Supplementary Material for “Programmable Motion Generation for Open-Set Motion Control Tasks”

Hanchao Liu[1,2]* Xiaohang Zhan[2]† Shaoli Huang[2] Tai-Jiang Mu[1]† Ying Shan[2]
[1] BNRist, Tsinghua University [2] Tencent AI Lab

A. Experiment Details

A.1 Tasks for Quantitative Evaluation

We design two evaluation protocols for sub-tasks in Section 4. The first protocol is *task with known constraints*, which means the added constraints are sampled from existing human motion datasets, *i.e.*, HumanML3D test set [4] in our experiments. In this way, in addition to non-semantic motion quality metrics and constraint errors, we can evaluate on semantic-related motion quality as well since we have groundtruth motions. The second protocol is *task with unseen constraints*, which means the added constraints do not come from existing motions and are designed by ourselves to evaluate the generation capability on real open-set motion control tasks. We experiment on one sub-task for known constraints in Table 1 in the main text, and four sub-tasks for unseen constraints in Table 2 in the main text.

Task with known constraints. For Task HSI-1 *head height constraint* in Table 1, we constrain the head height for three specified key-frames, *i.e.*, first, middle and last frames to be equal to that in the motions sampled from HumanML3D test set. The text prompt and motion length for generation are also obtained from that motion sample. The constraint error for evaluation is the mean absolute error (MAE) averaged over the three key-frames. We follow PriorMDM [12] for evaluating metrics including FID, R-precision and Diversity, and follow GMD [5] for evaluating Foot skating ratio. The quantitative evaluation is conducted on 544 generated samples.

Task with unseen constraints. In Table 2, for Task HSI-2 *avoiding overhead barrier*, we constrain the head height to be lower than 0.5 m for the middle frame and higher than 1.5 m for the first and last frames to ensure normal standing poses at the beginning and the end. We also constrain the heights for both feet to be close to the ground. Note that this is a challenging task due to the low head height, and the combined constraints prevent trivial generations like *stepping on stairs* or *always lying on the ground*. The constraint error for evaluation is defined as MAE for the head height and foot heights.

For Task HSI-3 *walking inside a square*, we constrain the walkable area to be a square $-1 < x < 1, -1 < z < 1$. The constraint error for evaluation is defined as the per-joint MAE averaged over x- and z-axis and all frames.

$$\begin{aligned}\mathrm{Err}(x) = \frac{1}{4NN_j}\sum_{t=1}^{N}\sum_{j=1}^{N_j}\sum_{dim\in\mathcal{D}} \\ \max(-x^{pos}_{j,t,dim}-1,0)+\max(x^{pos}_{j,t,dim}-1,0)\end{aligned} \tag{1}$$

where $\mathcal{D}$ includes x-axis and z-axis and N_j is the number of joints.

For Task GEO-1 *hand touching wall*, we constrain the left hand (joint 20) always on a vertical plane. The plane is randomly sampled with its distance to the origin no greater than 3. The constraint error for evaluation is defined as the mean distance between the controlled hand and the given plane averaged over all frames.

For Task HOI-1 *moving object*, we constrain on the global positions of the left hand (joint 20) at the first and last key-frames. We specify a set of beginning and end hand positions. The constraint error for evaluation is defined as the mean distance between the hand and the goal averaged over the two key-frames.

For the first three tasks, *i.e.*, Task HSI-2, HSI-3 and GEO-1, the text prompts and motion lengths are sampled from a selected set of samples from HumanML3D test set, mainly involving the action *walking*. The sample ids are listed below: 000130, 000178, 000285, 000337, 000363, 000600, 000665, 000679, 000759, 000998, 000099, 000696, 000700, 003703, 001161, 001617, 001848, 003193, 003437, 004455 and their mirrored ones. For Task HOI-1, we manually compose a set of text prompts related to action *moving* such as *a person moves an object from a place to another place*. The quantitative evaluation for each unseen task is conducted on running 32 generated samples.

---

* Work done during an internship at Tencent AI Lab.

† Joint corresponding authors. `xhangzhan@tencent.com`, `taijiang@tsinghua.edu.cn`

A.2 Baseline Details

Unconstrained MDM. The original motion representation for MDM [13] contains both local joint positions and joint rotations. For simplicity we recover global joint positions (joint positions in the global coordinate) from local joint positions. The unconstrained MDM only serves as a numerical reference.

