

SegMo: Segment-aligned Text to 3D Human Motion Generation

Supplementary Material

Bowen Dang¹ Lin Wu² Xiaohang Yang³ Zheng Yuan¹ Zhixiang Chen^{1*}
¹University of Sheffield ²University of Glasgow ³Queen Mary University of London

¹{bdang2, zheng.yuan1, zhixiang.chen}@sheffield.ac.uk ²l.wu.1@research.gla.ac.uk ³xiaohang.yang@qmul.ac.uk

A. Text Segment Extraction Module

A.1. Ablation Results using Different LLMs

We evaluated the impact of replacing the LLM used in the text segment extraction module by substituting Qwen 3:8B [9] with Llama 3:8B [1] and Qwen 2.5:7B [7], as reported in Table 1. As shown, Qwen 3:8B produced higher-quality text segments, resulting in superior overall performance.

A.2. Prompt for Text Segment Extraction

The prompt used for generating text segments is shown in Figure 1.

B. Motion Segmentation Module

While equal segmentation provides a simple yet effective baseline, it does not explicitly capture the semantic structure of motion. Therefore, we further explored other segmentation methods, including the Change-Point-Detection (CPD)-based method that focuses on detecting motion transitions, and the clustering-based method that aims to detect motion primitives. Additionally, we process the BABEL [6] annotations to create a validation set for evaluating these segmentation methods.

Change-Point-Detection (CPD)-based Segmentation:

The CPD-based method detects change points directly from a motion sequence and treats them as segmentation boundaries. We first apply a sliding window to extract motion segments from the complete motion sequence, and then employ a kernel-based change point detection method to identify change points within these segments. Specifically, we use a Gaussian kernel for detection, implemented using the Python `ruptures` [8] package.

Clustering-based Segmentation: The main idea of clustering-based segmentation is to first extract motion primitives from large-scale motion data and then utilize

them to segment motion sequences. This method consists of two steps:

Motion Primitive Construction: A motion primitive is defined as a short motion segment representing a specific motion pattern. Starting with a large motion dataset, such as AMASS [4], we apply a sliding window to extract a large number of motion segments. These segments are then grouped using a clustering algorithm such as K-Means [3], with similar motion segments clustered together to form a motion primitive library.

Motion Segmentation: After constructing the motion primitive library, we use it to segment the motion sequences. Given a motion sequence, the goal is to segment it into distinct motion patterns. First, we apply a sliding window to extract all motion segments. For each segment, we compute its distance to the center of each motion primitive cluster, yielding a distance matrix $D \in \mathbb{R}^{N \times K}$, where N is the number of segments and K is the number of clusters. This distance matrix is treated as a cost matrix, and the optimal segmentation is determined by finding the cutting points that minimize the total distance, which can be efficiently solved using a dynamic programming algorithm.

Evaluation: BABEL [6] provides frame-level action labels for the AMASS dataset. However, a single frame may be associated with multiple labels, which is not suitable for our setting. To address this, we first process the annotations to ensure that each frame contains only one label, and then extract the segmentation points between consecutive segments. Based on this, we construct a validation set to evaluate the performance of different segmentation methods. We measure the error of the segmentation points in the token sequence, and the comparison is summarized in Table 2. The results show that the CPD-based and clustering-based methods yield a lower mean error, while uniform segmentation achieves a smaller variance of the errors.

