Supplementary Material for LaMoGen: Language to Motion Generation Through LLM-Guided Symbolic Inference

Junkun Jiang, Ho Yin Au, Jingyu Xiang, Jie Chen *
Department of Computer Science, Hong Kong Baptist University, HKSAR
{csjkjiang, cshyau, csjyxiang, chenjie}@comp.hkbu.edu.hk

This supplementary material provides additional details that could not be included in the main paper due to page limitations. The contents are organized as follows:

- **Details of LabanLite** (Sec. 1): Includes a comprehensive definition of LabanLite, semantic interpretations of Laban symbols, as well as the relationships among Laban symbols and Laban instances. Look-up tables for constructing the Conceptual Description database for LLM-based composition are also provided, accompanied by visual examples for clearer understanding.
- **Automatic Laban Code Detection Workflow** (Sec. 2): Describes the formulated detection process and the predefined thresholds used for discretization.
- **Laban Benchmark** (Sec. 3): Provides the mathematical definition of the Laban metric, details on the construction of the HumanML3D-Laban dataset, and illustrative examples.
- **Experiment Details** (Sec. 4): Includes training setups, implementation specifics of the proposed models, and prompts used for LLM-based composition.
- **Experimental Results** (Sec. 5): Extends beyond the main paper by presenting ablation studies, user studies, and qualitative comparisons. Supplementary LLM-composed examples demonstrate that LLMs can interpret ambiguous text and compose Laban symbols.
- **Limitations and Future Directions** (Sec. 7): Discusses the current limitations of the proposed framework and outlines possible future research directions.

The complete code and related data will be released following approval of the publication status.

1. Details of LabanLite

We begin by introducing Labanotation to provide readers with a quick understanding of Laban theory (Sec. 1.1). We then describe LabanLite (Sec. 1.2), which constitutes our primary contribution.

*Corresponding author.
Project page: https://jjkislele.github.io/LaMoGen/

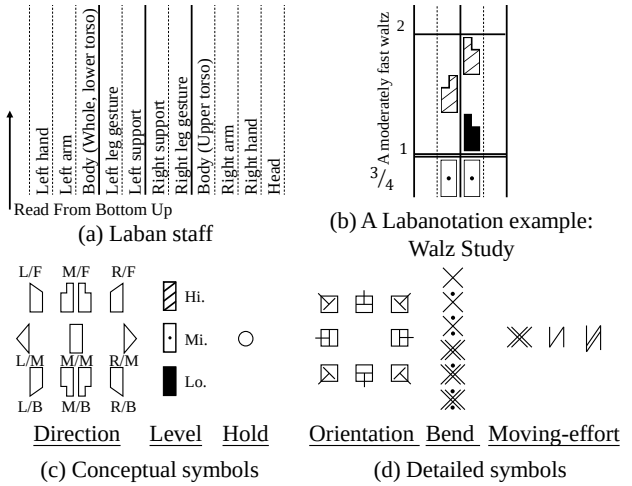
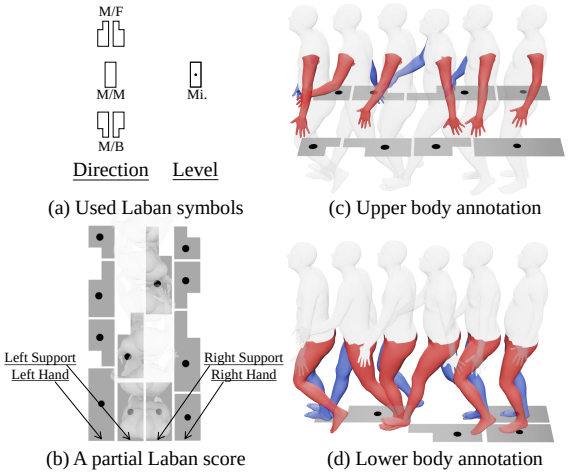


Figure 1. **Preliminary of Labanotation**. (a) Illustration of a Laban staff. (b) A segment of a Laban score [16]. **We further extend Labanotation into LabanLite**. LabanLite categorizes Laban symbols into (c) Conceptual symbols, where "L", "R", "F", "M", and "B" denote left, right, forward, middle, and backward directions. "Hi.", "Mi.", and "Lo." denote high, middle, and low levels. (d) Detailed symbols, where "Orientation", "Bend" and "Moving-effort" describe the kinematic details.