IK and IK+Reg. We implement IK as an ablated version of our method, in which the gradient ∇F is back-propagated to motion x instead of the latent vector z. We also consider a variant IK with regularization (IK+Reg.), in which we add a L2-norm regularization term on all joints $L_{reg} = |x_{[i+1]} - x_{[i]}|$, where i is the temporal index. This results in a combined error function $L_{constraint} + wL_{reg}$. We empirically set the regularization weight $w = 1.0$. We obtain global positions from joint rotations with a human skeleton template with fixed bone lengths. Like our method, IK and its variants can also handle arbitrary open-set control tasks, so we compare with IK and IK+Reg. in all quantitative and qualitative experiments.

Inpainting-based methods. MDM Edit and PriorMDM finetuned control are inpainting-based methods. They support motion control tasks by assigning exact joint trajectories. However, they cannot natively handle tasks described by constraints, especially, inequality constraints. Moreover, PriorMDM needs to finetune the network for controlling a specified joint and only finetuned models for hand, foot and root trajectories are provided [12]. For the above reasons, we only compare with MDM Edit on trajectory control-based tasks, *i.e.*, Task HSI-1, Task GEO-1 and Task HOI-1, and we compare with PriorMDM on Task GEO-1 and Task HOI-1, which only involves hand trajectory control.

In their original papers, MDM Edit and PriorMDM finetuned control only support inpainting with root trajectories and valid local joint positions. Since the constraints for tasks defined in Section 4 are majorly represented in global coordinates, we adapt MDM Edit and PriorMDM control to handle control signals in global positions. Specifically, we first generate a sample and take its root trajectory. We then use ad-hoc tricks to generate a trajectory for the control joint in global positions that satisfies the given constraint and further convert it to local positions given the root trajectory. Finally we inpaint both the root trajectory and the local trajectory of the control joint. Similar to IK, as recovering from local joint positions yields invalid bone lengths (see Table 3 in the main paper), we obtain the global motion from joint rotations using a human skeleton template with fixed bone lengths. Also, for PriorMDM we use model blending [12] for inpainting both root trajectory and control joint trajectory.

The ad-hoc tricks are designed as follows: for Task HSI-1 and HOI-1, we directly set the key-frame positions with the required constraint. For Task GEO-1, we project the generated hand trajectory onto the given plane to obtain the new hand trajectory in the global positions.

A.3 Implementation Details

Following unconstrained MDM, we also recover global positions from local joint positions. Since the error function for each task may vary, while optimizing with learning rate 0.005 and 100 optimization steps generally works well for a majority of tasks, we may also increase the initial learning rate to up to 0.05 for faster convergence in some cases. Besides, we may add regularization term using absolute position constraint to preserve desired motion characteristics for root trajectory or body parts in some cases.

Constraint relaxation. We only apply constraint relaxation on Task GEO-1, Task GEO-2 and Task HOI-1, which involves absolute position constraints of point, line and plane. It takes advantage of translation invariance of motion for fast convergence and compensates for the limited horizontal space coverage of root trajectories in the original motion prior. For Task GEO-1, we relax the plane constraint by fitting the generated hand trajectory on an optimal vertical plane. For Task GEO-2, we relax the line constraint by fitting the foot trajectories on an optimal line. For Task HOI-1, we relax the required beginning and end points A, B to fall on the line connecting the beginning and end points generated by the model $\hat{A}, \hat{B}$ and keep their middle points the same, *i.e.*, $A_{relax} = P + \frac{\hat{A}-P}{|\hat{A}-P|}\frac{|A-B|}{2}$, $B_{relax} = P + \frac{\hat{B}-P}{|\hat{B}-P|}\frac{|A-B|}{2}$, where $P = (\hat{A} + \hat{B})/2$.

In practice, we update the constraint using the aforementioned relaxation strategy every K steps and minimize the constraint error for x using the updated constraints. In this way the whole optimization process can be implemented as relax-and-minimize loops. For a fair comparison, IK and IK+Reg. also use constraint relaxation for experiments in Table 2 in the main paper.

