APPENDIX

TM2T: Stochastic and Tokenized Modeling for the Reciprocal Generation of 3D Human Motions and Texts

Abstract. This supplementary provides more details on data pre-process, implementations details, evaluation metrics, baseline implementations, AMT user study, motion token contexts, text-based modification, inference time analysis and network architecture.

A Data Preprocess

Pose Representation. For pose representation, we extract root angular velocity, root linear velocities, root height, local joint positions, velocities, 6D rotations [16] and foot contacts from raw motions as in [5]. This results in 263 and 251 dimensional pose vectors for HumanML3D (22-joint skeleton) and KIT-ML (21-joint skeleton) dataset respectively. After all, Z-score normalization is applied to both datasets.

During training motion quantization model, to mitigate foot sliding phenomenon, the decoder D is asked to additionally predict foot contact information which is not provided to the encoder E. We also scale the magnitude of root angular velocity, root linear velocities, root height and foot contacts by a value of 5 to amplify their importance. To improve the robustness of our approach, during learning motion2text and text2motion models, we randomly cutting off 0 to 4 frames at the head or tail of pose sequences, which increases the data variance while not scarifying the quality.

B Implementation Details

Our framework is implemented by PyTorch. Our codebook $\mathcal{B}$ contains 1024 1024-dimentional embedding vectors. Encoder and decoder in motion quantization are two 1D convolutional/upsampling layers with resblocks. Weighting factor β is set to 1. Transformers for motion2text and text2motion have 4 and 3 attention layers respectively, both with 8 attention heads with 512 hidden size. The GRU based text2motion model have encoder with hidden size of 512 while the decoder is modeled as 1-layer GRU with hidden size of 1024. This GRU model is trained with teacher force ratio of 0.4. Bidirectional GRUs with hidden size 1024 are used for motion & text

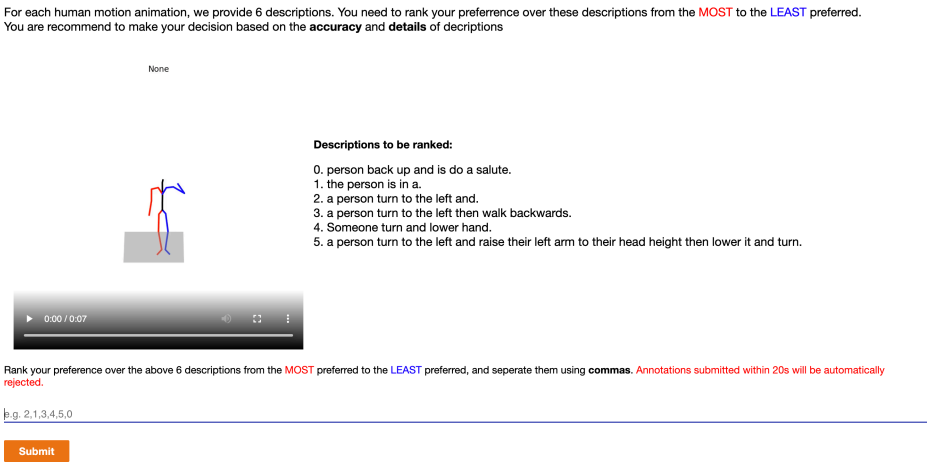

Fig. 1. User study interface for evaluating motion2text on Amazon Mechanical Turk.

feature extractors. Adam is used for all experiments with learning rate of 0.0002. We use the codebase NLPEval [1] to calculate linguistic metrics (e.g., Bleu, Rouge). In text2motion, we use pre-trained 300-dimensional word embedding vectors from GloVe [10].

C Evaluation Metrics

We first detail the process of obtaining motion and text feature extractors, and then statistical metrics for evaluating stochastic text-to-motion generation. Since metrics for motion2text translation have been well defined in existing literature [9,15,8,13], we would like to skip the introduction for them.

Motion and Text Feature Extractors learn to produce geometrically closed feature vectors for matched text-motion pairs, and vice versa. Specifically, input text and motion are transformed to two semantic vectors $\mathbf{s}$ and $\mathbf{p}$ respectively using two separate bi-directional GRUs. Then, we enforce feature vectors from matched text-motion pairs to be as close as possible, while mismatched feature vectors to be separated with a margin of at least m. This is approached by optimizing the networks with the following contrastive loss:

$$\mathcal{L}_{cst} = (1-y)(\|\mathbf{s}-\mathbf{p}\|_2^2)^2 + (y)\{\max(0, m - \|\mathbf{s}-\mathbf{p}\|_2^2)\}^2, \quad (1)$$

[1] https://github.com/Maluuba/nlg-eval

where $y \in \{0, 1\}$ that $y = 0$ if **st** and **p** comes from matched text-motion pairs, and vice versa. m is set to 10 for both datasets. Note that test sets are untouched in this process.

