ReMoS: 3D Motion-Conditioned Reaction Synthesis for Two-Person Interactions –Appendix–

Anindita Ghosh[1,2,3], Rishabh Dabral[2,3], Vladislav Golyanik[2,3], Christian Theobalt[2,3], and Philipp Slusallek[1,3]

[1] German Research Center for Artificial Intelligence (DFKI)
[2] Max-Planck Institute for Informatics (MPII)
[3] Saarland Informatics Campus

We provide additional details on the loss functions used for training ReMoS, more statistics on the ReMoCap dataset, and describe how the datasets and baselines are prepared for evaluation. We also show some additional results.

1 Additional Details of Loss Functions

Kinematic Loss Terms. We describe the details of the velocity, acceleration, bone length and foot sliding losses loss terms from Eqn. 10 in the main paper. To improve the temporal consistency of the motion [12], we minimize the joint velocities and joint accelerations between two consecutive frames of the ground-truth reactive motions, X, and the synthesized reactive motions, $\hat{X}$, defined as

$$\mathcal{L}_{vel} = \frac{1}{N-1}\sum_{n=0}^{N-1}\left\|\left(X^{n+1}-X^{n}\right)-\left(\hat{X}^{n+1}-\hat{X}^{n}\right)\right\|_2^2, \tag{1.1}$$

$$\mathcal{L}_{acc} = \frac{1}{N-2}\sum_{n=0}^{N-2}\left\|\left(X^{n+2}-2X^{n+1}+X^{n}\right)-\left(\hat{X}^{n+2}-2\hat{X}^{n+1}+\hat{X}^{n}\right)\right\|_2^2, \tag{1.2}$$

where N is the total number of frames.

Additionally, we introduce a bone length consistency loss, $\mathcal{L}_{bone}$, to ensure that the synthesized reactor joint positions satisfy the skeleton consistency [5]. We define this loss as

$$\mathcal{L}_{bone} = \left\|\mathbf{B}\left(X\right)-\mathbf{B}\left(\hat{X}\right)\right\|_2^2, \tag{1.3}$$

where $\mathbf{B}$ represents the bone lengths in a pre-defined human body kinematic tree.

Further, foot sliding is a common artifact in motion synthesis [8,9]. We constrain this by ensuring that the toe joint in contact with the ground plane has zero velocity. We use a binary foot contact loss [11, 12] on the foot joints of the synthesized pose to ensure that the output motion does not slide across the ground plane, defined as

$$\mathcal{L}_{foot} = \frac{1}{N-1}\sum_{n=0}^{N-1}\left\|\left(\hat{X}^{n+1}-\hat{X}^{n}\right)\cdot\hat{\mathbb{1}}_{foot}^{n}\right\|_2^2, \tag{1.4}$$

where $\hat{\mathbb{1}}^n_{foot} \in \{0, 1\}$ is the foot-ground contact indicator for the synthesized reactive motion $\hat{X}^n$ at each frame n.

2 ReMoCap Dataset Analysis

Our proposed ReMoCap dataset covers two types of motion, namely the Lindy Hop dance and the martial art technique of Ninjutsu (see Sec. 4 in the main paper).

Lindy Hop motion capture. The Lindy Hop part of the dataset consists of 8 dance sequences captured at 50 fps, each around 7.5 minutes long, resulting in around 174.2K motion frames. We had 4 trained dancers, 2 males (denoted A and B) and 2 females (denoted C and D), participate in the Lindy Hop motion capture. We pair the dancers as (A, C), (B, D), (A, D), and (B, C). Of these pairings, (A, D) contains dance sequences not performed by the other three pairs (in terms of twists and maneuvers). We also capture multiview RGB videos at 50 fps from 116 camera views for each sequence, which can benefit two-person pose reconstruction work in the future. We show samples from these videos in Fig. 2.1. From these sequences, almost 145.2K frames have a hand-in-hand contact between the two dancers with a contact threshold of 50 mm between the finger joints of the two dancers. By increasing the contact threshold to 100 mm, the number of frames where the two dancers have contact increases to 147.5K.

