– Supplementary Materials –
InsActor: Instruction-driven Physics-based Characters

Contents

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

A Simulation Environment

As mentioned in the main text, our experiments are mainly executed with Brax [2] for its differentiability. Our character model, guided by DeepMimic [6], is a humanoid with 13 links and 34 degrees of freedom, weighing 45 kg and measuring 1.62m in height. Contact is applied to all links with the floor. Facilitated by GPU-accelerated environment simulations, the physics simulator runs at 480 FPS. To optimize gradient propagation, the character's joint limits are eased. System configurations, such as friction coefficients, align with DeepMimic's parameters.

B Diffusion Policy

For the diffusion policy, we build up an 8-layer transformer as the motion decoder. As for the text encoder, we first directly use the text encoder in the CLIP ViT-B/32 [7], and then add four more transformer encoder layers. The latent dimension of the text encoder and the motion decoder are 256 and 512, respectively. As for the diffusion model, the number of diffusion steps T is 1000, and the variances β_t are linearly increased from 0.0001 to 0.02. We opt for Adam [5] as the optimizer to train the model with a 0.0002 learning rate. We use 4 NVIDIA A100 for the training, and there are 256 samples on each GPU, so the total batch size is 1024. The total number of iterations is 40K for KIT-ML and 100K for HumanML3D.

C Skill Discovery

For the skill discovery, both the encoder and decoder are a 3-layered Multi-layer Perceptron (MLP), each layer with 512 nodes. The dimension of the latent vector is 64. We choose the weight factor λ as 0.01 except for the ablation study of λ. We opt for Adam as the optimizer to train the model with a 0.0003 learning rate. We use one NVIDIA A100 for the training, and the batch size is 300. The total number of iterations is 10K.

D Baselines

Since there are no publicly available implementations of the two compared methods DReCon [1] and PADL [4], in this section, we elaborate on our implementation of the two compared approaches. In addition, we will also detail the implementation of the high-level policy and the low-level policy used in the hierarchical design ablation.

D.1 Adapted DReCon

For the kinematic controller, we directly use the pretrained diffusion planner as a replacement for the Motion Matching. For the target state-conditioned policy, we use a three-layer MLP with 512 hidden units as the neural network. The target state is input to the networks together with the current observation. The target state is normalized to be relative to the current state in the global frame. We use the training method in DiffMimic [8] and the average evaluation pose error converges to below 0.02.

D.2 Adapted PADL

We use a three-layer MLP with 512 hidden units as the neural network. A clip embedding with 512 dimensions from ViT-B/32 is input to the neural net together with the current state observation. The training follows the aforementioned procedures.

D.3 High-level Policy

We first train the skill discovery module as described above. Then, we rollout the skill discovery module on all training motion sequences to collect skill trajectories. Following MocapAct [10], we repeat the rollout 16 times with different random seeds. Then we train a diffuser on the joint representation of state and skill, as described in Diffuser [3]. Finally, we let the diffuser generate both the initial state and the following skills given a human instruction.

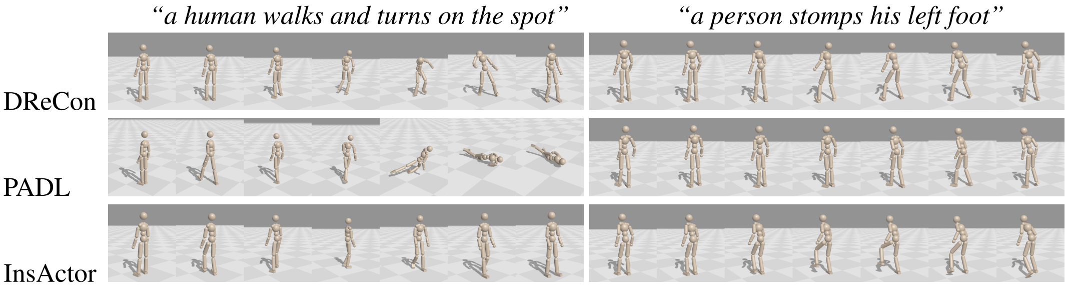
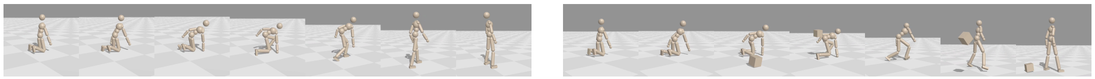

"a human walks and turns on the spot"
"a person stomps his left foot"
DReCon
PADL
InsActor

Figure 1: Quantatative comparison with corresponding instructions. Each row represents one method and column correspond to the instruction.



a) *b)*

Figure 2: Robustness test results. *a)* InsActor executes without perturbation. *b)* InsActor executes with perturbation caused by 2kg box hitting the character.

D.4 Low-level Policy

We use the same encoder-decoder architecture as described in the skill discovery module. Additionally, the language condition is encoded by a 512-dimensional CLIP ViT-B/32 embedding and input into the encoder in replace of the target state.