The aforementioned text and motion feature extractor are then engaged in the following metrics for evaluating text2motion generation.

- **Frechet Inception Distance** (FID): Features are extracted from real motions in test set and generated motions from corresponding descriptions. Then FID is calculated between the feature distribution of generated motions vs. that of the real motions. FID is an important metric widely used to evaluate the overall quality of generated motions.
- **Diversity**: Diversity measures the variance of the generated motions across all descriptions. From a set of all generated motions from various descriptions, two subsets of the same size S_d are randomly sampled. Their respective sets of motion feature vectors $\{\mathbf{v}_1, ..., \mathbf{v}_{S_d}\}$ and $\{\mathbf{v}'_1, ..., \mathbf{v}'_{S_d}\}$ are extracted. The diversity of this set of motions is defined as
  Diversity $= \frac{1}{S_d} \sum_{i=1}^{S_d} \|\mathbf{v}_i - \mathbf{v}'_i\|$
  $S_d = 300$ is used in experiments.
- **MultiModality**: Different from diversity, multimodality measures how much the generated motions diversify within each text description. Given a set of motions with C descriptions. For c-th description, we randomly sample two subsets with same size S_m , and then extract two subset of feature vectors $\{\mathbf{v}_{c,1}, ..., \mathbf{v}_{c,S_m}\}$ and $\{\mathbf{v}'_{c,1}, ..., \mathbf{v}'_{c,S_m}\}$. The multimodality of this motion set is formalized as
  Multimodality $= \frac{1}{C \times S_m} \sum_{c=1}^{C} \sum_{i=1}^{S_m} \|\mathbf{v}_{c,i} - \mathbf{v}'_{c,i}\|$
  $S_m = 10$ is used in experiments.

D Baseline Implementation

For motion2text translation, unfortunately all baselines have not released their implementations yet. We re-implement SeqGAN [4], RAEs [14] and Seq2Seq(Att) [11] according to the descriptions in their published papers.

For text2motion generation, we re-implement Seq2Seq [7] following its description in paper. In the official implementation of Hier [3], the model is trained to generate motion with fixed length (32 frames).

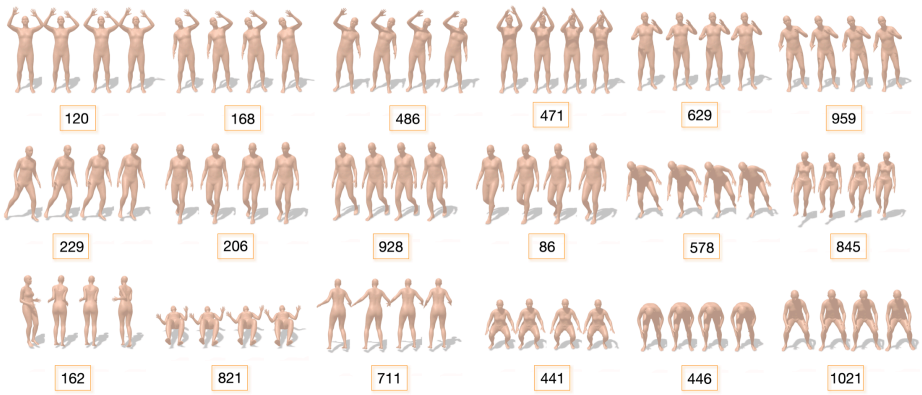

Fig. 2. Exemplar motion tokens and their associated local spatial-temporal contexts, visualized in 4-frame motion segments.

We extend their implementation with curriculum learning to enable the motion generation with variable lengths. Proper modifications are also made to the official implementations of Text2Gesture [2] and Language2Pose [1], to fit in our scenario such as kinematic structure. To adapt MoCoGAN [12] and Dance2Music [6] in our application, we re-use their source code and replace the categorical condition in MoCoGAN and audio signals in Dance2Music with our text features. Due to the specific architecture of their discriminator design, they are only able to generate motions with fixed lengths.