Ninjutsu motion capture. The Ninjutsu part of the dataset consists of 79 sequences each captured at 25 fps. The sequences vary in length with a total number of around 99.8K motion frames resulting in around 66.5 minutes of motion. We had 5 trained, male Ninjutsu artists participate in the Ninjutsu motion capture. We pair them in all possible combinations and ask them to perform different variations of motion. Along with the 3D pose, we also capture multiview RGB videos at 25 fps using 116 cameras. We show samples from these videos in Fig. 2.2. From these sequences, almost 81K frames have contact-based interactions between the two performers, where the closest distance between any joints is 50 mm.

3 Dataset and Baseline Preparation

We discuss the preparation of the different datasets and the baseline methods for our evaluation purposes.

3.1 Dataset Preparation

Preparing the Lindy Hop data in ReMoCap. We split the dataset such that motions captured from dancer pairs (A, C), (B, D), and (B, C) are in our training set, and motions captured from pair (A, D) are in our test set. We

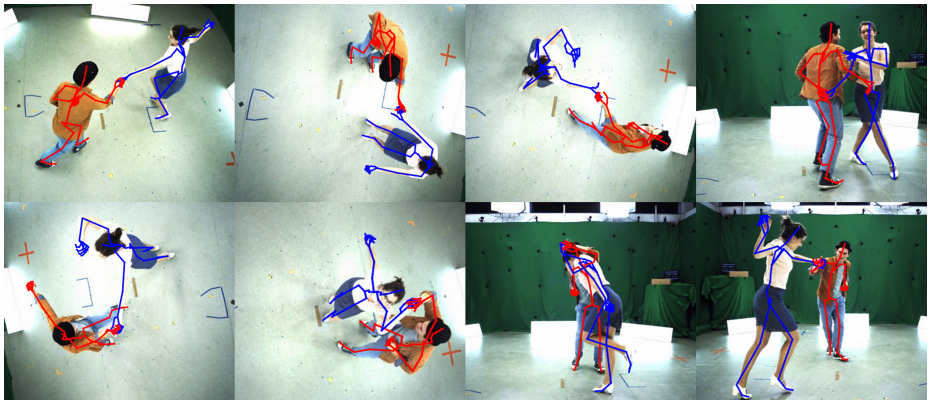

Fig. 2.1: Samples from the Lindy Hop motion capture for the ReMoCap dataset. We show multi-view RGB samples with corresponding 3D poses from our Lindy Hop motion capture performed by trained dancers. Lindy-hop requires coordination between the two dancers, while also allowing individual dancers the freedom to perform their own motions. This makes it suitable for testing our reactive motion synthesis approach.

downsample each motion sequence to 20 fps and filter the frames where the dancing partners have hand-to-hand contact between the actors' and reactors' finger joints. We represent each character using 27 body joints and 22 hand joints. We convert the 3D joint angle representations into joint positions using forward kinematics and then convert them to root local representations, as explained in Sec. 5.1 in the main paper. For training, we use a sequence length of 20 frames.

Preparing the Ninjutsu data in ReMoCap. We divide the whole dataset into roughly 3 : 1 train-test ratio and take 28 sequences of diverse attacking and maneuvering motions for testing, and the rest for training. We downsample each motion sequence to 10 fps, and filter out the frames where the pairs are more than 1 meter of each other. We represent each character using 27 body joints and 22 hand joints. We convert the 3D joint angle representations into joint positions using forward kinematics and then convert them to root local representations, as explained in Sec. 5.1 in the main paper. For training, we use a sequence length of 50 frames.

