Controllable Human-Object Interaction Synthesis

Jiaman Li[1], Alexander Clegg[2], Roozbeh Mottaghi[2],
Jiajun Wu[1], Xavier Puig[2†], C. Karen Liu[1†]

[1]Stanford University, [2]FAIR, Meta

The supplementary material includes details on the implementation of the interaction synthesis model and the scene-aware interaction synthesis pipeline, details of the evaluation metrics and baselines. We also encourage readers to watch our supplementary video for more qualitative results.

1 Implementation Details

Data Representation. To ease the training, the motion representation is converted to a canonicalized coordinate frame, following prior work [3, 5, 8]. Specifically, given a sequence of object and human poses $\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_T$, we apply a rotation around the up axis ($+\boldsymbol{z}$) to the motion sequence such that the first frame's human body left axis (the direction of a vector from the root joint to the left hip joint) is aligned with $+\boldsymbol{x}$ axis. Also, we apply a translation to transform the root joint position of the human to be at $x = 0, y = 0$. In addition, to improve the robustness of the model for different object geometry inputs with various orientations, we employ a strategy that randomly samples a frame from a sequence and uses the object geometry of the selected frame as input geometry condition.

Model Architectures. Our denoising network employs a transformer-based model architecture with four self-attention blocks. Each self-attention block consists of a multi-head attention layer and a position-wise feed-forward layer. The number of attention heads in the self-attention layer is 4. The dimension of the key, query, and value in the self-attention block is 256. Our model is trained on motion data with a window size of 120 in 30 fps.

Training Details. We use PyTorch [6] to implement our interaction synthesis model. We use Adam optimizer [2] and start the training with a learning rate 0.0001 and a batch size 32. The training takes approximately 35 hours to converge using a single NVIDIA Titan RTX GPU.

Different Guidance Terms. In the sampling process of the trained diffusion model, we apply different guidance terms to enforce contact constraints to improve the realism of the generated interactions. As described in the paper, we use reconstruction guidance formulated as follows,

$$\tilde{\boldsymbol{\tau}}_0 = \hat{\boldsymbol{\tau}}_0 - \alpha \boldsymbol{\Sigma}_n \nabla_{\boldsymbol{\tau}_n} F(\hat{\boldsymbol{\tau}}_0). \tag{1}$$

F consists of three terms determined by the equation below,

$$F_{\text{all}} = \lambda_1 F_{\text{contact}} + \lambda_2 F_{\text{feet}} + \lambda_3 F_{\text{obj}}. \tag{2}$$

† indicates equal contribution.

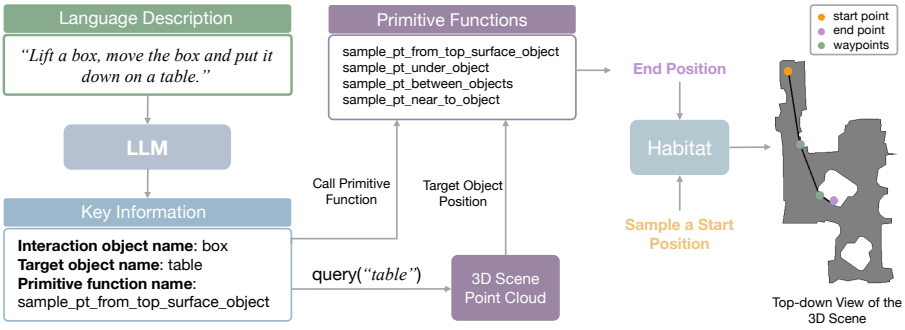

Fig. 1: Extracting the object waypoints from the language description and the 3D scene.

The loss weight for each term is determined by first experimenting with different values for a single guidance term and then adding all the terms together. We empirically use $\alpha = 1e^7, \lambda_1 = 1, \lambda_2 = 100, \lambda_3 = 30$ in our experiments.

2 Details of Interaction Synthesis in 3D Scenes

Sparse Waypoints Generation. Given a language description and a 3D scene point cloud with semantic labels, we first extract the target 3D position and then leverage the functions provided by the Habitat [4] to plan a collision-free path. We use an example to illustrate the process of extracting waypoints in Figure 1. Currently, the primitive functions consist of sampling a point near an object, sampling a point on the top surface of an object, and sampling a point under an object. This set of primitive functions can be enriched to support more complex language descriptions.