E User Study

Fig. 1 shows the interface of our user survey for evaluating motion2text translation on Amazon Mechanical Turk. For each human motion animation, 6 generated descriptions from different source are randomly reordered. AMT users are asked to rank their preference over these 6 descriptions based on the judgement on accuracy and degree of details. Only users with *master* recognition are considered.

F Motion Token Contexts

To visualize the local context associated with each motion token, we decode individual tokens using the quantization decoder D, that produces short 4-frame motion segments for each token. In Fig. 2, we

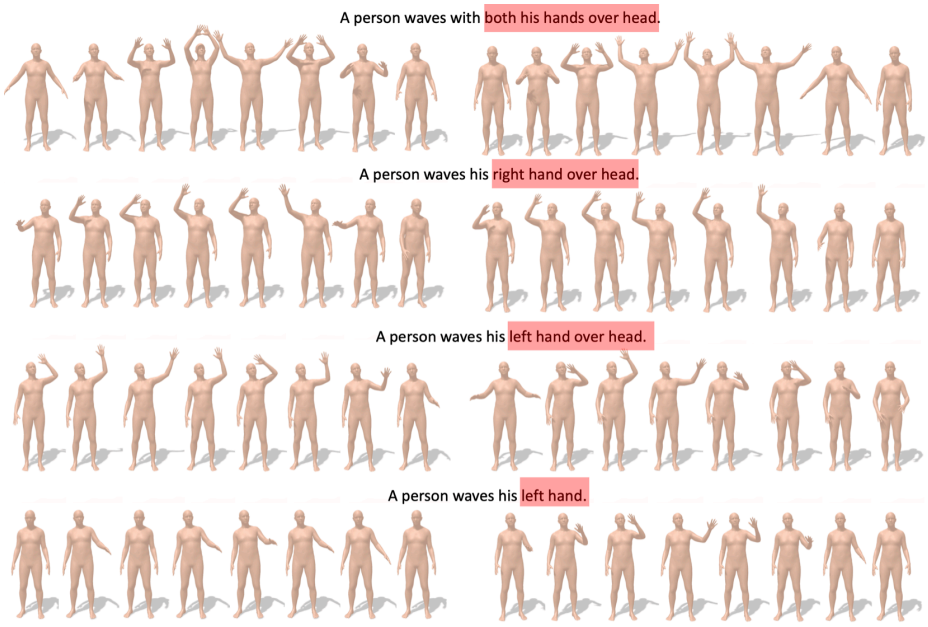

Fig. 3. Examples of text-to-motion mapping by modifying specific parts of text descriptions (highlighted in red box). For each description, we show two resultant motions.

present a gallery of learned motion tokens, as well as the motion segments reflecting their contexts. Note given a tuple of motion tokens, the quantization decoder D learns to naturally mingle their local context with seamless transitions, rather than simply concatenating their motion segments.

G Text Modifications

We also generate 3D motions from language by modifying fixed components of the input text descriptions (Fig. 3). Our text2motion is able to capture the subtle semantic differences (e.g., "both/left/right hand", "over head") in text descriptions.

H Inference Time Analysis

Time consumption of generating 300 motions from different methods on one Nvidia2080Ti: Seq2Seq (14s), Language2Pose (10s), MoCoGAN (1s), Dance2Music (1s), Text2Gesture (250s), Hier(39s), Ours

(9s). Benefiting from reduced time length, our approach is able to provide the same amount of motions with even less time cost than most baselines.

I Network Architecture

Table 1 elaborates the networks we are using for HumanML3D dataset. The dimension of input vectors may vary accordingly while applying to KIT-ML dataset.

