in the CMU dataset. Specifically, the characters in the CMU dataset occasionally stand on stairs of different heights or swing from a trapeze, making it challenging to extract contact accurately or difficult for the model to learn the relationship.

In conclusion, our results reveal the presence of the relationship in both datasets. Notably, we do not focus on developing a superior architecture to enhance the precision further since the learned $f_\delta(\cdot)$ function performs sufficiently well for purposes of decoupling contact.

Table 1. The precision and recall rate of the learned $f_\delta(\cdot)$ on STYLE100 dataset and CMU dataset.

	Precision	Recall	Scale Precision	Scale Recall
STYLE100	0.8229	0.8754	0.7676	0.8455
CMU	0.7182	0.7083	0.6188	0.7005

3.3 Ablation study

3.3.1 Constraining foot velocity for controlling the contact. Our method is the first to constrain contact timing by constraining hip velocity. Constraining foot velocity is a possible alternative. However, foot velocity is relevant to both style and leg movements, thus constraining velocity causes loss conflict and undermines the quality and style effects. To validate this, we experimented to measure the performance of the method that constrains foot velocity. Because the pretrained manifold is designed to relate the hip to the contact timing, it is unnecessary when constraining foot velocity directly. Therefore, we remove it and directly output the motion by the transformer. Additionally, we replace $f_\delta(\mathbf{H})$ in Eq. 3 with a function that extracts the contact states based on foot velocity.

After training, the resulting character exhibited jitter around the contact frame, indicating unnaturalness. The quantitative results, as shown in Tab. 2, validate that constraining foot velocity significantly hindered the style.

$$L_{\text{ctt}} = ||f_\delta(\mathbf{H}_{scc}) - f_\delta(\mathbf{h}_c)||_2^2 + ||f_\delta(\mathbf{H}_{ssc}) - f_\delta(\mathbf{h}_s)||_2^2, \quad (3)$$

where $f_\delta(\cdot)$ is the learned function that captures the relationship between the hip velocity and the contact timing.

Table 2. Comparisons between our method with the method constraining velocity.

Methods	(FMD ↓	SA ↑)
Constrain velocity	1222	0.185
Ours	**72**	**0.943**

Table 3. Comparisons between our manifold with three previous manifolds and a variation of our manifold. The abbreviation "Rec" represents the reconstruction metric.

Methods	Ct P.	Ct R.	FMD_{vel}	Fv	Rec.
CMU					
MLD (pos)	0.4232*	0.5717*	0.0572	0.50	0.88
MLD (vel)	0.4321*	0.6893*	0.0596	0.29	1.03
MVAE (vel)	0.6355	0.5464	0.0189	0.35	0.68
Frame-level (pos)	0.2740	0.7410	0.0175	0.80	**0.50**
Frame-level (vel)	0.3948	0.7138	0.0182	0.53	0.67
VQVAE (pos)	0.4989	0.7335	**0.0106**	0.45	1.69
VQVAE (vel)	0.7319	0.7678	0.0120	0.23	1.80
Ours (pos)	0.5981	0.7787	0.0160	0.33	0.64
Ours (vel)	**0.7403**	**0.7930**	0.0171	**0.17**	0.83
100STYLE					
MLD (rot)	0.3899*	0.4760*	0.0269	1.47	0.39
MLD (pos)	0.3889*	0.4757*	0.0261	1.42	0.38
MLD (vel)	0.4618*	0.4311*	0.0265	0.92	0.36
MVAE (rot)	0.6145	0.5221	0.9100	1.13	**0.26**
MVAE (vel)	0.6129	0.4843	0.0265	0.84	0.49
Frame-level (rot)	0.4985	0.6953	0.0191	1.42	0.62
Frame-level (pos)	0.4999	0.6953	0.0190	1.26	0.61
Frame-level (vel)	0.5613	0.6480	0.0190	0.95	0.79
VQVAE (rot)	0.8269	0.8579	0.0447	0.76	1.44
VQVAE (pos)	0.8162	0.8688	0.0440	0.79	1.45
VQVAE (vel)	0.8633	0.8554	0.0475	**0.67**	1.53
Ours (rot)	0.8117	**0.8917**	0.0157	0.99	0.76
Ours (pos)	0.8156	0.8875	**0.0156**	0.93	0.76
Ours (vel)	**0.8782**	0.8712	0.0157	0.79	0.91

3.3.2 Data representations. Table 3 presents the metrics of different manifolds. The comparison is conducted using position-based, velocity-based, and rotation-based representations. Our results demonstrate that the velocity-based representation exhibits fewer foot skating artifacts, and higher contact precision compared to the other two representations.

3.3.3 Style Loss. If the latent variable z fails to capture the style variation correctly, adding style loss can considerably impair performance in the style transfer task. This is demonstrated in Table 4, which shows that including style loss during training results in worse FMD and SA metrics for the MLD model.

Table 4. Comparisons between different manifolds for our framework.

