Plan, Posture and Go: Towards Open-vocabulary Text-to-Motion Generation

Supplementary Material

In this supplementary material, we provide additional details and experiments not included in the main paper due to limitations in space.

- Appendix A: Qualitative results of precise control for motion generation in our formulation.
- Appendix B: Details of the motion representations.
- Appendix C: Details of the prompt engineering and the entire system message template.
- Appendix D: Details of our dual-diffusion models.
- Appendix E: Details of the evaluation metrics.

Note: Blue characters denote the main paper's reference.

A Precise Pose Control

As shown in Fig. 1, our *Posture-denoiser* module can accurately generate poses from edited pose descriptions, which becomes particularly beneficial when users are not content with the key pose descriptions produced by the *Motion Planner*. In such cases, users have the option to manually edit the description of specific body parts to gain precise control over the generated poses. For example, in the Control #1 of Pose Description #1, we replace the description of "their left arm is behind the right with their right elbow bent at 90 degrees." with "the left hand is beside the right arm and the right hand is to the left of the left hand". It's obvious the position of the hands changes correctly. As well, when we delete the description of "The left foot is stretched backwards and is behind the right foot" in the Control #1 of Pose Description #2, the position of the left foot changes correctly. Furthermore, we could find that although the scripts are simple and limited to a small space, they are expressive to cover all possible postures due to their compositional nature.

B Motion Representations

Skinned Multi-Person Linear model (SMPL) [8] [13] is a skinned vertex-based model that accurately represents various body shapes in natural human poses. It's widely utilized in the human motion generation field. So we adopted the SMPL-based human model as well. SMPL begins with an artist-created mesh with N = 6890 vertices and K = 23 joints. The mesh has the same topology for men and women, spatially varying resolution, a clean quad structure, a segmentation into parts, initial blend weights, and a skeletal rig. We follow the SMPL-based representation of TEMOS [10] and construct the feature vector.

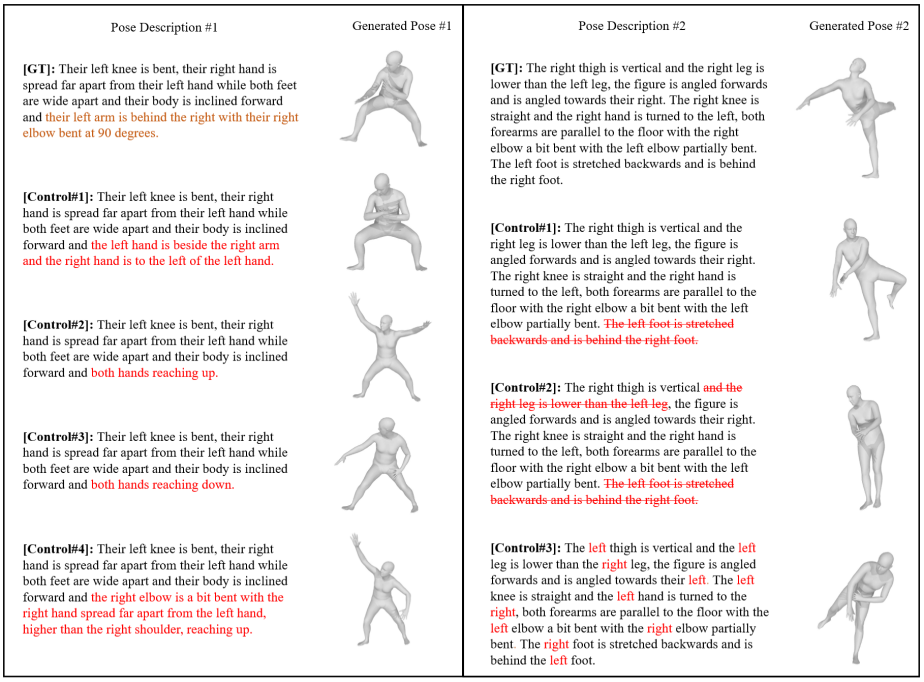

Fig. 1: Examples of Precise Pose Control. In the first example #1, we control the position of hands, with the original and modified hand descriptions represented in brown and red respectively. In the second example #2, "The" represents deleting the description. To determine whether the description of a body part is the reason for the proper positioning of that body part, we removed the corresponding descriptions to see if the body joints would change as a result of this operation. We test the model's understanding of the left and right sides of the body.

- **Translation.** It consists of two parts. The first part is the velocity of the root joint in the global coordinate system. The second part is the position of the root joint for the Z axis. The dimension of the translation is 3.
- **Root Orientation.** It contains one rotation, and we utilize the 6D continuous representation [15] to store it. So the dimension of the root orientation is 1 * 6 = 6.
- **Pose Body.** The motion data is from the SMPL-H [8,13] version of AMASS. Because we focus on the movement of the human body, we removed all rotations on the hands, resulting in 21 rotations). The same as root orientation, we utilize the 6D continuous representation [15]. So the dimension of the pose body is 21 * 6 = 126.