References

1. Ahuja, C., Morency, L.P.: Language2pose: Natural language grounded pose forecasting. In: 2019 International Conference on 3D Vision (3DV). pp. 719–728. IEEE (2019) 4
2. Bhattacharya, U., Rewkowski, N., Banerjee, A., Guhan, P., Bera, A., Manocha, D.: Text2gestures: A transformer-based network for generating emotive body gestures for virtual agents. In: IEEE Virtual Reality and 3D User Interfaces (VR). pp. 1–10. IEEE (2021) 4
3. Ghosh, A., Cheema, N., Oguz, C., Theobalt, C., Slusallek, P.: Synthesis of compositional animations from textual descriptions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1396–1406 (2021) 3
4. Goutsu, Y., Inamura, T.: Linguistic descriptions of human motion with generative adversarial seq2seq learning. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 4281–4287. IEEE (2021) 3
5. Holden, D., Komura, T., Saito, J.: Phase-functioned neural networks for character control. ACM Transactions on Graphics (TOG) **36**(4), 1–13 (2017) 1
6. Huang, R., Hu, H., Wu, W., Sawada, K., Zhang, M., Jiang, D.: Dance revolution: Long-term dance generation with music via curriculum learning. arXiv preprint arXiv:2006.06119 (2020) 4
7. Lin, A.S., Wu, L., Corona, R., Tai, K., Huang, Q., Mooney, R.J.: Generating animated videos of human activities from natural language descriptions. Learning **2018**, 1 (2018) 3
8. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: Text summarization branches out. pp. 74–81 (2004) 2
9. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the Association for Computational Linguistics. pp. 311–318 (2002) 2
10. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014) 2
11. Plappert, M., Mandery, C., Asfour, T.: Learning a bidirectional mapping between human whole-body motion and natural language using deep recurrent neural networks. Robotics and Autonomous Systems **109**, 13–26 (2018) 3
12. Tulyakov, S., Liu, M.Y., Yang, X., Kautz, J.: Mocogan: Decomposing motion and content for video generation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1526–1535 (2018) 4

13. Vedantam, R., Lawrence Zitnick, C., Parikh, D.: Cider: Consensus-based image description evaluation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4566–4575 (2015) 2
14. Yamada, T., Matsunaga, H., Ogata, T.: Paired recurrent autoencoders for bidirectional translation between robot actions and linguistic descriptions. IEEE Robotics and Automation Letters **3**(4), 3441–3448 (2018) 3
15. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with bert. arXiv preprint arXiv:1904.09675 (2019) 2
16. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5745–5753 (2019) 1

Components	Architecture
Quantization Encoder (E)	Conv1d(259, 1024, kernel_size=(4,), stride=(2,), padding=(1,)) LeakyReLU(negative_slope=0.2, inplace=True) (ResBlock): Sequential((0): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)) (1): LeakyReLU(negative_slope=0.2, inplace=True) (2): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,))) Conv1d(1024, 1024, kernel_size=(4,), stride=(2,), padding=(1,)) LeakyReLU(negative_slope=0.2, inplace=True) (ResBlock): Sequential((0): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)) (1): LeakyReLU(negative_slope=0.2, inplace=True) (2): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)))
Quantization Decoder (D)	(ResBlock): Sequential((0): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)) (1): LeakyReLU(negative_slope=0.2, inplace=True) (2): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,))) (ResBlock): Sequential((0): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)) (1): LeakyReLU(negative_slope=0.2, inplace=True) (2): Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,))) Upsample(scale_factor=2.0, mode=nearest) Conv1d(1024, 1024, kernel_size=(3,), stride=(1,), padding=(1,)) LeakyReLU(negative_slope=0.2, inplace=True) Upsample(scale_factor=2.0, mode=nearest) Conv1d(1024, 263, kernel_size=(3,), stride=(1,), padding=(1,)) LeakyReLU(negative_slope=0.2, inplace=True) Conv1d(263, 263, kernel_size=(3,), stride=(1,), padding=(1,))
Codebook	Embedding(1024, 1024)
Text (GRU) Encoder	(input_emb):Linear(in_features=300, out_features=512, bias=True) (gru): GRU(512, 512, batch_first=True, bidirectional=True)
Motion (GRU) Decoder	(input_emb): Embedding(1027, 1024) (z2init): Linear(in_features=1024, out_features=1024, bias=True) (gru): ModuleList((0): GRUCell(1024, 1024)) (att_layer): AttLayer((W_q): Linear(in_features=1024, out_features=1024, bias=True) (W_k): Linear(in_features=1024, out_features=1024, bias=False) (W_v): Linear(in_features=1024, out_features=1024, bias=True) (softmax): Softmax(dim=1)) (att_linear): Sequential((0): Linear(in_features=2048, out_features=1024, bias=True) (1): LayerNorm((1024,), eps=1e-05, elementwise_affine=True) (2): LeakyReLU(negative_slope=0.2, inplace=True)) (gru): ModuleList((0): GRUCell(1024, 1024)) (positional_encoder): PositionalEncoding() (mu_net): Linear(in_features=1024, out_features=128, bias=True) (trg_word_prj): Linear(in_features=1024, out_features=1027, bias=False)

Table 1. Architecture of our networks on dataset HumanML3D.