E Training and Inference time

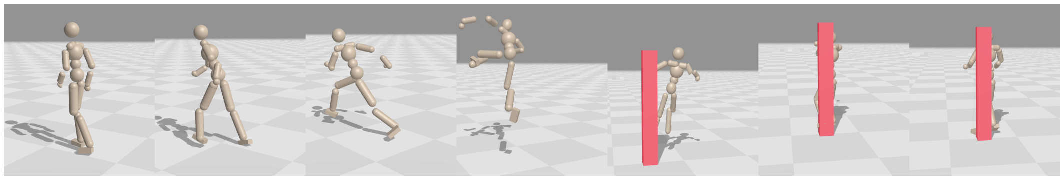
Training for the diffusion model on HumanML-3D takes approximately 16 hours on 4 NVIDIA A100 GPUs. The training for the skill discovery module on HumanML-3D takes approximately 40 hours on a single NVIDIA A100 GPU. The inference time for the diffusion planner using DDIM [9] generally takes less than a second to generate a 180-frame plan, and the skill mapping module takes less than 3 seconds to execute the plan after Just In Time compilation. We show a demo interface that runs on a single NVIDIA A100 GPU in the supplementary video for qualitative evaluation of the inference speed.

F Qualitative Results

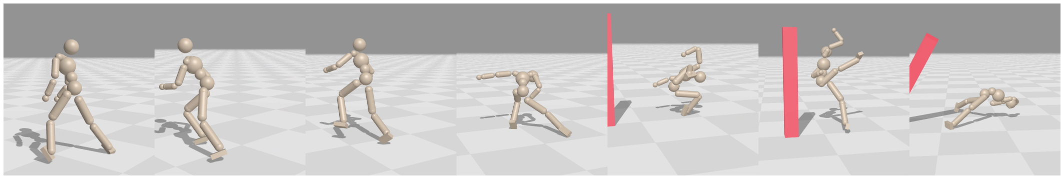
In this section, we present a detailed qualitative evaluation to further demonstrate the effectiveness of InsActor.

Qualitative Comparison with Baselines. As shown in Table 3, we conduct a qualitative comparison of InsActor with two baselines introduced in the main text: DReCon [1] and PADL [4]. The comparison is derived from the results of two different instructions, "a human walks and turns on the spot" and "person stomps his left foot". In contrast to DReCon, which fails to comprehend the high-level instructions, and PADL, which struggles to generate reliable control, InsActor exhibits the ability to successfully execute the stipulated commands.

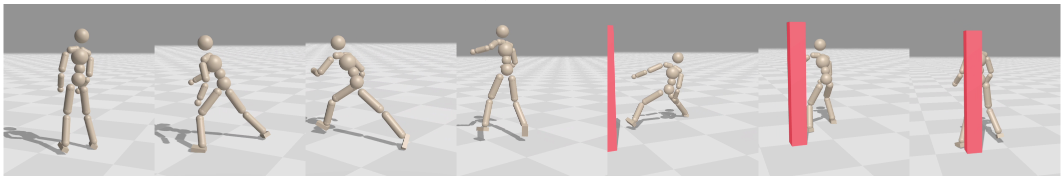
Qualitative Assessment of Robustness. In an effort to highlight the robustness of InsActor, we showcase its performance under both perturbed and non-perturbed conditions in Table 2. This involves introducing a 2kg box to strike the character at random positions. Impressively, InsActor maintains the ability to generate plausible animations under such perturbations, underscoring its resilience and adaptability in a variety of unpredictable scenarios.



a) Planned Motion



b) DReCon



c) InsActor

Figure 3: Waypoint results InsActor with corresponding waypoints. When compared with DReCon, InsActor successfully reaches the waypoint without falling and effectively follows the planned motion.

Table 1: **Quantitative results on the standard text-to-motion benchmark HumanML3D.**

Methods	R Precision↑			Multimodal Dist↓	FID↓	Diversity↑
	Top 1	Top 2	Top 3			
MDM	-	-	0.611	5.566	**0.544**	**9.559**
MotionDiffuse	0.491	0.681	**0.782**	**3.113**	0.630	9.410

Qualitative Examination of Waypoint Heading. In addition to the aforementioned analyses, we delve into a qualitative examination of waypoint heading. Compared with DReCon, InsActor successfully reaches the waypoint without falling as planned, demonstrating the flexibility and robustness of InsActor.