So the final dimension of the feature is $3 + 1 * 6 + 21 * 6 = 135$. In our experiments, we remove all the motion sequences that are less than 64 frames, and the motion sequences longer than 64 frames are processed as 64 frames. The data sample will be in $\mathbb{R}^{64*135}$.

```
# read motion
motion_parms = {
    'trans': motion[:, :3], # controls the global body position
    'root_orient': motion[:, 3 : 3 + 6], # controls the global root
    ↪ orientation
    'pose_body': motion[:, 9 : 9 + 21 * 6], # controls the body
}
```

Listing 1: motion data

When we perform the translation and rotation prediction, we only know the local attributes of the adjacent poses, but not their global location information. So we must normalize the pose position. Following [10], the translation of neighboring poses is subtracted and represented by the instantaneous velocity as the translation attribute of the current frame.

```
# extract the root gravity axis
# for smpl it is the last coordinate
root = trans[..., 2]
trajectory = trans[..., [0, 1]]

# Comoute the difference of trajectory (for X and Y axis)
vel_trajectory = torch.diff(trajectory, dim=-2)
# 0 for the first one => keep the dimentionality
vel_trajectory = torch.cat((0 * vel_trajectory[..., [0], :],
    ↪ vel_trajectory), dim=-2)
```

Listing 2: translation normalization

C Prompt Design

Fig. 2 illustrates the complete prompt used by our *Motion Planner* in Sec. 3.1. We first define the overall objective and task requirements and then establish

Table 1: Parameterizations of the modules

Posture-denoiser	Pose Size	126	Go-denoiser	Key Poses Size	F * 126
	Condition DIM.	768		Condition DIM.	256
	Latent DIM.	512		Latent DIM.	512
	Resnet & Att. DIM.	512		Motion Size	N * 135
Posture Planning	Text & Pose Encoder	512		Att. DIM.	512

five fundamental rules of body parts including bending degree, relative distance, relative position, orientation, and ground contact. Additionally, we specify the body parts to which each rule applies. Through this rule-based approach, we can guide the LLM to generate precise key pose descriptions, achieving fine-grained control over poses. Next, we define the output format of the LLM and provide examples of pose descriptions for the LLM to reference. These examples are sourced from Posescript [2]. Finally, a user prompt of motion description is provided to instruct the LLM to generate the key pose descriptions. By harnessing the powerful LLM, the user prompts are no longer confined to specific formats but are open-vocabulary, allowing for expressions like *"Jump on one foot"* and *"Experiencing a profound sense of joy"*.

Go-denoiser. We employ an AdamW [9] optimizer with a batch size of 64 and a learning rate of 1e-4. The latent dim is 512, the number of transformer layers is 8, and the number of heads is 4. We utilize GELU [4] as the activation function and the dropout is 0.1. The mask probability of the model's condition is 0.1. We use the cosine schedule and the number of diffusion steps is 100.

D Implementation Details

Posture-Denoiser. We employ an AdamW [9] optimizer for 1000 epochs with a batch size of 512 and a learning rate of 1e-4. The number of layers is $N = 3$ and the latent dim is set to 512. Following [6], we use the *linear* schedule, where $\beta_{start} = 0.00085$ and $\beta_{end} = 0.012$. Our model is trained using poses with automatically generated captions (PoseScript-A [2]).

Posture Planning. For the two encoders used in this module, we use the pre-trained text-to-pose retrieval model from [2], which is trained on PoseScript-A using the batch-based classification loss [14].

Go-Denoiser. We employ an AdamW [9] optimizer with a batch size of 64 and a learning rate of 1e-4. The latent dim is 512, the number of transformer layers is 8, and the number of heads is 4. We utilize GELU [4] as the activation function and the dropout is 0.1. The mask probability of the model's condition is 0.1. We use the cosine schedule and the number of sampling steps is 100.

Please refer to Tab. 1 for the Parameterizations of the modules.

You are a helpful assistant that help me to describe the body pose in a motion. I will give you a motion description. Envision a motion scene and create eight distinct pose descriptions at a frame rate of 0.5 fps. Ensure that each description is self-contained. The difference between two adjacent descriptions must be small, considering the small interval. Your response should be detailed as possible. Before you write each description, you must follow these instructions. These are primary rules:

Rule 1. Characterize the degree of bending of body parts.

1.1 You should select the description word from the list: ['completely bent', 'almost completely bent', 'bent at right angle', 'partially bent', 'slightly bent', 'straight'].

1.2 You should select the body part form the list: [left knee, right knee, left elbow, right elbow].

Rule 2. Classify the relative distances between different body parts.