Heuristics for Preparing Waypoints Input Conditions. The waypoints extracted using Habitat do not include the corresponding time step information. To utilize these waypoints as input conditions for our interaction synthesis model, we design a heuristic-based strategy to assign appropriate frames to the waypoints such that we can have a series of waypoints distributed every 30 frames.

We start by adding 20 uniformly distributed points between every two waypoints. We compute the Euclidian distance of each pair with consecutive points denoted as $d_1, d_2, ..., d_{N-1}$, where N represents the number of all the waypoints after the insertion operation. We then define a distance range $[d_{\text{min}}, d_{\text{max}}]$ and iterate through the distance sequence. We accumulate the distance values and add the point $\boldsymbol{p}_i$ to our final waypoints list if there exists $\sum_{n=j}^{i} d_n > d_{\text{min}}, \sum_{n=j}^{i} d_n < d_{\text{max}}$, where j denotes the frame index of the previous added waypoint.

Data Preparation. We use the 3D scenes of the Replica dataset [9] to evaluate the scene-aware interaction synthesis pipeline. We focus on the evaluation of long-term interaction synthesis, thus, we discard the small-sized scenes that are not suitable for long-term sequence evaluation. To prepare the evaluation dataset for long-term interaction synthesis, we generate 20 sequences for each pair of the language description and the 3D scene. We then apply our heuristic-based strategy

described above to prepare input waypoint conditions and discard the sequences with a duration of less than 8 seconds. Finally, we obtained 165 sequences of two different scenes for evaluation. Other 3D scene datasets like HM3D [7] can also be processed using the same approach and directly used in our current pipeline.

Long Sequence Generation. Our interaction synthesis model is trained on motion data with a window size of 120 frames. To generate long-term interactions consisting of multiple windows, we adopt an interpolation strategy to smoothly connect two consecutive windows following prior work [11]. We use 10 frames as overlap frames for two consecutive windows. For a generated motion sequence $\boldsymbol{X}_1^{k-1}, \boldsymbol{X}_2^{k-1}, ..., \boldsymbol{X}_T^{k-1}$ of window $k-1$ and $\boldsymbol{X}_1^k, \boldsymbol{X}_2^k, ..., \boldsymbol{X}_T^k$ of window k, we apply interpolation on the last 10 frames of the previous window $\boldsymbol{X}_{T-9}^{k-1}, ..., \boldsymbol{X}_T^{k-1}$ and the first 10 frames of the current window $\boldsymbol{X}_1^k, ..., \boldsymbol{X}_{10}^k$. Specifically, we apply linear interpolation for the human joint positions and object positions and spherical linear interpolation (slerp) for human joint rotation and object rotation defined as follows.

$$\hat{\boldsymbol{P}}_t = lerp(\alpha_t, \boldsymbol{P}_{T-(10-t)}^{k-1}, \boldsymbol{P}_t^k), t = 1, 2, ..., 10, \tag{3}$$

$$\hat{\boldsymbol{Q}}_t = slerp(\alpha_t, \boldsymbol{Q}_{T-(10-t)}^{k-1}, \boldsymbol{Q}_t^k), t = 1, 2, ..., 10. \tag{4}$$

In the above equations, $\boldsymbol{P}_t$ represents the human joint positions and object positions, $\boldsymbol{Q}_t$ represents human joint rotation and object rotation, and the blending weights α_t are linearly decaying weights from 1 to 0.

3 Details of Evaluation Metrics

The standard text-to-motion task was explored using datasets like HumanML3D [1], which exhibit a distribution significantly different from that of the FullBody-Manipulation dataset. As a result, we cannot directly utilize their pre-trained motion and text encoders for computing R-precision and FID. Instead, we train related feature extractors using the FullBodyManipulation dataset for evaluation purposes.

Motion Autoencoder Training. Following T2M [1], we first train a motion autoencoder to represent a window of motion using latent vectors. This autoencoder comprises an encoder and a decoder, both adopting a temporal convolution model architecture. Specifically, we use global joint positions as our data representation. We canonicalize the sequence of 3D joint positions by aligning the facing direction of the first frame to a consistent direction similar to prior work [8]. Although we experimented with other data representations, such as combinations of joint rotation and positions, we found that representing both led to jittery reconstructed motions. Consequently, we solely utilize 3D joint positions for data representation based on empirical evidence.