1.1. Preliminary of Labanotation

Labanotation employs distinctive symbols arranged on a vertical staff to form a Laban score, documenting the spatial positions and temporal sequences of various body parts [16]. As illustrated in Fig. 1(a), a standard Laban staff consists of 11 columns, each corresponding to a specific body part. Symbols are placed in each column from bottom to top, where the position and sequence reflect the type of movement as well as its start and end times. The length of each symbol indicates the movement duration, rendering Labanotation an event-wise annotation scheme. Choreographers often omit symbols for movements deemed insignificant. Fig. 1(b) shows a sample score for a waltz study. Choreographers synchronize the staff with music us-



(a) Used Laban symbols (c) Upper body annotation

(b) A partial Laban score (d) Lower body annotation

Figure 2. Illustration of a partial Laban score. This figure provides a visual explanation of the annotation process for a forward walk movement.

ing bar lines and time signatures. Notably, important movements—such as the right foot's step—are clearly marked with a symbol, while less relevant movements (e.g., the left foot's status) are often omitted. As a result, some parts of the score contain blank spaces where symbols are absent.

1.1.1. Quick Tutorial of Labanotation

To facilitate understanding, we provide a brief tutorial on Labanotation that serves as a prerequisite for later sections. Here, we explain how to read a Laban score and introduce the annotation protocol, enabling readers to quickly grasp its essential principles.

Labanotation applies specific constraints and simplifications to the symbol sequence for improved human readability, resulting in a set of visualization rules:

1. Columns that are not of primary interest may be left blank, allowing dancers/performers interpretive freedom for unspecified movements. For example, if a choreographer wishes to focus on a jump, the lower body columns (Left support and Right support) contain relevant symbols, while arm-related columns may be left empty, giving the dancer creative liberty for the upper body.
2. For the Left Support and Right Support columns, a symbol present in one column at a particular time typically means the corresponding position in the other column is left empty.
3. Direction and Level symbols are combined for simplicity.
4. A new Laban symbol appears in a column only when it differs from the preceding symbol; otherwise, it is omitted for clarity and conciseness.

For further clarity, Fig. 2 presents a partial Laban score representing a forward walk. In this example, the Laban staff consists of four columns, from left to right: Left hand, Left support, Right support, and Right hand. This score abstracts a complex motion sequence into a concise sequence using only four types of symbols: Level "Mi."; and Direction "M/F", "M/M", and "M/B". Throughout the walking sequence, the initial state—both feet together and hands naturally lowered—is represented as Direction "M/M" with Level "Mi." in all four columns. The left foot then steps forward accompanied by a backward swing of the right arm, which is recorded as a forward-medium symbol set (Direction "M/F" and Level "Mi.") in the support column, and a backward symbol set (Direction "M/B" and Level "Mi.") in the hand column. In the final state, the left foot continues forward (Direction M/F and Level "Mi."), while the right foot remains behind.

1.2. LabanLite: A Motion Representation Enhanced over Labanotation

As discussed in the main paper, to enhance suitability for digital encoding and facilitate composition by large language models (LLMs) in subsequent processes, we introduce LabanLite. LabanLite preserves the expressive power of Labanotation while providing three key enhancements:

1. **Symbol classification**: Symbols are classified as either *conceptual* (representing the primary movement structure) or *detailed* (specifying fine-grained attributes);
2. **Frame-wise annotation**: The event-wise annotation of Labanotation is replaced by frame-wise annotation. Each symbol is converted into a *Laban instance* at every frame, and staff columns are redefined as *body-part groups* with dedicated *attribute fields*;
3. **Structured description**: Each conceptual symbol is accompanied by a strictly formatted textual description (i.e., `<body-part group> <moving semantic> in <time> seconds`) termed ***Conceptual Description***. This standardization enables unambiguous conversion from symbolic event-wise annotations to frame-wise instances via direct textual mapping.

In the following, we first introduce the definitions of conceptual and detailed symbols (Sec. 1.2.1). Next, we describe frame-wise annotation in LabanLite, including Laban instances and their attribute fields within body-part groups (Sec. 1.2.2). We then explain the conversion process from event-wise symbol sequences to frame-wise instance sequences (Sec. 1.2.3). Finally, we introduce the Conceptual Description and detail the construction of the Conceptual Description database (Sec. 1.2.4).

1.2.1. Conceptual and Detailed Laban Symbols

In LabanLite, Laban symbols are categorized into two types:

- **Conceptual symbols** (*Direction, Level, Hold*): Describe general movement concepts and the structural aspects of

motion;

- **Detailed symbols** (*Orientation, Bend, Moving-effort*): Capture subtle details of individual body part movements.

Tables 9, 10, 11, 12, 13, and 14 show the names, graphical appearances, and *partial semantic meanings* of Laban symbols: Direction, Level, Hold, Orientation, Bend, and Moving-effort, respectively. We refer to the semantic meanings as "partial" because the full meaning of a symbol depends on additional factors such as the body-part group, its duration (length), and its specific position on the Laban staff. Additionally, for the "M/F" and "M/B" symbols in the Direction category, each typically has two graphical forms. The specific form is determined by the staff column (i.e., body part): for example, if the "M/F" symbol is placed in the "Left hand" column, the left-facing form is used; in the "Right hand" column, the right-facing form is used.