Preparing the Extreme Pose Interaction (ExPI) Dataset [3]. The ExPI dataset consists of 2 pairs of professionals performing acrobatics and Lindy Hop aerial sequences. It consists of 16 different acrobatic actions. Each couple consists of a leader and a follower. We aim to synthesize the motions of the followers as they react to the leaders' movements. We use the *common action split* proposed by the original authors [3], and split the dataset into train and test sets such that all the actions performed by (A, B) are in the train set and all the actions performed by (C, D) are in the test set. We represent each subject using 16 joints

Fig. 2.2: Samples from the Ninjutsu motion capture for the ReMoCap dataset. We show multi-view RGB samples with corresponding 3D poses from the Ninjutsu motion capture performed by trained artists. In contrast to existing martial arts datasets [7, 13], we include finger joint motion capture and moves of varying interaction complexity.

(omitting the *'lhead'* and *'rhead'* joints) and convert the global 3D joint positions given in the dataset to root relative joint representations, as explained in Sec. 5.1 in the main paper. Since the ExPI dataset does not have hand motions, we only train with body motions and forego the hand diffusion stage. We train ReMoS for about 20K iterations on the ExPI dataset using the Adam optimizer [4], with a base learning rate of 10^{-5} and a batch size of 32.

Preparing the Character-Character Dataset (2C) [7]. The 2C dataset consists of full-body motions of kickboxing actions performed by pairs of participants. The interactions include motions such as *kicking* and *punching*, with diverse reactions such as *avoiding* and *being hit*. We use the pose sequence of the leading character, who throws the punches and kicks, as the acting sequence for our model. We aim to synthesize the full body motion of the reacting character, who is blocking or avoiding the moves, as our output. Following the split of MixNMatch [2], we use a roughly 3 : 1 train-test ratio to train our method. Each character contains 25 joints and we convert the 3D joint angle representations into joint positions using forward kinematics and then convert them to root relative joint position representations, as explained in Sec. 5.1 in the main paper. Since the 2C dataset does not have hand motions, we only train with body motions and forego the hand diffusion stage. We train ReMoS for about 25K iterations on the 2C dataset using the Adam optimizer [4], with a base learning rate of 10^{-5} and a batch size of 16.

Preparing the InterHuman Dataset [5]. We report additional results on the InterHuman dataset in this appendix. It consists of human-human interactions for daily motions, such as passing objects, greeting, and communicating, and

professional activities, such as, Taekwondo, Latin dance, and boxing. It consists of a total of 7,779 motions with 22 joints per person. We randomly select the pose sequence of one of the characters as the acting sequence for each motion to train our model. We aim to synthesize the full body motion of the corresponding other character in each motion as our output. We follow the split of InterGen [5] for our experiments. Since the InterHuman dataset does not have hand motions, we only train with body motions and forego the hand diffusion stage. We train ReMoS for about 45K iterations on the InterHuman dataset using the Adam optimizer [4], with a base learning rate of 10^{-5} and a batch size of 64.

3.2 Baseline Preparation

As we mention in Sec. 5.2 in the main paper, we use InterFormer [1], MixNMatch [2], ComMDM [6], RAIG [10] and InterGen [5] as baselines. We describe how we use each of these methods in our setting.

InterFormer [1]. InterFormer consists of a transformer network with temporal and spatial attentions. It takes an input acting sequence Y and encodes it with spatial and temporal self-attention. It also needs the initial pose of the reactor X and predicts the subsequent frames of the reactor in an autoregressive manner. It uses information from skeletal adjacency matrices and an interaction distance module that provides information on the interactions. We use the normalization technique mentioned in Sec. 5.1 in the main paper to normalize the actor's and the reactor's body poses. We train InterFormer on an NVIDIA RTX A4000 GPU for about 20K iterations for both the LindyHop and the Ninjutsu sets of ReMoCap, using the Adam optimizer [4] with a base learning rate of 10^{-5} and a batch size of 128. We use 207 dimensional latent embedding and 6 layers in the transformer decoder with 3 heads to calculate the attention.