*Corresponding author

Method	R-Precision \uparrow			FID \downarrow	MM-Dist \downarrow	Diversity \rightarrow
	Top 1	Top 2	Top 3			
Real	0.511 \pm .003	0.703 \pm .003	0.797 \pm .002	0.002 \pm .000	2.974 \pm .008	9.503 \pm .065
SegMo (Llama 3 8B)	0.550 \pm 0.003	0.744 \pm 0.003	0.835 \pm 0.002	0.047 \pm 0.002	2.795 \pm 0.007	9.574 \pm 0.068
SegMo (Qwen 2.5 7B)	0.542 \pm 0.003	0.738 \pm 0.003	0.831 \pm 0.003	0.063 \pm 0.002	2.830 \pm 0.007	9.587 \pm 0.089
SegMo (Mean Pooling Agg)	0.548 \pm 0.003	0.743 \pm 0.002	0.833 \pm 0.002	0.065 \pm 0.002	2.812 \pm 0.008	9.583 \pm 0.052
SegMo (Max Pooling Agg)	0.541 \pm 0.002	0.738 \pm 0.002	0.830 \pm 0.002	0.057 \pm 0.002	2.834 \pm 0.006	9.611 \pm 0.062
SegMo (Query-to-Token CA Agg)	0.549 \pm 0.002	0.744 \pm 0.002	0.837 \pm 0.002	0.052 \pm 0.002	2.783 \pm 0.006	9.559 \pm 0.120
SegMo (CLS-token SA Agg)	0.545 \pm 0.002	0.742 \pm 0.002	0.834 \pm 0.002	0.051 \pm 0.002	2.813 \pm 0.007	9.591 \pm 0.102
SegMo	0.553 \pm 0.003	0.748 \pm 0.003	0.838 \pm 0.002	0.042 \pm 0.002	2.782 \pm 0.008	9.632 \pm 0.079

Table 1. Ablation results of replacing the LLMs and the aggregation module on the HumanML3D test set. For each metric, we repeat the evaluation 20 times and report the average with a 95% confidence interval. **Red** and **Blue** indicate the best and the second-best results.

	Error (mean)	Error (std)
uniform Seg	3.61	3.01
CPD-based Seg	2.95	3.46
clustering-based Seg	2.97	4.04

Table 2. Evaluation of different motion segmentation methods on the BABEL dataset.

C. Motion Aggregation Module

The aggregation module aggregates motion tokens within a segment into a motion segment token, and we experimented with several strategies: (1) Mean Pooling, (2) Max Pooling, (3) Query-to-Token Cross-Attention, and (4) CLS-token Self-Attention. The comparison of different motion aggregation modules is presented in Figure 2. As illustrated in Table 1, Mean-Max aggregation achieves the best overall performance.

D. Text-Motion Alignment Module

Here we present more details of the alignment modules we have explored in the discussion section, including the Batch-Level Alignment, Global Alignment, and Token Alignment.

Batch-Level Alignment: We introduce a batch-level alignment module, where the contrastive loss is computed across all segments within a batch, which extends the negative samples beyond those within a sample. The alignment loss is defined as follows:

$$\begin{aligned}
\mathcal{L}_{t2m} &= -\frac{1}{B \cdot A} \sum_{i=1}^B \sum_{j=1}^A \log \frac{\exp(\text{sim}(\mathbf{t}_j^i, \mathbf{m}_j^i)/\tau)}{\sum_{k=1}^B \sum_{l=1}^A \exp(\text{sim}(\mathbf{t}_j^i, \mathbf{m}_l^k)/\tau)}, \\
\mathcal{L}_{m2t} &= -\frac{1}{B \cdot A} \sum_{i=1}^B \sum_{j=1}^A \log \frac{\exp(\text{sim}(\mathbf{t}_j^i, \mathbf{m}_j^i)/\tau)}{\sum_{k=1}^B \sum_{l=1}^A \exp(\text{sim}(\mathbf{t}_l^k, \mathbf{m}_j^i)/\tau)}, \\
\mathcal{L}_{align} &= \frac{1}{2}(\mathcal{L}_{t2m} + \mathcal{L}_{m2t}).
\end{aligned} \tag{1}$$

Global Alignment: Global alignment aligns the entire textual description with the complete motion sequence. The alignment loss is defined as follows:

$$\begin{aligned}
\mathcal{L}_{t2m} &= -\frac{1}{B} \sum_{i=1}^B \log \frac{\exp(\text{sim}(T^i, M^i)/\tau)}{\sum_{j=1}^B \exp(\text{sim}(T^i, M^j)/\tau)}, \\
\mathcal{L}_{m2t} &= -\frac{1}{B} \sum_{i=1}^B \log \frac{\exp(\text{sim}(T^i, M^i)/\tau)}{\sum_{j=1}^B \exp(\text{sim}(T^j, M^i)/\tau)}, \\
\mathcal{L}_{align} &= \frac{1}{2}(\mathcal{L}_{t2m} + \mathcal{L}_{m2t}).
\end{aligned} \tag{2}$$

$\text{sim}(T^i, M^j)$ denotes the cosine similarity between the i -th text and j -th motion sequence.