1.2.2. Body-Part Group and Movement Attributes

LabanLite extends the definition of staff columns from Labanotation by introducing body-part groups. Each column now represents a body-part group associated with a set of movement attributes, corresponding to specific sets of Laban symbols.

Importantly, a single body-part group may be represented by multiple staff columns since a group can include multiple body parts, each with distinct attributes and corresponding Laban symbols. This design reflects the constraints of human kinematics—not every body part possesses the same attribute fields. For example, the Level symbol is not applicable to the head column, as the head cannot independently move to arbitrary heights; thus, the Head does not have the Level attribute.

This design enables simultaneous recording of both conceptual and detailed movement information. Detailed definitions of body-part groups and their attributes are provided in Table 15.

For greater clarity, consider the movement "wave left hand." The relevant body-part group is "Upper-L," which includes the "Left arm" and "Left hand" staff columns. The Direction, Level, and Hold symbols are assigned to the "Left hand" column to indicate the high-level movement concept (e.g., raising up the left arm), while the Bend symbol is placed on the "Left arm" column to capture lower-level movement details (e.g., flexing and stretching the left arm). Both symbols are assigned to the body-part group "Upper-L".

1.2.3. Symbol Instantiation: From Event-wise to Frame-wise

Traditional Labanotation is fundamentally an event-wise annotation and motion analysis system: a placed Laban symbol indicates the corresponding body part's movement, along with its onset and duration. However, modeling temporal coordination across multiple body parts becomes

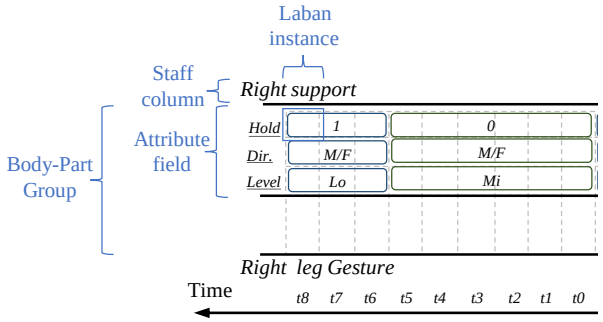
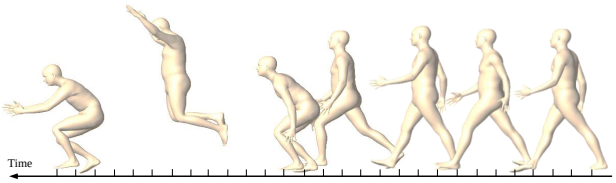
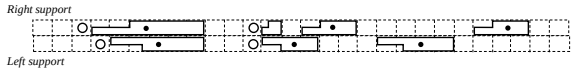


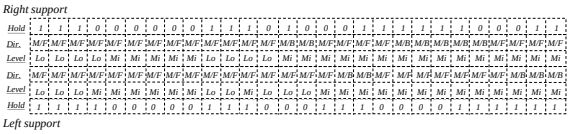
Figure 3. Illustration of the relationships among Body-Part Groups, attribute fields, Laban staff columns, and Laban instances.



(a) Key frames of a *"Walk and then jump"* motion



(b) A human-interpretable Laban symbol sequence



(c) A computational-ready Laban instance sequence

Figure 4. Visual comparison between a human-interpretable symbol sequence and a computational-ready instance sequence. The illustrated Laban staff columns correspond to Left support and Right support.

challenging with event-wise symbol sequences. This is because symbols with differing onset times and durations create asynchronous event streams that are difficult to align and analyze together, limiting their suitability for computation-driven tasks such as automatic decoding or generation.

To make the symbol sequence computationally tractable, we define a *Laban instance* as a unique pairing of a symbol and a Body-Part Group at a specific frame. This enables a frame-wise annotation, where explicit values are assigned to every attribute for each body-part group in every frame. The relationships among Laban symbols, staff columns, Laban instances, and Body-Part Groups are illustrated in Fig. 3.

Specifically, as described in the main paper, the instantiation process is formulated as follows. Given an input motion sequence of T frames, $X = \{x_t\}_{t=1}^{T}$, we propose an Automatic Symbol Detection Workflow $\mathcal{F}$, which converts

X into a sequence of instances $S = \mathcal{F}(X)$, where:

$$S = \{s_t^{i,j} \mid t \in [1, T],\ i \in [1, A_j]\ \ j \in [1, G]\}. \quad (1)$$

Here, G denotes the total number of Body-Part Groups, and A_j represents the attribute field dimension for Group j. Note that different Body-Part Groups may be associated with distinct sets of motion attribute fields, due to variations in their movement ranges and functional roles.