MixNMatch [2]. MixNMatch proposes an end-to-end framework to synthesize stylized reactive motion informed by multi-hot action labels. It operates in one of two settings, *interaction mixing* and *interaction matching*. In *interaction mixing*, it generates a reaction combining different classes of reactive styles according to the multi-label indicator. In *interaction matching*, it generates the reactive motion corresponding to the interaction type and the input motion. Our setting is similar to *interaction matching*, where we input the acting sequence into the model and synthesize the reactive sequence. We mask out the action label defining the interaction type from the input and train the reactor's motion X based on the actor's motion Y. We use the normalization technique mentioned in Sec. 5.1 in the main paper to normalize the actor's and the reactor's body poses. We train MixNMatch on an NVIDIA RTX A4000 GPU for about 3.6K iterations for both the LindyHop and the Ninjutsu sets of ReMoCap, using the Adam optimizer [4] with a base learning rate of 10^{-5} and a batch size of 16. We use 256 LSTM neurons for each spatial slice and 1,200 for the attention layer.

ComMDM [6]. ComMDM is proposed as a communication block between two MDMs [11] to coordinate interaction between two persons. It uses single-person motions from a pre-trained MDM as fixed priors, and a parallel composition with few-shot training that shows how two single-person motions coordinate for interactions. ComMDM is a single-layer transformer model that inputs the activations coming from the previous layer from the two MDM models, and learns to generate a correction term for the MDM models along with the initial pose of each person. ComMDM was originally trained for two motion tasks: *prefix completion* and *text-to-motion synthesis*. We follow the *prefix completion* setting of ComMDM which does not use textual annotations as a condition and was trained to complete 3 seconds of motion given a 1 second prefix. We train ComMDM on an NVIDIA RTX A4000 GPU for about 24K iterations for both the LindyHop and the Ninjutsu sets of ReMoCap, using the Adam optimizer [4] with a base learning rate of 10^{-5} and a batch size of 64. We use 256 dimensional latent embedding for the ComMDM block. During inference, we provide the full ground truth motion of actor Y into the first MDM module. Thus, the ComMDM block takes in the ground truth features from the first MDM module and the learned features from the second MDM module. In turn, the output of the second MDM module is the reactive motion X for our setting.

RAIG [10]. Role-Aware Interaction Generation (RAIG) is a diffusion-based model that learns two-person interactions by generating single-person motions for a designated role. The role is supplied in the form of textual annotations, which are translated into active and passive voices to ensure the text is consistent with each role. The model generates interactions with two transformers that share parameters, and a cross-attention module connecting them. The active and passive voice descriptions are proveded as inputs to the corresponding transformers responsible for generating the actor and the reactor. The transformers consist of cross-attention modules both for language and motion. To use RAIG as a baseline for our annotation-free setting, we mask out the cross-attention module for the language in both the transformers and train to generate two-person motions unconditionally. We normalize the interactions as described in the original paper [10]. We train RAIG on an NVIDIA RTX A4000 GPU for about 20K iterations for both the LindyHop and the Ninjutsu sets of ReMoCap, using the Adam optimizer [4] with a base learning rate of 2^{-4} and a batch size of 32. We use 512 dimensional latent embedding and 8 attention blocks. During inference, we freeze the transformer that learns the actor's motion Y. The other transformer generates the reactor's motion X, being influenced by the actor's ground truth motion.

InterGen [5]. InterGen is a diffusion-based approach that generates two-person motions from text prompts. It was originally trained by conditioning on rich textual annotations. It uses cooperative denoisers with novel weight-sharing and a mutual attention mechanism to improve interactions between two persons. To use it as a baseline in our annotation-free setting, we mask out the text embeddings from the model input, and train InterGen to generate two-person

Table 4.1: Quantitative evaluation on the InterHuman dataset [5]. We compare ReMoS with state-of-the-art motion synthesis methods on the InterHuman [5] dataset. ↓: lower is better, ↑: higher is better, →: values closer to GT are better. **Bold** indicates best.

Methods	MPJPE (mm) ↓	MPJVE (mm) ↓	FID (body) ↓	Div →	Multi-modality ↑
GT	–	–	–	7.74	–
ComMDM [6]	76.4	2.75	0.72	7.17	1.71 ± 0.5
RAIG [10]	83.2	2.76	0.67	7.26	2.01 ± 0.6
InterGen [5]	69.5	2.61	0.59	7.32	2.11 ± 0.6
ReMoS (ours)	**66.7**	**2.56**	**0.56**	**7.33**	$\mathbf{2.13 \pm 0.3}$

Table 4.2: Trainable parameter counts.