Token Alignment: Token alignment aligns each motion token with the text segment it belongs to. The alignment loss is defined as follows:

$$\mathcal{L}_{align} = -\frac{1}{B * L} \sum_{i=1}^B \sum_{j=1}^L \log \frac{\exp(\text{sim}(\mathbf{t}_{seg(\mathbf{x}_j^i)}^i, \mathbf{x}_j^i)/\tau)}{\sum_{k=1}^A \exp(\text{sim}(\mathbf{t}_k^i, \mathbf{x}_j^i)/\tau)}. \tag{3}$$

L denotes the maximum motion length, and $\mathbf{t}_{seg(\mathbf{x}_j^i)}^i$ denotes the corresponding text segment to which the motion token \mathbf{x}_j^i belongs.

Comparison Analysis: Although batch-level alignment introduces more negative samples, its performance does not surpass that of our proposed method. A possible reason is that similar or identical motion segments often exist within a batch—especially for common actions such as walking or running—making it difficult for the model to discriminate among them. Global alignment, on the other hand, ignores fine-grained segment correspondences and therefore fails to capture the temporal structure between text and motion. Token alignment fails to align a static motion token with a text segment representing a dynamic motion.

You are a helpful assistant. Your task is to extract and temporally order human actions from a sentence describing a person performing one or more actions.

Guidelines:

- If the sentence describes **only one action**, return it as a full sentence without using the “#” symbol.
- If the sentence describes **multiple actions**, return each as a full sentence, in the correct temporal order, separated by the “#” symbol.
- Preserve descriptive modifiers such as “quickly”, “slowly”, “two times”, “as if”, “like”, etc.
- Normalize expressions such as “appears to” or “seems to” into direct action statements.
- Do **not** include any extra text (e.g., “Output:”, quotes, or explanations). Return only the result string in the specified format.

Examples:

(1) Input: a person runs quickly after walking in a circle.

Output: a person walks in a circle#a person runs quickly.

Comment: “walks” comes before “runs” due to the temporal cue “after”.

(2) Input: a person jumps two times, then walks while waving the hands.

Output: a person jumps two times#a person walks while waving the hands.

Comment: “two times” is a modifier; “walks” and “waves the hands” occur simultaneously.

(3) Input: a person takes a box off the table and puts it on the floor.

Output: a person takes a box off the table#a person puts a box on the floor.

Comment: The sentence contains two sequential actions—taking the box and then putting it down.

(4) Input: a person is standing and waving the hands.

Output: a person is standing and waving the hands.

Comment: The sentence contains two simultaneous actions—standing and waving the hands.

(5) Input: a person is bowing left and right.

Output: a person is bowing left and right.

Comment: “left and right” is a descriptive modifier and should be preserved.

(6) Input: a person is jumping around like he is in an accident.

Output: a person is jumping around like he is in an accident.

Comment: “like he is in an accident” is a descriptive modifier and should be preserved.

(7) Input: a person appears to wave the hands.

Output: a person waves the hands.

Comment: “appears to” is not an action and is removed.

Now process the following input:

Figure 1. Prompt for generating text segments using LLM.

E. Segment Number Distribution Analysis

Figure 3 shows the segment number distributions of the HumanML3D [2] and KIT-ML [5] datasets across training, validation, and test sets. More than half of the samples in HumanML3D contain multiple segments, whereas

most samples in KIT-ML consist of only a single segment. This discrepancy highlights that richer compositional structure makes segment-level alignment more beneficial. Consequently, our alignment module has a more pronounced effect on HumanML3D compared to KIT-ML.

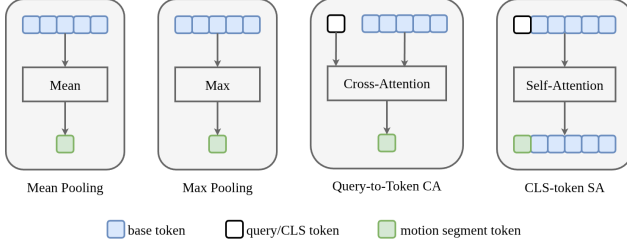


Figure 2. The comparison of different motion aggregation modules.