Discussion: Human-Interpretable vs. Computational-Ready. As previously discussed, Labanotation applies specific rules and simplifications to improve human readability, often leaving columns or time intervals blank to keep the score clear and concise. In contrast, computational applications such as decoding motion from Laban codes (described in Sec. 3.1.2 of the main paper) or generating new codes require instantiating every symbol at every frame—ensuring each attribute field for every Body-Part Group has ***an explicit, unambiguous value, without omissions***.

To address these needs, we introduce the concept of *Laban instances*, which makes the Laban symbol sequence fully dense and computationally ready. Fig. 4 compares human-interpretable event-wise sequences and computational-ready frame-wise Laban instance sequences for lower body movement, highlighting the differences between a traditional Laban score and our proposed approach.

1.2.4. Conceptual Description and Conceptual Description Database

As defined in the main paper (Sec. 3.1.1), a conceptual symbol can be mapped to a Conceptual Description with the format "`<body-part group> <moving semantic> in <time> seconds`". We maintain a Conceptual Description database to enable LLMs to compose motion plans at the conceptual level via retrieval-augmented prompting (Sec. 3.3.1).

Table 16 and Table 17 provide definitions for Conceptual Descriptions. Taking the support body-part group as an example, "Support-L" represents the left support and "Support-R" represents the right support; here, we focus on the right support. The moving semantics for this group are defined in Table 16, such as "steps to right." Through the Automatic Laban Code Detection Workflow, we can obtain the duration of this movement (e.g., 2 seconds). Thus, the corresponding Conceptual Description is "`Support Right steps to right in 2 seconds`".

This description can be unambiguously mapped back to its conceptual symbol, providing LLMs with an explicit interface to interpret and synthesize motion. By modifying the conceptual description, LLMs can alter the corresponding Laban symbols and thereby revise the motion plan. Owing to the standardized format of conceptual descriptions, these mappings are precise and facilitate effective symbolic motion synthesis from textual input. In this supplementary material, Sec. 5.3.1 presents examples of LLM-edited conceptual results, illustrating two key capabilities: (1) LLMs can edit existing conceptual symbols while preserving the temporal structure, and (2) even when provided with ambiguous descriptions, LLMs are able to compose symbols that closely align with the user's intended meaning.

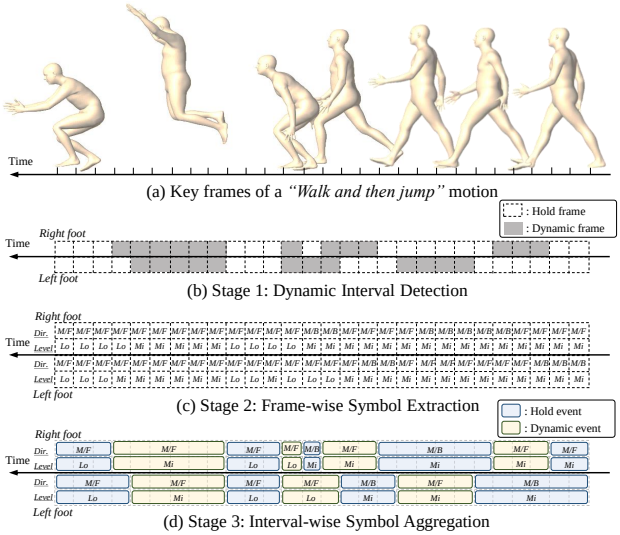


Figure 5. Illustration of extracting lower-body conceptual symbols: (a) Given a motion sequence; (b) we segment the sequence into dynamic and hold intervals by computing foot velocity; (c) in parallel, frame-wise Laban symbols are identified; (d) for each interval, aggregated frame-wise symbols yield the most representative symbol for the interval.

2. Automatic Laban Code Detection Workflow

As described in the main paper, the proposed workflow consists of three steps: (1) Dynamic Interval Segmentation, which identifies dynamic and hold intervals; (2) Frame-wise Symbol Extraction, which converts the pose of each frame into corresponding Laban symbols; and (3) Interval-wise Symbol Aggregation, which selects the most representative Laban symbol for each interval. As illustrated in Fig. 5, we demonstrate an example of the lower body symbol detection process. Here, we provide a detailed formulation of the Frame-wise Symbol Extraction step, including the associated predefined thresholds.

Specifically, all motion sequences are parameterised using the SMPL model [13] without facial and finger key joints. Key joints such as the left/right hand, elbow, shoulder, hip, knee, foot, pelvis, and spine2 are extracted to compute the corresponding Laban symbol sequences.

2.1. Details of Frame-wise Symbol Extraction

Direction and Level symbols. To ensure consistency, all body motions are transformed into a canonical space. This is achieved by fixing the root (pelvis) of each pose at the origin, aligning the triangular plane formed by the pelvis, right leg, and left leg key joints with the xz-plane, and orienting the body to face the negative y-axis. For each key joint (left/right hand, elbow, foot, knee), the L_2-norm distance to the pelvis is projected onto the x, y, and z planes, denoted as a, b, and c, respectively. These projected distances are compared against predefined thresholds to assign Direction and Level symbols for different body part groups (Support-L, Support-R, Upper-R, Upper-L).