Training Feature Extractors using Contrastive Loss. With the pre-trained motion autoencoder, we then train a motion feature extractor and a text feature extractor using contrastive loss following prior work [1]. We adopt a GRU-based model architecture for both extractors to process encoded motion vectors and

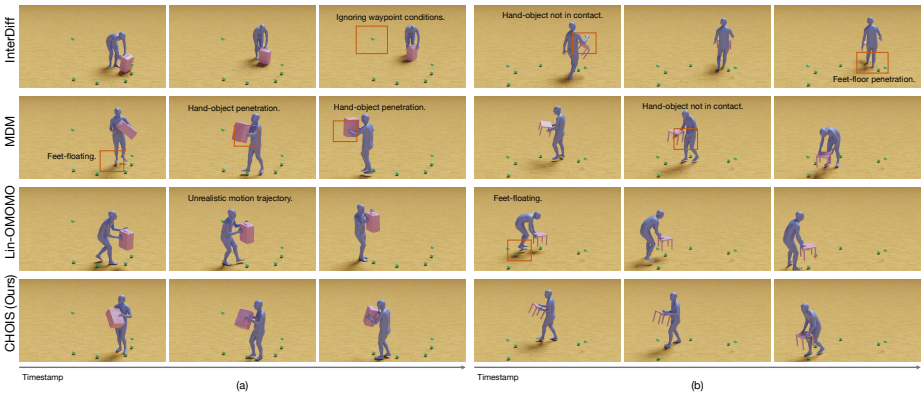

Fig. 2: Qualitative Comparisons with Baselines.

input text, respectively. The design of the contrastive loss aims to reduce the feature distance between paired text and motion while increasing it between unpaired data. Employing the trained motion autoencoder, motion feature extractor, and text feature extractor, we compute R-precision and FID following prior work on text-to-motion task [1].

4 Details of Baselines

In this section, we elaborate on how we adapt different methods to our problem setting as baselines and present extensive experiment analysis for each approach. We showcase some qualitative comparisons with baselines in Figure 2 and comparisons with OMOMO ablations in Figure 3. We encourage readers to watch our supplementary video for qualitative results.

InterDiff. InterDiff [12] addresses the task of predicting future human-object interactions based on the previous 10 frames. This task differs from ours, which synthesizes interactions based on the initial state, text, and sparse waypoints. To align InterDiff with our setting, we enhanced their model with text and sparse waypoint embeddings. Although we attempted to use only the initial frame's interaction state to align with our setting, we encountered training stability issues and thus continued using the past 10 frames, following InterDiff's original setup.

The input conditions are entangled in InterDiff since InterDiff sums up all the conditional embeddings as a single embedding vector. This might be the major reason that the generated results of InterDiff ignore the text and waypoint conditions. In addition, typical artifacts in the results of InterDiff include feet-floor penetration and severe sliding, leading to unrealistic interaction generations. Please watch our supplementary video for the qualitative comparisons.

MDM. MDM [10] focuses on synthesizing human motions from language descriptions, unlike our task, which involves synthesizing both human and object motions and incorporates conditions like sparse waypoints. To adapt MDM to our task, we incorporated our object motion representation, employing a basis

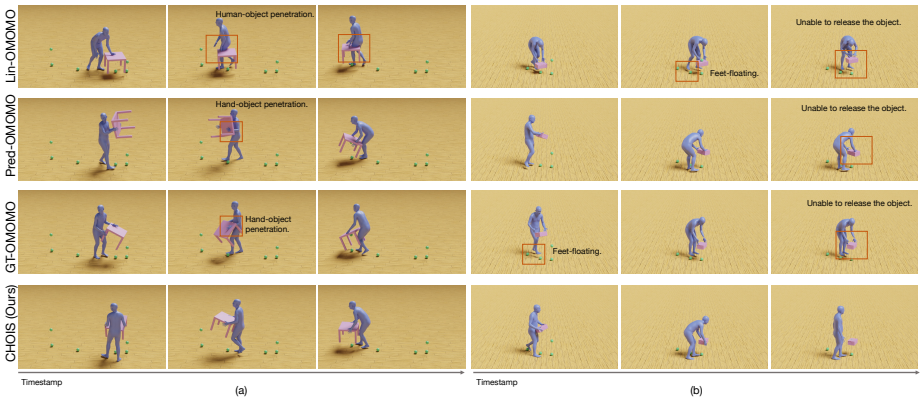

Fig. 3: Qualitative Comparisons with OMOMO Ablations.

point set (BPS) representation and an MLP to derive a low-dimensional input vector. We also included sparse waypoint embeddings and extended the model's outputs to include object rotation and positions.