F. Failure Cases

Figure 4 shows some failure cases of our method. In Figure 4 (a), although our method generates a marching motion while raising the arm, it neglects the keyword “*in place*”, leading to an incorrect result. This suggests that our model sometimes struggles to capture subtle textual modifiers. In Figure 4 (b), when the specified motion length is short but the textual description contains multiple consecutive actions, our method may fail to generate the complete sequence, such as omitting the final “*stop*” action. This suggests that stronger constraints are needed to encourage the model to allocate appropriate time durations for each action.

References

- [1] Meta AI. Introducing meta llama 3: The most capable openly available llm to date. <https://ai.meta.com/blog/meta-llama-3/>, 2024. 1
- [2] 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. 3
- [3] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982. 1
- [4] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5442–5451, 2019. 1
- [5] Matthias Plappert, Christian Mandery, and Tamim Asfour. The kit motion-language dataset. *Big data*, 4(4):236–252, 2016. 3
- [6] Abhinanda R Punnakal, 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. 1
- [7] Qwen Team. Qwen2.5: A party of foundation models. <https://qwenlm.github.io/blog/qwen2.5/>, 2024. 1
- [8] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selec-

tive review of offline change point detection methods. *Signal Processing*, 167:107299, 2020. 1

- [9] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 1

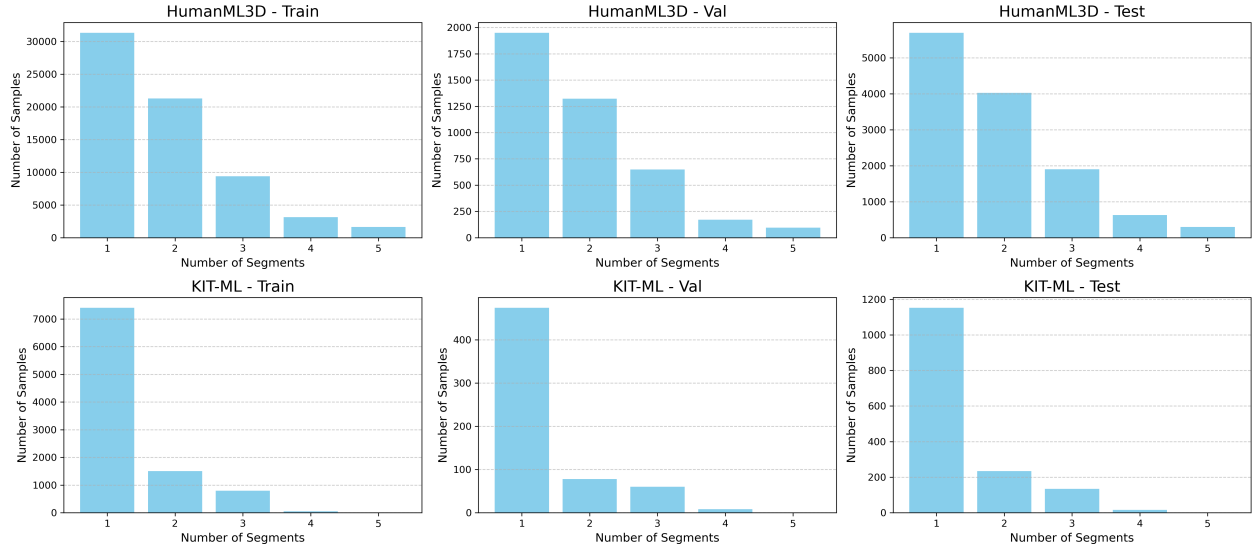


Figure 3. Segment number distributions of HumanML3D and KIT-ML datasets across training, validation, and test set. The x-axis denotes the number of segments, and the y-axis indicates the corresponding sample counts.

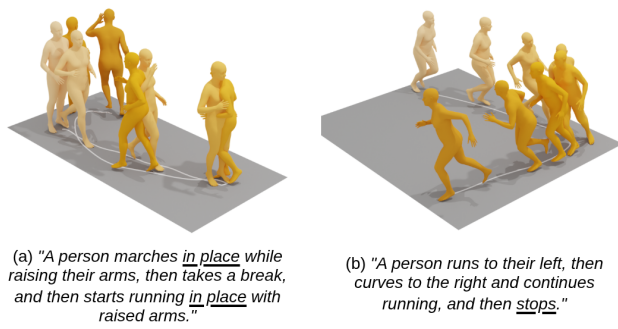


Figure 4. Failure cases of our method.