For the lower body, the thresholds are as follows:

- Direction (x-axis): $a < -0.1$ is assigned "R"; $a > 0.3$ is "L"; otherwise, "M".
- Direction (y-axis): $b < -0.15$ is "F"; $b > -0.05$ is "B"; otherwise, "M".
- Level (z-axis): $0 > c > -0.8$ is "Lo."; $c > 0$ is "Hi."; otherwise, "Mi.".

For the upper body:

- Direction (x-axis): $a < -0.1$ is "R"; $a > 0.3$ is "L"; otherwise, "M".
- Direction (y-axis): $b < -0.2$ is "F"; $b > 0.1$ is "B"; otherwise, "M".
- Level (z-axis): $c < -0.2$ is "Lo."; $c > 0.1$ is "Hi."; otherwise, "Mi.".

Figure 6 illustrates the distance calculation for the left hand as an example.

Hold symbol. The velocity magnitude of each hand and foot is analysed. Local maxima in the x, y, and z velocity components are identified to determine the turning points of the wrist's three-dimensional trajectory. Frames in which the velocity falls below a predefined threshold are labelled as "hold". The threshold is set to 0.015 for the feet and 0.0005 for the hands.

Bend symbol. Euler angles between adjacent body segments are computed and discretised into six intervals, each spanning $30°$.

Orientation symbol. The facing orientation is determined by calculating the angle between the line connecting the hip key joints and the negative y-axis. The resulting angle is quantised into eight discrete directions, each spanning $45°$.

Moving-effort symbol. The global absolute velocity of the pelvis is computed and discretised using predefined intervals. The velocity components on the xy- and yz-planes are assigned to one of five speed categories: 0 (very slow), 1 (slow), 2 (normal), 3 (fast), and 4 (very fast), according to the following rules: For both horizontal and vertical velocity components:

- $0.1 < v \leq 0.5$: label 1;
- $0.5 < v \leq 1.0$: label 2;
- $1.0 < v \leq 2.0$: label 3;
- $v > 2.0$: label 4;
- $v \leq 0.1$: label 0.

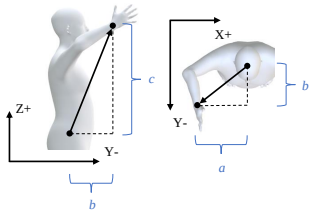


Figure 6. Illustration of extracting Direction and Level symbols for the right hand, in a canonical space.

3. Details of Laban Benchmark

3.1. Laban Metrics

Given the ground-truth Laban instance sequence S and the generated Laban instance sequence $\hat{S}$, the proposed three metrics, including Semantic Alignment (SMT), Temporal Alignment (TMP) and Harmonious Alignment (HMN), calculate the similarity between S and $\hat{S}$ using the Longest Common Subsequence (LCS) length.

Specifically, for SMT and TMP, we measure the Left/Right hand and foot body parts, while for HMN, we measure the body part pairs of [Left hand, Right hand], [Left foot, Right foot], [Left hand, Left foot], [Left hand, Right foot], [Right hand, Left foot], and [Right hand, Right foot]. In the main paper, we report the average HMN scores of [Left hand, Left foot], [Left hand, Right foot], [Right hand, Left foot], and [Right hand, Right foot] pairs due to the table scale limit.

Based on Eq. 1, a Laban instance sequence is defined by $S = \{s_t^{i,j}\}$ where t, i, j denote the frame index, attribute index, and Body-Part Group index, respectively. We combine the same attribute from each Body-Part Group across frames to form a duration-ignored Laban instance sequence $\tilde{S}$:

$$\tilde{S} = \{\tilde{s}_n^{i,j} \mid n \in [1, N_{i,j}],\ i \in [1, A_j]\ j \in [1, G]\}, \quad (2)$$

where n denote the duration-ignored Laban instance index and $N_{i,j}$ represents the total instance number of i-th attribute and j-th Body-Part Group.

To calculate the selected body part's Laban metric, we fix the attribute index and Body-Part Group index, considering the subset $S_{i^\star,j^\star} = \{s_t^{i^\star,j^\star} \mid t \in [1, T]\}$ and the duration-ignored subset $\tilde{S}_{i^\star,j^\star} = \{\tilde{s}_n^{i^\star,j^\star} \mid n \in [1, N_{i,j}]\}$.