MDM struggles to generate human and object motions in contact because of the lack of contact constraints. In addition, MDM applies waypoint conditions by projecting all the sparse waypoints to a single conditional vector, which is less effective compared to our mechanism used in CHOIS. Also, MDM suffers from feet-floating artifacts. Please watch our supplementary video for the qualitative comparisons.

OMOMO Ablations. OMOMO [3] is designed to synthesize human motions from a complete sequence of object states, which is less complex than our task of synthesizing both object and human motions while adhering to conditions such as text and sparse waypoints. For thorough comparisons, we adapted OMOMO into different variants: Lin-OMOMO, Pred-OMOMO, and GT-OMOMO. It is important to note that Pred-OMOMO and GT-OMOMO are not strictly baselines, as Pred-OMOMO incorporates our proposed object synthesis module, and GT-OMOMO relies on ground truth object motions.

Lin-OMOMO cannot generate realistic human motion trajectories because of the unrealistic object motion trajectory. Compared to Lin-OMOMO, Pred-OMOMO can synthesize more realistic object motions because of our carefully designed object motion synthesis module, and thus result in more plausible human motions in interaction. However, since Pred-OMOMO includes three stages, there would be accumulation errors introduced in each stage, leading to unrealistic hand joint predictions which further leads to artifacts such as hand-object penetration. GT-OMOMO provides perfect object geometry states, but it still cannot prevent unrealistic hand predictions and suffers from feet-floating issues. Furthermore, all of these OMOMO variants cannot generate release motions, as they are based on a strict assumption that once the contact frame is detected, the following frames will be in contact until the end frame. Conversely, our CHOIS model is capable

of generating motions with realistic contact and release behaviors. Please watch our supplementary video for the qualitative comparisons.

5 Discussions and Limitations

Our approach has a few limitations. First, due to the lack of detailed finger movement data in the FullBodyManipulation dataset, our method cannot produce accurate hand interactions involving finger motions. Second, it does not incorporate physical principles, which may result in objects exhibiting slightly unnatural movements before contact with a human in certain sequences. Third, our framework of synthesizing long-term interactions in 3D environments currently only allows for interactions with a single object and does not support interactions involving multiple objects. To enable such multi-object interactions, integration with a scene-aware navigation model is necessary.

References

1. Guo, C., Zou, S., Zuo, X., Wang, S., Ji, W., Li, X., Cheng, L.: Generating diverse and natural 3D human motions from text. In: CVPR (2022) 3, 4
2. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015) 1
3. Li, J., Wu, J., Liu, C.K.: Object motion guided human motion synthesis. ACM Trans. Graph. **42**(6) (2023) 1, 5
4. Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., Batra, D.: Habitat: A Platform for Embodied AI Research. In: ICCV (2019) 2
5. Mir, A., Puig, X., Kanazawa, A., Pons-Moll, G.: Generating continual human motion in diverse 3D scenes. In: 3DV (2024) 1
6. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019) 1
7. Ramakrishnan, S.K., Gokaslan, A., Wijmans, E., Maksymets, O., Clegg, A., Turner, J.M., Undersander, E., Galuba, W., Westbury, A., Chang, A.X., Savva, M., Zhao, Y., Batra, D.: Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In: NeurIPS Datasets and Benchmarks Track (2021) 3
8. Rempe, D., Birdal, T., Hertzmann, A., Yang, J., Sridhar, S., Guibas, L.J.: Humor: 3D human motion model for robust pose estimation. In: ICCV (2021) 1, 3
9. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The Replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019) 2
10. Tevet, G., Raab, S., Gordon, B., Shafir, Y., Bermano, A.H., Cohen-Or, D.: Human motion diffusion model. In: ICLR (2023) 4
11. Tseng, J., Castellon, R., Liu, C.K.: EDGE: Editable dance generation from music. In: CVPR (2023) 3
12. Xu, S., Li, Z., Wang, Y.X., Gui, L.Y.: InterDiff: Generating 3D human-object interactions with physics-informed diffusion. In: ICCV (2023) 4