A.4 Experiment Details for Bone Length Preserving

We provide more experimental details for Table 3 in the main paper. For the generated motions in Task HSI-1 in Table 1, we investigate the neck length (bone length between joint 12 and 15) at the key-frames where the head height constraint is imposed. We empirically set a range between 0.08-0.025 and 0.08+0.025, and the neck length which falls outside this range is considered as incorrect bone length. The bone length incorrect ratio is defined as the ratio of key-frames with incorrect neck lengths in all the generated key-frames. We find that unconstrained MDM and our method have low incorrect ratio even if we directly recover global positions from local joint positions. However, if we recover motions generated by MDM Edit from local joint positions, the incorrect ratio becomes very large, indicating that a great percentage of the generated samples

Task GEO-1: hand touching wall			
Method	Foot Skate	Max Acc.	C.Err.
IK w/o relax.	0.375	0.209	0.210
IK w/ relax.	0.187	0.147	**0.010**
Ours w/o relax.	0.094	0.129	0.118
Ours w/ relax.	0.110	0.104	**0.023**
Task HOI-1: moving object			
Method	Foot Skate	Max Acc.	C.Err.
Ours w/o relax.	0.078	0.077	0.069
Ours w/ relax.	0.114	0.068	**0.028**

Table A1. Effect of constraint relaxation. Constraint relaxation helps better reach constraints related to horizontal positions for optimization-based methods.

Task HSI-1: head height constraint				
Method	Foot Skate	Diversity	FID	C.Err.
MDM (Unconstrained)	0.086	9.656	0.545	0.118
Ours ($N_S = 1$)	0.075	9.611	0.556	0.012
Ours ($N_S = 5$)	0.072	9.422	0.648	**0.002**

Table A2. Effect of initial point search. N_S denotes the number of searches. Using a random initial point search leads to significantly smaller constraint error. It provides a solution for generating motions that better adhere to the given constraint.

are of invalid human layouts. For this reason, we choose to recover global motion from joint rotations for inpainting-based methods MDM Edit and PriorMDM.

A.5 Additional Analysis

Effect of constraint relaxation. As in Table A1, the constraint relaxation strategy significantly reduces the constraint error for goal reaching tasks on the horizontal plane, such as task *hand touching wall* and *moving object*. While the constraints are better satisfied, we observe slight decrease in motion quality, which is indicated by Foot Skate. Also, it is shown that the constraint relaxation is a general optimization strategy since there is a significant decrease in the constraint error for IK as well.

Effect of initial point search. The initial noise z may affect the final constraint error if the initialized motion is too far away from reaching the constraints. A straightforward way would be to sample random noise z in several runs and pick the result with the smallest constraint error. We conduct experiment on Task HSI-1 using the same setting as Table 1 in the main paper and compare the results of $N_S = 1$ and $N_S = 5$. Here N_S denotes the number of initial point searches. The results are shown in Table A2. We observe that using a random initial point search leads to significantly smaller constraint error but at the cost of diversity and FID scores. It provides a solution for generating motions that better adhere to the given constraint.

Diversity of generated motions. By optimizing the latent vector of generated motions to conform to the motion prior, our method can generate diverse motions under the same constraint. For example, in the task of *left hand always touching head*, apart from single hand touching the face, we observe that constraining only one hand can also give rise to the touching of another hand. (see Fig. 1 and Fig. 4 in the main paper).

B. Details for Motion Programming by LLM

Our programmable motion generation framework also makes automatic programming possible with the aid of large language models (LLM). As in Fig. B1, in order to generate code for the error function F, we first feed instructions to GPT [2] with the rules and ingredients for motion programming, e.g., input arguments and functions in the atomic constraint library. After that, one can feed the textual description for an arbitrary open-set motion control task to GPT. In Fig. B1 we show the textual input fed to GPT as well as the raw code output by GPT for Task GEO-1, HSI-3 and HSI-4 in the main paper. We observe that an LLM can pick correct atomic constraints, logical operations (e.g. “$>$”, “$<$”), and procedural operations (e.g. if-else clauses) for given tasks. Note that the code blocks labeled with GPT markers for Task GEO-1 and Task HSI-4 in Fig. 4 in the main paper are slightly modified in the coding style to make them consistent with other manually written code, without changing the code logic.