Semantic Alignment (SMT) evaluates the similarity of inter-body part Laban instances while disregarding their durations. Given an attribute index-fixed, Body-Part Group-fixed duration-ignored subset $\tilde{S}_{i^\star,j^\star}$ and its generation $\hat{\tilde{S}}_{i^\star,j^\star}$, we formulate the LCS, computing under dynamic programming as follows:

$$f(u,v) = \begin{cases} 0 & \text{if } u = 0 \text{ or } v = 0, \\ f(u-1, v-1) + 1 & \text{if } \tilde{s}_u^{i^\star,j^\star} = \hat{\tilde{s}}_v^{i^\star,j^\star}, \\ \max(f(u-1,v), f(u,v-1)) & \text{otherwise,} \end{cases} \tag{3}$$

Where u and v denote the index of the duration-ignored Laban instance sequence, i.e., $u, v \in N_{i^\star,j^\star}$. Such that, the SMT between $\tilde{S}_{i^\star,j^\star}$ and $\hat{\tilde{S}}_{i^\star,j^\star}$ is calculated by the normalised length of the LCS:

$$\text{Sim}_{\text{SMT}}(\tilde{S}_{i^\star,j^\star}, \hat{\tilde{S}}_{i^\star,j^\star}) = \frac{f(N_{i^\star,j^\star}, \hat{N}_{i^\star,j^\star})}{\max(N_{i^\star,j^\star}, \hat{N}_{i^\star,j^\star})}. \tag{4}$$

Temporal Alignment (TMP) evaluates the similarity of each inter-body-part Laban instance while considering each symbol's duration, to ensure that not only the types but also the temporal extents of the motions are consistent between the ground truth and the generated sequence. To account for the duration of each symbol, the inputs are changed to Laban instance sequences $S_{i^\star,j^\star}$ and $\hat{S}_{i^\star,j^\star}$, and we modify Eq. 3 as:

$$g(u,v) = \begin{cases} 0 & \text{if } u = 0 \text{ or } v = 0, \\ g(u-1, v-1) + 1 & \text{if } s_u^{i^\star,j^\star} = \hat{s}_v^{i^\star,j^\star}, \\ \max(g(u-1,v), g(u,v-1)) & \text{otherwise,} \end{cases} \tag{5}$$

where u and v denote the frame index, i.e., $u, v \in T$. The Temporal Alignment score is then:

$$\text{Sim}_{\text{TMP}}(S_{i^\star,j^\star}, \hat{S}_{i^\star,j^\star}) = \frac{g(T,\hat{T})}{\max(T,\hat{T})}. \tag{6}$$

Harmonious Alignment (HMN) evaluates the synchronous occurrence of Laban symbols across pairs of body parts. Given a body part pair specified by $[(i_1, j_1), (i_2, j_2)]$, and their corresponding duration-ignored Laban instance sequences $\tilde{S}_{i_1,j_1}$, $\tilde{S}_{i_2,j_2}$ for the ground truth, and $\hat{\tilde{S}}_{i_1,j_1}$ and $\hat{\tilde{S}}_{i_2,j_2}$ for the generated motion, we proceed as follows. For each instance $\tilde{s}_u^{i_1,j_1} \in \tilde{S}_{i_1,j_1}$, where $1 \le u \le N_{i_1,j_1}$, we identify the instance $\tilde{s}_v^{i_2,j_2} \in \tilde{S}_{i_2,j_2}$, where $1 \le v \le N_{i_2,j_2}$, whose temporal span overlaps with $\tilde{s}_u^{i_1,j_1}$. If the intersection over union of their durations exceeds 50%, we consider $\tilde{s}_u^{i_1,j_1}$ and $\tilde{s}_v^{i_2,j_2}$ to occur synchronously. These synchronously occurring symbol pairs are collected into a sequence of combined symbol tuples, denoted by $\mathbb{S}_{j_1,j_2} = \{(\tilde{s}_u^{i_1,j_1}, \tilde{s}_v^{i_2,j_2})\}$, and similarly for the generated sequence $\hat{\mathbb{S}}_{j_1,j_2}$. Finally, the HMN similarity is computed as:

$$\text{Sim}_{\text{HMN}}(\mathbb{S}_{j_1,j_2}, \hat{\mathbb{S}}_{j_1,j_2}) = \text{Sim}_{\text{SMT}}(\mathbb{S}_{j_1,j_2}, \hat{\mathbb{S}}_{j_1,j_2}). \tag{7}$$

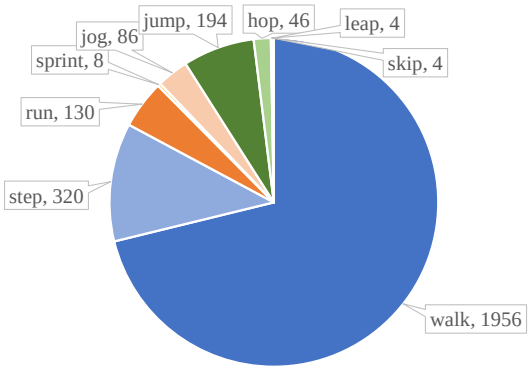


Figure 7. Distribution of action classes in the HumanML3D-Laban dataset.