Method	Params (full model)
InterFormer	$8.2M$
MixNMatch	**6.5M**
ComMDM	$22.2M$
RAIG	$81.2M$
InterGen	$170M$
ReMoS (ours)	$17.4M$

motions (both actor and reactor) unconditionally. We use the non-canonical motion representation proposed in the original paper [5]. During inference, we use the customization used in InterGen for *person-to-person generation*. We take a single-person motion (the actor's motion Y) as input, and freeze it during the forward diffusion process. The frozen weights from the first person propagate into the model, which then uses the ground truth actor's motions to reconstruct the second person's motion (the reactor's motion X). We train InterGen on an NVIDIA RTX A4000 GPU for about 30K iterations for both the LindyHop and the Ninjutsu sets of ReMoCap, using the Adam optimizer [4] with a base learning rate of 10^{-4}, a cosine LR scheduler, and a batch size of 64.

4 Additional Results

We provide additional results and the trainable parameter counts of all models. We further show how ReMoS can be used as a motion editing tool for character control applications.

4.1 Quantitative Evaluation on the InterHuman Dataset [5]

We report additional evaluation of ReMoS compared to its diffusion-based baselines on the InterHuman [5] dataset in Table 4.1. We report performance on the standard evaluation metrics, including MPJPE, MPJVE, FID, Diversity and Multi-modality. For Multi-modality, we generate each sequence 5 times and report numbers with a 95% confidence interval. InterHuman dataset does not provide hand motions, so we only evaluate the reactors' body motions. ReMoS achieves state-of-the-art performance in the aforementioned metrics in the InterHuman dataset, highlighting the utility of our method for diverse forms of two-person interactions.

Table 4.3: Quantitative evaluation on body joints. We compare the body synthesis module of ReMoS with state-of-the-art motion synthesis methods on body joints only. **Bold** indicates the best.

Methods	Lindy Hop (body only)				Ninjutsu (body only)			
	MPJPE ↓	MPJVE ↓	FID ↓	Div →	MPJPE ↓	MPJVE ↓	FID ↓	Div →
GT	-	-	-	7.62	-	-	-	11.5
MixNMatch	69.8	10.5	0.74	2.52	260.1	5.14	0.72	4.94
InterFormer	63.2	8.91	0.52	4.64	262.5	3.53	0.51	6.27
ComMDM	50.2	4.42	0.23	7.51	192.4	3.45	0.25	9.83
RAIG	68.5	4.01	0.26	9.02	188.3	4.25	0.19	10.14
InterGen	55.1	2.87	0.22	7.49	165.5	3.82	0.23	9.87
ReMoS (ours)	**40.2**	**2.21**	**0.12**	**7.52**	**137.2**	**3.19**	**0.16**	**10.26**

4.2 Trainable Parameters

We report the total number of trainable parameters of ReMoS as compared to the baseline methods. Table 4.2 shows that ReMoS has lesser trainable parameters than the existing diffusion-based two-person synthesis models [5, 6, 10].

4.3 Comparison with baselines without hand motions.

We compare the body synthesis module of ReMoS with baselines trained only on the body joints (Table 4.3). We report state-of-the-art performance for ReMoS even when finger joints are not included.

4.4 Motion Editing Applications of ReMoS

We describe how to use ReMoS as an interactive motion editing tool, providing control to animators for tasks such as *pose completion* and *motion in-betweening*. These are crucial applications that are possible due to the strong generative abilities of DDPMs. We provide visual results of these applications in our supplementary video.