Methods	(XZ ↓	Angle ↓)	(FMD ↓	SA ↑)	(Ct. P ↑	Ct. R ↑)	Fv ↓
- Style loss							
MotionPuzzle	5.5	0.046	87	**0.920**	0.467	0.476	1.68
MLD	3.3	0.027	135	0.751	0.862	0.872	**0.60**
Frame-Level	2.5	0.027	194	0.571	**0.919**	**0.926**	0.61
Ours	**2.4**	**0.013**	**85**	0.879	0.849	0.849	0.61
+ Style loss							
MLD	3.2	0.029	282	0.574	**0.804**	**0.813**	0.83
Frame-Level	2.9	0.036	134	0.729	0.798	0.841	0.61
Ours	**2.7**	**0.023**	**69**	**0.931**	0.785	0.802	**0.60**

3.4 Comparisons

The complete quantitative results of the comparison are shown in Tab. 5. The data in brackets represents the performance of previous methods that use Ik-based post-processing. Due to the lack of ground-truth contact timing in the interpolation with a factor of 0.5, we estimate the contact timing using the learned function $f_\delta(\cdot)$, with the input as the predicted hip velocity. This metric is used to assist in measuring the motion quality.

Notably, although Motion Puzzle's interpolation with a factor of 1.0 achieves superior contact precision-recall as this configuration is equivalent to reproducing the style sequence, it presents poor results on contact precision-recall for factor 0.5, evidencing its poor performance in interpolation.

REFERENCES

Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, and Gang Yu. 2023. Executing your commands via motion diffusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18000–18010.

Daniel Holden, Taku Komura, and Jun Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics* 36, 4 (2017), 1–13.

Daniel Holden, Jun Saito, and Taku Komura. 2016. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics* 35, 4 (2016), 1–11.

Deok-Kyeong Jang, Soomin Park, and Sung-Hee Lee. 2022. Motion puzzle: Arbitrary motion style transfer by body part. *ACM Transactions on Graphics* 41, 3 (2022), 1–16.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Ian Mason, Sebastian Starke, and Taku Komura. 2022. Real-time style modelling of human locomotion via feature-wise transformations and local motion phases. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 5, 1, 1–18.

Sebastian Starke, Ian Mason, and Taku Komura. 2022. DeepPhase: periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics* 41, 4 (2022), 1–13.

Xiangjun Tang, Linjun Wu, He Wang, Bo Hu, Xu Gong, Yuchen Liao, Songnan Li, Qilong Kou, and Xiaogang Jin. 2023. RSMT: Real-time stylized motion transition for characters. In *SIGGRAPH '23 Conference Proceedings* (August 6-10).

Wenjie Yin, Hang Yin, Kim Baraka, Danica Kragic, and Mårten Björkman. 2023. Dance style transfer with cross-modal transformer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 5058–5067.

He Zhang, Sebastian Starke, Taku Komura, and Jun Saito. 2018. Mode-adaptive neural networks for quadruped motion control. *ACM Transactions on Graphics* 37, 4 (2018), 1–11.

Table 5. Comparisons between our method with previous motion style transfer methods. (XZ, Angle) are trajectory metrics, (FMD, SA) are style metrics and (Ct P., Ct R.) are contact timing metrics. Fv represents the foot skating metric. Fv of the dataset is 0.64.

Methods	(XZ ↓	Angle ↓)	(FMD ↓	SA ↑)	(Ct P. ↑	Ct R. ↑)	Fv ↓
Style interpolation							
Aberman et al. (0.5)	N/A	N/A	N/A	N/A	0.862	0.864	0.72
Motion Puzzle (0.5)	5.6	0.049	N/A	N/A	0.558	0.655	1.24
Motion Puzzle (+ decouple, 0.5)	**1.1**	**0.007**	N/A	N/A	0.433	0.472	1.46
Ours (0.5)	2.2	0.017	N/A	N/A	**0.889**	**0.889**	**0.53**
Aberman et al. (1.0)	N/A	N/A	191 (247)	0.732 (0.574)	0.791 (**0.839**)	0.773 (0.798)	0.83 (0.67)
Motion Puzzle (1.0)	5.5	0.046	87 (167)	0.920 (0.760)	0.467 (0.800)	0.476 (0.866)	1.68 (0.78)
Motion Puzzle (+ decouple, 1.0)	1.7	**0.010**	**69** (145)	0.940 (0.812)	0.363 (0.733)	0.294 (**0.867**)	1.91 (0.87)
Ours (1.0)	**1.6**	0.014	72	**0.943**	0.782	0.799	**0.63**
Contact interpolation							
Aberman et al. (0.5)	N/A	N/A	180	0.769	0.341*	0.371*	1.60
Motion Puzzle (0.5)	45	0.465	**53**	**0.979**	0.343*	0.493*	1.39
Motion Puzzle (+ decouple, 0.5)	**5.7**	0.050	61	0.955	0.274*	0.262*	2.08
Ours (0.5)	6.4	**0.034**	65	0.948	**0.677***	**0.660***	**0.60**
Aberman et al. (1.0)	N/A	N/A	180	0.773	0.232	0.498	2.05
Motion Puzzle (1.0)	79	0.862	**40**	**0.990**	**0.874**	**0.919**	0.72
Motion Puzzle (+ decouple, 1.0)	3.6	**0.020**	52	0.968	0.458	0.270	1.91
Ours (1.0)	**2.9**	0.031	70	0.931	0.741	0.784	**0.58**
Trajectory interpolation							
Motion Puzzle (+ decouple, 0.5)	N/A	N/A	**52**	**0.964**	0.397	0.440	1.48
Ours (0.5)	N/A	N/A	55	0.963	**0.682**	**0.702**	**0.46**
Motion Puzzle (+ decouple, 1.0)	**2.7**	**0.015**	**48**	**0.970**	0.400	0.510	1.46
Ours (1.0)	8.9	0.031	65	0.943	**0.642**	**0.678**	**0.60**