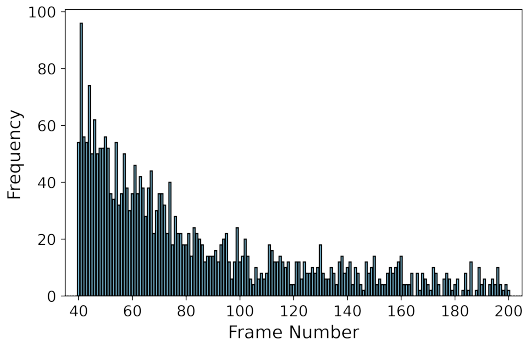


Figure 8. Distribution of the number of frames per motion sequence in the HumanML3D-Laban dataset.

Figure 9. **WordCloud** of the most frequent words in the HumanML3D-Laban descriptions, generated using the Python package `wordcloud` [12].

3.2. HumanML3D-Laban: Motion-Laban-Text Paired Dataset

This section presents the construction and characteristics of our HumanML3D-Laban dataset, including statistics on motion lengths, instructional text descriptions, and examples of text-motion pairs. Motion sequences were selected from the HumanML3D [6] dataset, covering mostly locomotions such as walking, running, stepping, and jumping. Following the annotation approach of BABEL-TEACH [2], these actions were further decomposed into atomic actions and annotated accordingly.

Semi-automatic Annotation. The annotation process was conducted in a semi-automatic manner. Initially, we manually inspected rendered HumanML3D motion sequences to extract detailed motion information, such as the number of steps, their body part sequence, and the action label (e.g., Walk forward: a three-step walk with the order: right, left, right). These details were stored in a JSON format with keys including "step number," "step order," and "action label." Subsequently, we utilised GPT-4.1 [1] to generate natural language descriptions by integrating the motion details. The prompt used for this step is shown in the second entry of Table 8. Finally, we paired the rephrased instructional text descriptions with their corresponding motions, forming the motion-laban-text pairs of the HumanML3D-Laban dataset. In accordance with the HumanML3D [6] evaluation protocol, we restricted the motion sequence lengths to between 40 and 200 frames.

Examples of motion details and motion-laban-text pairs. In the supplementary materials, we provide the complete set of instructional text descriptions that have been rephrased by an LLM. Specifically, within the "annotations" folder, each text file corresponds to a single motion instance and contains its associated instructional text description. For each motion instance, we used GPT-4.1 to generate five distinct rephrasings of the description, resulting in five different annotations per text file.

Statistics. Following the procedure in Guo et al. [6], all HumanML3D motion sequences were downsampled from 120 FPS to 20 FPS. We further filtered out motion sequences with fewer than 40 or more than 200 frames. As a result, the final dataset comprises 2,748 text-motion pairs, corresponding to approximately 18 hours of human motion data. Fig 7 illustrates the distribution of action labels, while Fig. 8 shows the distribution of frame lengths. Fig. 9 provides an overview of the most frequent words in the instructional text descriptions, drawn from a vocabulary of 1,220 unique words.

Table 1. The detailed structure of the Decoder in the Laban-Motion Encoder-Decoder module.

Encoder	Num.
In Dim.	512
Feat. Dim.	512
Depth	8
Head	8
Head Dim.	64
FFN Dim.	1024
Out Dim.	512

4. Experiment Details

4.1. Implementation Details

Network Architecture. For the Laban-Motion Encoder-Decoder, the Encoder operates as a rule-based, non-learning process, while the Decoder is implemented as a conventional Transformer-based decoder. The Decoder incorporates standard Attention modules [17], including Multi-headed Self-Attention blocks (MSAs), and Feed-Forward Network blocks (FFNs), with Layernorm (LN) applied before each module. The detailed architecture of the Decoder is summarised in Table 1, where "Head" and "Head Dim." refer to the number of attention heads and the dimensionality of the features in the MSA blocks, respectively. For the Motion Generator, we adopt the configuration from Huang et al. [7], Zhang et al. [21] for the Motion Generator. Specifically, a linear layer first projects the Laban code sequences, after which positional encoding is applied. The resulting sequence is then processed by a decoder-only Transformer comprising causal self-attention blocks.

Training settings. The Decoder and Generator are trained with AdamW optimiser (learning rate 1×10^{-4}, batch size 512). The Decoder is trained for 200k iterations, and the Generator is trained for 100k iterations. All experiments are conducted on a workstation equipped with 1 Intel Xeon Gold 6438Y+ CPU and 1 NVIDIA L40 GPU. We select the checkpoint with the lowest FID on validation for final evaluation.