2.1 You should select the description word from the list: ['close', 'shoulder width apart', 'spread apart', 'wide apart'].

2.2 You must compare the distances between these body part pairs as much as possible:
(left elbow, right elbow), (left hand, right hand), (left knee, right knee), (left foot, right foot),
(left hand, left shoulder), (left hand, right shoulder), (right hand, left shoulder), (right hand, right shoulder),
(left hand, right elbow), (right hand, left elbow), (left hand, left knee), (left hand, right knee), (right hand, left knee),
(right hand, right knee), (left hand, left ankle), (left hand, right ankle), (right hand, left ankle), (right hand, right ankle),
(left hand, left foot), (left hand, right foot), (right hand, left foot), (right hand, right foot)

Rule 3. Describe the relative positions of different body parts.

3.1 For the front-back direction, you should select the description word from the list: ['behind', 'in front of'].
For the up-down direction, you should select the description word from the list: ['below', 'above'].
For the left-right direction, you should select the description word from the list: ['at the right of', 'at the left of'].

3.2 You must compare the relative positioning between these body part pairs as much as possible:
(left shoulder, right shoulder), (left elbow, right elbow), (left hand, right hand), (left knee, right knee), (right foot, right foot),
(neck, pelvis), (left ankle, neck), (right ankle, neck), (left hip, left knee), (right hip, right knee), (left hand, left shoulder),
(right hand, right shoulder), (left foot, left hip), (right foot, right hip), (left wrist, neck), (right wrist, neck),
(left hand, left hip), (right hand, right hip), (left hand, torso), (right hand, torso), (left foot, torso), (right foot, torso)

Rule 4. Determine whether a body part is oriented 'vertical' or 'horizontal'.

4.1 You should select the description word from the list: ['vertical', 'horizontal'].

4.2 You need to determine as much as possible whether the body limb formed by the following pairs of body parts is 'vertical' or 'horizontal':
(left hip, left knee), (right hip, right knee), (left knee, left ankle), (right knee, right ankle),
(left shoulder, left elbow), (right shoulder, right elbow), (left elbow, left wrist), (right elbow, right wrist),
(pelvis, left shoulder), (pelvis, right shoulder), (pelvis, neck)

Rule 5. Identify whether a body part is in contact with the ground.

5.1 You should select the description word from the list: ['on the ground'].

5.2 You should select the body part form the list: [left knee, right knee, left foot, right foot].

You should write all the pose description together. The response should follow the format:
{"F1":"pose description",\n"F2":"pose description",\n"F3":"pose description",\n"F4":"pose description",\n "F5":"pose description",\n"F6":"pose description",\n"F7":"pose description",\n"F8":"pose description"}

Some sample pose descriptions are as follows:
"The right knee is unbent with the right leg next to the left while both hands are apart wider than shoulder width. The right upper arm, the left leg, the torso and the right thigh are straightened up while the right elbow is bent a bit. "
"The figure is in a yoga pose. Their left elbow is nearly bent and their hands are underneath both hips while their left knee is bent at 90 degrees while the figure is leaning backwards and their feet are front with their right elbow partially bent. Their left hand is further than shoulder width apart from the other. "
"Their left calf is straightened up with their left shoulder above their right shoulder while their right knee is straight while their left elbow and their left knee are a bit bent. Their knees are approximately shoulder width apart and the figure is angled towards the right side. Their left hand is reaching up and to the right of their right hand. It is in the right direction while their right hand is lying beneath their right hip, close to their right knee. "

Prompt: {User Prompt}

Fig. 2: The complete prompt for our *Motion Planner* in Sec. 3.1.

E Evaluation details

We will detail our evaluation metrics on the Experiments part in this section. In open-vocabulary text-to-motion experiments, we employ sentence transformers [11, 12] to compute the similarity between the text descriptions in IDEA-400 [7], a high-quality motion language subset within Motion-X, and the text descriptions in HumanML3D. We filter out pairs with similarity greater than a specified threshold α, *e.g.*, 0.45, yielding a motion language dataset comprising 368 text-motion pairs as our first test dataset.

```python
# calculate the similarity of motion descriptions and filter out pairs with similarity greater than a specified threshold
import torch
import torch.nn as nn
import torch.nn.functional as F
from transformers import AutoTokenizer, AutoModel, util

def mean_pooling(model_output, attention_mask):
    token_embeddings = model_output[0]  # First element of model_output contains all token embeddings
    input_mask_expanded = attention_mask.unsqueeze(-1).expand(token_embeddings.size()).float()
    return torch.sum(token_embeddings * input_mask_expanded, 1) / torch.clamp(input_mask_expanded.sum(1), min=1e-9)

# calculate the features of descriptions
class TextToSen(nn.Module):
    def __init__(self, device):
        super().__init__()
        self.device = device
        self.tokenizer = AutoTokenizer.from_pretrained('sentence-transformers/all-mpnet-base-v2')
        self.model = AutoModel.from_pretrained('sentence-transformers/all-mpnet-base-v2').eval().to(self.device)

    def forward(self, sentences):
        encoded_input = self.tokenizer(sentences, padding=True, truncation=True, return_tensors='pt').to(self.device)

        # Compute token embeddings
        with torch.no_grad():
            model_output = self.model(encoded_input)

        # Perform pooling
        sentence_embeddings = mean_pooling(model_output, encoded_input['attention_mask'])

        # Normalize embeddings
        sentence_embeddings = F.normalize(sentence_embeddings, p=2, dim=1)
```

```
        return sentence_embeddings

# calculate the similarity
similarity_score = util.pytorch_cos_sim(embeddings1, embeddings2)

if similarity_score > threshold:
    False
```