G Video

We show more qualitative results in the attached video. We list key timestamps as follows:

- Motion Plans - 0:45
- Random Skill Sampling - 1:04
- Comparative Study on Instruction-driven Generation - 1:11
- Robustness to Perturbations - 1:45
- Instruction-driven Waypoint Heading - 1:53
- Multiple-waypoint Following - 2:24
- Ablation on Weight Factor - 2:47
- Ablation on Hierarchical Design - 3:00
- Ablation on Instruction-driven Waypoint Heading - 3:18
- Demo Interface - 3:35

H Comparison between our diffusion planner and MDM [33]

We implemented our diffusion planner using the open-source code of MotionDiffuse [40]. We modified the feature dimensions to fit our state trajectories and changed the noise prediction to x_0 prediction for training efficiency as described in the main paper. Given that it is not possible to directly compare our motion planner to a pretrained MDM model as they have different generation space, we show a comparison between MDM and our codebase MotionDiffuse in Table 1, where MotionDiffuse achieves comparable quantitative results than MDM on a standard text-to-motion benchmark. Qualitatively, we do notice that our generated plans have more jittering than motions generated by either MDM or MotionDiffuse. This could be caused by the fact that MotionDiffuse uses temporal smoothing in the visualization but we did not smooth our plans in our visualization. We show in the following experiment that plan smoothing has a minimal effect on the tracking result.

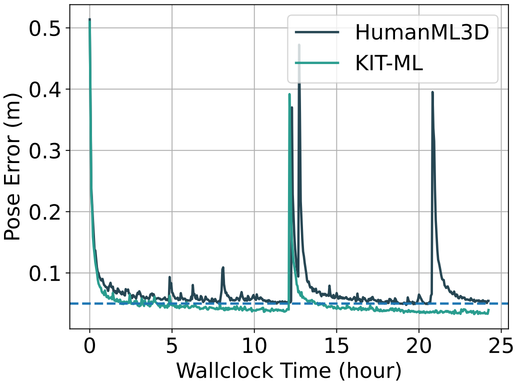


Figure 4: **Wallclock training time versus pose error for the InsActor skill mapping module training.** Blue dotted line denotes 0.05m pose error, an average DiffMmimic [28] tracking error.

Table 2: **Quantitative results on the HumanML3D test set.** Real motions: the test dataset. Planner: Our high-level planner. DReCon (Real motions): replace the generated plans with the test dataset. DReCon (Real motions): replace the generated plans with plans smoothed following [40].

Methods	R Precision↑			Multimodal Dist↓	FID↓	Diversity↑
	Top 1	Top 2	Top 3			
(a) Real motions	0.428	0.603	0.694	1.556	0.000	4.586
(b) Planner	0.434	0.625	0.723	1.507	0.314	4.538
(c) InsActor	0.331	0.497	0.598	1.971	0.566	4.165
(d) DReCon (Real motions)	0.343	0.494	0.578	2.009	0.086	4.441
(e) DReCon (Smooth)	0.268	0.391	0.463	2.594	1.271	4.092
(f) DReCon	0.265	0.391	0.470	2.570	1.244	4.070

I More ablations on planning and tracking

We show more ablation results in Table 2. Note that results in this table are not comparable with Table 1 as they are in different generation spaces. **1)** Compare (a) and (b), we observe that our diffusion planner achieves a strong generation performance. Note that our R Precision and Multimodal Dist are slightly higher than real motions since contrastive models can only give a rough estimation of the text-motion alignment. **2)** Compare (b) and (f), we observe that directly tracking the plan leads to a drastic performance drop, where InsActor (c) greatly alleviates the issue. **3)** Compare (d) and (f), we observe that our motion tracker is significantly better at tracking real motions than tracking generated plans, which verifies the performance of the DReCon motion tracker. **4)** Compare (e) and (f), we observe that although smoothing improves the visual quality of the plans, it has a minimal effect on the final result.

J Quantitative performance of low-level control

For the InsActor skill mapping module, we plot is evaluation pose error versus wallclock training time in Figure 4. Single-clip motion tracking pose error in DiffMimic [28] ranges from 0.017m to 0.097m with an average value around 0.05m. Our low-level controller successfully achieves similar control quality on large-scale motion databases.

References

[1] Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. Drecon: Data-driven responsive control of physics-based characters. *ACM Trans. Graph.*, 38(6), November 2019.

[2] C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax–a differentiable physics engine for large scale rigid body simulation. *arXiv preprint arXiv:2106.13281*, 2021.

[3] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[4] Jordan Juravsky, Yunrong Guo, Sanja Fidler, and Xue Bin Peng. Padl: Language-directed physics-based character control. Association for Computing Machinery, 2022.

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

[6] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 2018.

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[8] Jiawei Ren, Cunjun Yu, Siwei Chen, Xiao Ma, Liang Pan, and Ziwei Liu. Diffmimic: Efficient motion mimicking with differentiable physics. *ICLR*, 2022.

[9] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[10] Nolan Wagener, Andrey Kolobov, Felipe Vieira Frujeri, Ricky Loynd, Ching-An Cheng, and Matthew Hausknecht. Mocapact: A multi-task dataset for simulated humanoid control. *arXiv preprint arXiv:2208.07363*, 2022.