More evaluation. As in Table A3, we design 20 unique tasks (including those presented in the main paper), and evaluate the success rate of LLM programming via comparing to manual programming. With little prompt engineering, the success rate turns out to be 14/20. In failure cases, it typically picks incorrect inequality logical operations, or provides excessive and incorrect physical constraints. Nevertheless, we find that LLM comes up with novel constraints beyond manual programming, e.g. tilt angle constraint for the action *balancing*.

C. Discussion and Limitations

Sources of error. As we propose a general framework for open-set motion control tasks, the performance of individual modules can be further improved. First, we observe some unrealistic poses and motion artifacts in our generated motions. Since the FID score shows that our results have similar quality to unconstrained MDM (See Table 1 in the main paper), a possible solution is to enlarge the pre-trained model together with more training data. Also, for complex tasks, either an end user or an LLM may have diffi-

Evaluation tasks	
walking with hand always touching face.	✓
walking inside a square.	✓
carrying a ball.	
carrying a heavy ball.	
walking with feet on a straight line.	✓
walking with hand touching a wall.	✓
walking in a gap between two walls.	✓
walking to avoid an overhead barrier.	✓
picking object from A to B.	✓
walking with velocity constraint on three frames.	✓
standing and keeping balanced with single foot.	
walking with head height constraint on three frames.	✓
lying on a bed.	✓
sitting on a chair.	
kicking a ball in the last frame.	✓
walking with both hands in contact.	✓
jumping over a barrier.	
pointing to a direction with left arm.	✓
dancing with specified velocity magnitude on three frames.	✓
twisting for two circles.	

Table A3. Evaluation on motion programming by LLM. Tasks that are successfully handled by LLM are labeled with ✓.

culty of crafting detailed and appropriate constraints, which is likely to lead to unnatural motions. Second, the constraint error sometimes remains big compared to IK, for example, for the unseen Task HSI-2. Although it is reasonable that IK directly optimizes on motion x and thus has less difficulty for reaching the constraint, we will further investigate better optimization approaches to solve this issue. Possible solutions include (1) combining optimized and IK-based motion in the denoising process, (2) relaxing on the parameters of the generation model and involving it in the optimization process like [8], and (3) searching for more suitable optimizers.

Moreover, the action semantics for the generated motion is observed to change slightly in the experiments, e.g. for Task HSI-1. This calls for more suitable generation models and optimization strategies that can better adhere to the text condition.

Coverage of the proposed constraint library. We examine the coverage of our proposed library for daily motions on BABEL-120 dataset [11]. We find that nearly 16% of the actions involve periodic, rotational or symmetric movement, whose control is not directly supported by our library. We plan to further add frequency-domain, rotational and symmetric constraints into our library.

Comparison with reinforcement learning and trajectory optimization. We note that RL-based [6, 9, 10] and trajectory optimization approaches [1] also build compositional reward or goal functions for specialized motion control tasks, and we here provide a discussion for these approaches: (1) Based on our experiments, the error function design in this work is not as difficult as reward design in reinforcement learning (RL), not only because the error function only handles the constraints, but the optimization in latent space is easier to converge than RL training, since the pre-trained model already provides a neat and smooth manifold as the optimization space. (2) The pre-trained generation model is easier to accommodate more motion skills and scale up with more data. This is the main consideration for us towards solving open-set tasks. RL usually requires specific design to support diverse tasks [14]. (3) It is easier than RL to control the semantics via text condition. (4) RL and ours are complementary. RL has better physics-grounded qualities. (5) Compared to trajectory optimization, optimizing latent code better preserves semantics imposed by text condition. Besides, optimizing latent code may be more advantageous for composing novel types of actions since it acts like semantic interpolation in the data distribution. Trajectory optimization normally optimizes on one reference motion [3, 7].

Time performance. Currently it costs a few minutes for each customized task, but is still much better than previous works that require collecting new data and training new networks. We have not focused a lot on improving optimization efficiency in this work, which might be a direction in the follow-up works. Although not applicable to real-time generation, it is suitable for off-line content creation due to its high customizability.