Listing 3: similarity filter

R-Precision is determined by ranking the Euclidean distances between the motion and text embeddings, given one motion sequence and K text descriptions (1 ground-truth and $K-1$ randomly selected mismatched descriptions).

Frechet Inception Distance (FID) measures the distributional difference between the generated and real motion by applying FID [5] to the extracted motion features.

Multimodal Distance (MM-Dist) is calculated as the average Euclidean distance between each text feature and the corresponding generated motion feature derived from that text.

In go-denoiser experiments, we utilize Average Position Error (APE) and Average Variance Error (AVE) to evaluate our methods [10]. We report the listed four metrics in Tab. 3.

Root joint errors. Take the 3 coordinates of the root joint.

Global trajectory errors. Tak3 only the X and Y coordinates of the root joint. It is the red trajectory on the ground in the visualizations of Fig. 7.

Mean local errors. Average the joint errors in the body's local coordinate system,

Mean global errors. Average the joint errors in the global coordinate system.

As in JL2P [1], Ghosh et al. [3] and TEMOS [10], the APE for a specific joint j is determined by computing the mean of the L2 distances between the generated and ground truth joint positions across the frames (F) and samples (N):

$$APE[j] = \frac{1}{NF} \sum_{n \in N} \sum_{f \in F} \left\| H_f\left[j\right] - \hat{H}_f\left[j\right] \right\|_2 \tag{1}$$

As introduced in Ghosh et al [3] and TEMOS [10], the Average Variance Error (AVE), quantifies the distinction in variations. This metric is defined as the mean of the L2 distances between the generated and ground truth variances for the joint j.

$$AVE[j] = \frac{1}{N} \sum_{n \in N} \left\| \delta\left[j\right] - \hat{\delta}\left[j\right] \right\|_2 \tag{2}$$

where,

$$\delta\left[j\right] = \frac{1}{F-1} \sum_{f \in F} \left(H_f\left[j\right] - \tilde{H}_f\left[j\right] \right)^2 \in R^3 \tag{3}$$

denotes the variance of the joint j.

References

1. Ahuja, C., Morency, L.P.: Language2pose: Natural language grounded pose forecasting. In: 3DV (2019)
2. Delmas, G., Weinzaepfel, P., Lucas, T., Moreno-Noguer, F., Rogez, G.: Posescript: 3d human poses from natural language. In: ECCV. pp. 346–362 (2022)
3. Ghosh, A., Cheema, N., Oguz, C., Theobalt, C., Slusallek, P.: Synthesis of compositional animations from textual descriptions. In: ICCV (2021)
4. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016)
5. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. NIPS (2017)
6. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NIPS **33**, 6840–6851 (2020)
7. Lin, J., Zeng, A., Lu, S., Cai, Y., Zhang, R., Wang, H., Zhang, L.: Motion-x: A large-scale 3d expressive whole-body human motion dataset. Advances in Neural Information Processing Systems (2023)
8. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: A skinned multi-person linear model. SIGGRAPH Asia (2015)
9. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
10. Petrovich, M., Black, M.J., Varol, G.: Temos: Generating diverse human motions from textual descriptions. In: ECCV. pp. 480–497 (2022)
11. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (2019)
12. Reimers, N., Gurevych, I.: Making monolingual sentence embeddings multilingual using knowledge distillation. In: EMNLP (2020)
13. Romero, J., Tzionas, D., Black, M.J.: Embodied hands: Modeling and capturing hands and bodies together. SIGGRAPH Asia **36**(6) (2017)
14. Vo, N., Jiang, L., Sun, C., Murphy, K., Li, L.J., Fei-Fei, L., Hays, J.: Composing text and image for image retrieval-an empirical odyssey. In: CVPR. pp. 6439–6448 (2019)
15. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: CVPR (2019)