Pose Completion with Controlled Joints. When an animator manually customizes some of the reactor's body joints to align with specific animation tasks, ReMoS can automatically synthesize the reactor's remaining body joints to complete the reactor's motion. We achieve this by providing the forward-diffused values of the controlled joints as the network input at each diffusion step. For example, to synthesize the motions of the remaining joints of the reactor's body given customized motions for some joints J_i and J_k, we set

$$X_B^{(0)} = f_{\theta_B}\left(X_B^{(t)}, t, Y_B, \mathbb{1}_{\{J_i, J_k\}}\right), \tag{4.1}$$

where $\mathbb{1}_{\{J_i, J_k\}}$ is a mask we use at each denoising step on all frames to ensure that the joints J_i and J_k are not denoised. Instead, we populate J_i and J_k with

the identical noise vectors as the ones used during forward diffusion, while introducing random noise to the rest of the joints throughout the sequence. Thus, animators can incorporate flexible spatial control over chosen joints while ReMoS synthesizes the remaining joints of the reactor to faithfully capture the interaction. In Fig. 5b in the main paper, we show the results of a pose-completion application where we manually control the right-hand wrist joint of the reactor and let ReMoS synthesize the remaining body joints conditioned on the actor.

Motion In-Betweening. Likewise, we can use the existing framework to perform motion in-betweening for the reactive sequence. We achieve this by providing some keyframes of the reactive motion and letting ReMoS synthesize the intermediate frames using a motion in-betweening routine. To synthesize the reactor's motion between two given keyframes N_a and N_b through reverse diffusion, we set

$$X_B^{(0)} = f_{\theta_B}\left(X_B^{(t)}, t, Y_B, \mathbb{1}_{\{N_a, N_b\}}\right), \tag{4.2}$$

where $\mathbb{1}_{\{N_a, N_b\}}$ is a mask we use at each denoising step to ensure that all joints at frames N_a and N_b are not denoised. Thus, ReMoS can fill in the motions between the two seed frames as shown in Fig. 5c in the main paper.

References

1. Chopin, B., Tang, H., Otberdout, N., Daoudi, M., Sebe, N.: Interaction transformer for human reaction generation. IEEE Transactions on Multimedia (2023)
2. Goel, A., Men, Q., Ho, E.S.L.: Interaction Mix and Match: Synthesizing Close Interaction using Conditional Hierarchical GAN with Multi-Hot Class Embedding. Computer Graphics Forum (2022)
3. Guo, W., Bie, X., Alameda-Pineda, X., Moreno-Noguer, F.: Multi-person extreme motion prediction. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
4. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
5. Liang, H., Zhang, W., Li, W., Yu, J., Xu, L.: InterGen: Diffusion-based multi-human motion generation under complex interactions. International Journal for Computer Vision (IJCV) (2024)
6. Shafir, Y., Tevet, G., Kapon, R., Bermano, A.H.: Human motion diffusion as a generative prior. In: International Conference on Learning Representations (ICLR) (2024)
7. Shen, Y., Yang, L., Ho, E.S.L., Shum, H.P.H.: Interaction-based human activity comparison. IEEE Transactions on Visualization and Computer Graphics (2020)
8. Shimada, S., Golyanik, V., Xu, W., Pérez, P., Theobalt, C.: Neural monocular 3d human motion capture with physical awareness. ACM Transactions on Graphics (ToG) (2021)
9. Shimada, S., Golyanik, V., Xu, W., Theobalt, C.: Physcap: Physically plausible monocular 3d motion capture in real time. ACM Transactions on Graphics (ToG) (2020)
10. Tanaka, M., Fujiwara, K.: Role-aware interaction generation from textual description. In: International Conference on Computer Vision (ICCV) (2023)

11. Tevet, G., Raab, S., Gordon, B., Shafir, Y., Bermano, A.H., Cohen-Or, D.: Human motion diffusion model. arXiv preprint arXiv:2209.14916 (2022)
12. Tseng, J., Castellon, R., Liu, K.: Edge: Editable dance generation from music. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
13. Zhang, W., Liu, Z., Zhou, L., Leung, H., Chan, A.B.: Martial arts, dancing and sports dataset: A challenging stereo and multi-view dataset for 3d human pose estimation. Image and Vision Computing **61** (2017)