References

[1] Mazen Al Borno, Martin De Lasa, and Aaron Hertzmann. Trajectory optimization for full-body movements with complex contacts. *IEEE transactions on visualization and computer graphics*, 19(8):1405–1414, 2012. 4

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 3

[3] Erik Gärtner, Mykhaylo Andriluka, Hongyi Xu, and Cristian Sminchisescu. Trajectory optimization for physics-based reconstruction of 3d human pose from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13106–13115, 2022. 4

[4] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5152–5161, 2022. 1

[5] Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2151–2162, 2023. 1

[6] Ariel Kwiatkowski, Eduardo Alvarado, Vicky Kalogeiton, C Karen Liu, Julien Pettré, Michiel van de Panne, and Marie-Paule Cani. A survey on reinforcement learning methods in character animation. In *Computer Graphics Forum*, pages 613–639. Wiley Online Library, 2022. 4

[7] Libin Liu and Jessica Hodgins. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 4

[8] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7474–7489, 2021. 4

[9] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017. 4

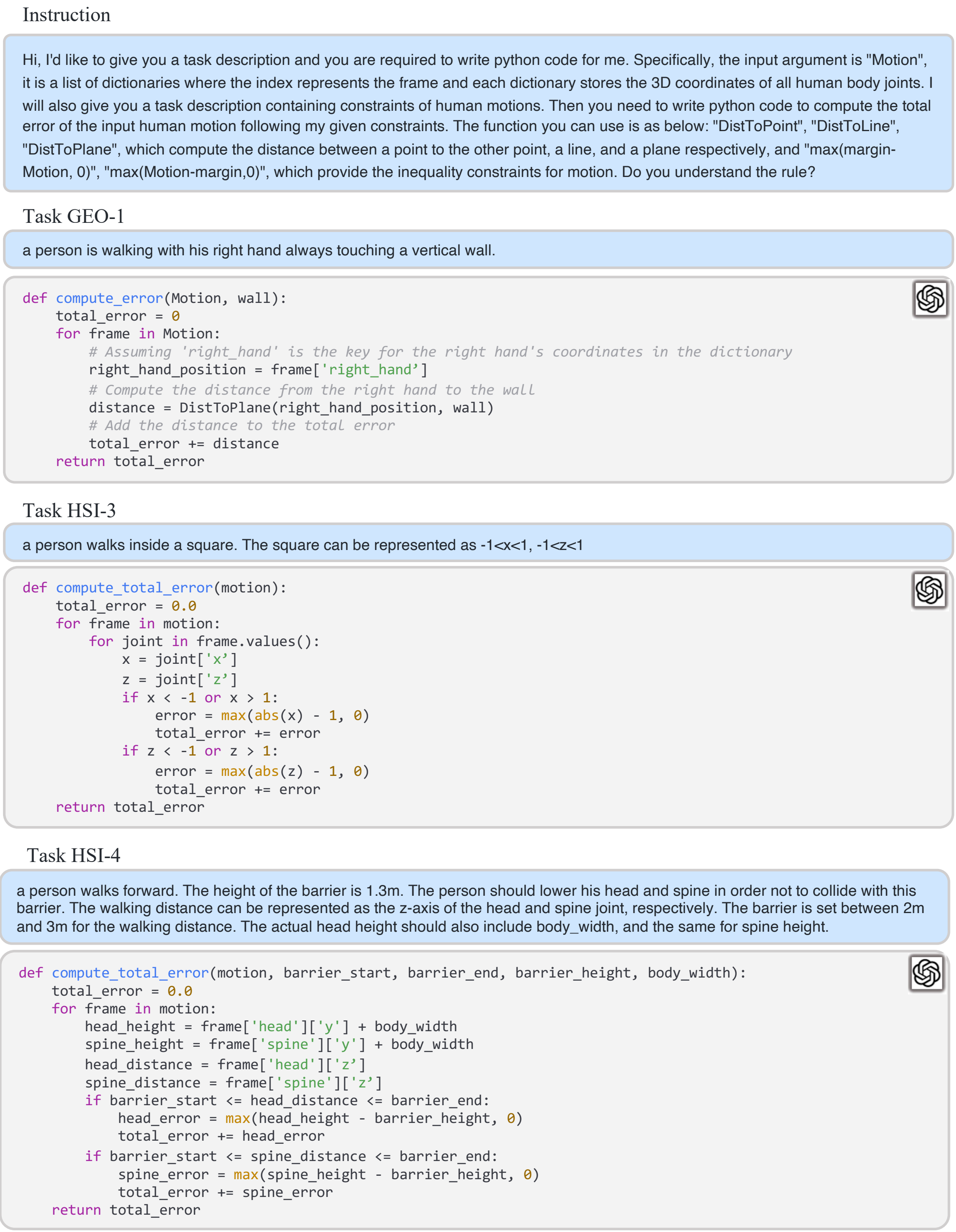
[10] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018. 4

[11] Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. Babel: Bodies, action and behavior with english labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 722–731, 2021. 4

[12] Yoni Shafir, Guy Tevet, Roy Kapon, and Amit Haim Bermano. Human motion diffusion as a generative prior. In *The Twelfth International Conference on Learning Representations*, 2023. 1, 2

[13] Guy Tevet, Sigal Raab, Brian Gordon, Yoni Shafir, Daniel Cohen-or, and Amit Haim Bermano. Human motion diffusion model. In *The Eleventh International Conference on Learning Representations*, 2022. 2

[14] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. A scalable approach to control diverse behaviors for physically simulated characters. *ACM Transactions on Graphics (TOG)*, 39(4):33–1, 2020. 4



Instruction

Hi, I'd like to give you a task description and you are required to write python code for me. Specifically, the input argument is "Motion", it is a list of dictionaries where the index represents the frame and each dictionary stores the 3D coordinates of all human body joints. I will also give you a task description containing constraints of human motions. Then you need to write python code to compute the total error of the input human motion following my given constraints. The function you can use is as below: "DistToPoint", "DistToLine", "DistToPlane", which compute the distance between a point to the other point, a line, and a plane respectively, and "max(margin-Motion, 0)", "max(Motion-margin,0)", which provide the inequality constraints for motion. Do you understand the rule?

Task GEO-1

a person is walking with his right hand always touching a vertical wall.

```python
def compute_error(Motion, wall):
    total_error = 0
    for frame in Motion:
        # Assuming 'right_hand' is the key for the right hand's coordinates in the dictionary
        right_hand_position = frame['right_hand’]
        # Compute the distance from the right hand to the wall
        distance = DistToPlane(right_hand_position, wall)
        # Add the distance to the total error
        total_error += distance
    return total_error
```

Task HSI-3

a person walks inside a square. The square can be represented as -1<x<1, -1<z<1

```python
def compute_total_error(motion):
    total_error = 0.0
    for frame in motion:
        for joint in frame.values():
            x = joint['x’]
            z = joint['z’]
            if x < -1 or x > 1:
                error = max(abs(x) - 1, 0)
                total_error += error
            if z < -1 or z > 1:
                error = max(abs(z) - 1, 0)
                total_error += error
    return total_error
```

Task HSI-4

a person walks forward. The height of the barrier is 1.3m. The person should lower his head and spine in order not to collide with this barrier. The walking distance can be represented as the z-axis of the head and spine joint, respectively. The barrier is set between 2m and 3m for the walking distance. The actual head height should also include body_width, and the same for spine height.

```python
def compute_total_error(motion, barrier_start, barrier_end, barrier_height, body_width):
    total_error = 0.0
    for frame in motion:
        head_height = frame['head']['y'] + body_width
        spine_height = frame['spine']['y'] + body_width
        head_distance = frame['head']['z’]
        spine_distance = frame['spine']['z’]
        if barrier_start <= head_distance <= barrier_end:
            head_error = max(head_height - barrier_height, 0)
            total_error += head_error
        if barrier_start <= spine_distance <= barrier_end:
            spine_error = max(spine_height - barrier_height, 0)
            total_error += spine_error
    return total_error
```

Figure B1. Motion Programming by LLM. After feeding the instruction to GPT, we provide the textual description for an arbitrary open-set motion control task. GPT will output code for the corresponding error function. We observe that GPT understands concept like *touching wall* by picking the correct *distToPlane* constraint, and picks correct inequality operations for tasks like *avoiding overhead barrier* and *walking inside a square*.