Testing-time settings. Recall that, during all testing-time inference procedures, the first stage of our generation framework utilizes an LLM to compose motion plans at the conceptual level via retrieval-augmented prompting (see Sec. 3.3.1 in the main paper). Specifically, the retrieval pool exclusively consists of motion-Laban annotations from the training set; no Laban information from the evaluation set is disclosed to the LLMs. Consequently, at evaluation, the sequences of conceptual symbols are generated entirely by

the LLM, relying solely on the motion information and textual descriptions provided in the training set, as well as user input textual descriptions. The LLM then reasons over the alignment between textual concepts and symbolic motion patterns to generate a novel sequence of conceptual symbols based on the input text.

Hyperparameters. We adopt the motion feature extractor [6] to convert HumanML3D-Laban, HumanML3D, and KIT-ML motions into features of dimensions 263, 263, and 261. HumanML3D-Laban and HumanML3D share the same evaluator to get text and motion embeddings. Following [21], we set $\lambda = 0.5$. Laban codebook contains 37 Laban categories and 158 distinct codes, with a size of 158×512. We employ two commercial LLM models named GPT-4 mini (`gpt-4.1-mini-2025-04-14`) and GPT-4 (`gpt-4.1-2025-04-14`) from Achiam et al. [1], two open-source LLM models named Qwen3 (*qwen3-32b*) from Yang et al. [20] and DeepSeekV3 (*deepseek-v3*) from DeepSeek-AI [4] for composition. We use CLIP [14] (`ViT-B/32`) for text encoding.

4.2. Related to Large Language Models

In the main paper, LLMs are utilised to: (1) autonomously plan and compose Laban code sequences through explicit and interpretable symbolic reasoning (Sec. 4.2.1); and (2) semi-automatically generate instructional text descriptions for the HumanML3D-Laban dataset during annotation (Sec. 4.2.2). Here, we provide a detailed explanation of the procedures for using LLMs in Laban code sequences composition.

4.2.1. High-level Symbolic Motion Planning

Recall that we utilize LLMs to compose conceptual Laban symbol sequences through retrieval-augmented prompting for high-level symbolic motion planning. Specifically, the retrieved references (from the Conceptual Description database) describe motions using sequential Conceptual Descriptions (CDs) (Sec. 3.1.1 in the main paper). By semantically understanding these CDs and imitating their expression forms, LLMs are able to compose new CDs tailored to the target textual descriptions. According to the characteristics of LabanLite, the composed CDs can be unambiguously converted to Laban instances, which are then mapped to Laban codes (Sec. 3.1.2 in the main paper), and ultimately generate the corresponding motions via our LLM-Guided Text-Laban-Motion Generator (Sec. 3.3 in the main paper). Below, we detail the prompt design used for LLM-based symbolic motion composition. Examples of LLM outputs are provided in Sec. 5.3.1 of this supplementary material.

The prompt consists of the following components (as formulated in Table 8): (1) the structure definition of the retrieved CD references, (2) the definition of the CD structure, (3) several retrieved CD references, (4) the target textual description from user input.

Structure Definition of Retrieved CD References. We first define the format of the retrieved CD references. Each reference comprises `[number]`, `[Caption]`, `[Support]`, `[Left hand]`, and `[Right hand]`. Here, [number] indicates the reference index, `[Caption]` provides a high-level semantic description of the motion, and `[Support]`, `[Left hand]`, and `[Right hand]` respectively detail the motions of specific body parts, each in CD format.

Definition of CDs. We then specify the format for CDs as "`<body-part group> <moving semantic>` in `<time>` seconds", along with the complete set of possible CDs. This ensures that the LLM selects from a predefined set rather than inventing new CDs, which could hinder the downstream generation process. To optimize LLM token usage, we simplify the representation by introducing two lookup tables (see Table 16 and Table 17) for lower and upper human bodies, and use integer indices to denote CDs. With this approach, the LLM outputs the integer indices rather than full text, adhering strictly to the simplified CD definitions. Consequently, the output CDs are presented in the form "`(right, 2, 0.25)`", indicating `<body-part group>`, `<moving semantic>`, and `<time>`, respectively.

Retrieved CD References. Following the above CD definition, the simplified retrieved CD references are provided to the LLM.

Target Textual Description. Finally, the user's target textual description for the desired motion is also added to the prompt. The LLM composes the new motion by leveraging the high-level caption and the motion details from the retrieved CD references, along with the user's textual input.

4.2.2. Generate Texts for Laban Benchmark

In our Laban benchmark (Sec. 4 of the main paper), we construct a paired dataset comprising motion, Laban symbols, and textual instructional descriptions. Each data entry consists of: (1) a motion sequence, (2) its corresponding sequence of Laban symbols, and (3) a textual instructional description. The instructional texts are generated via a semi-automated annotation process.

To create these instructional descriptions, we first manually review the rendered SMPL motion videos and annotate critical motion details, including step count, step sequence, and action labels for each motion. These annotated motion details are then provided as input to an LLM (GPT4.1),

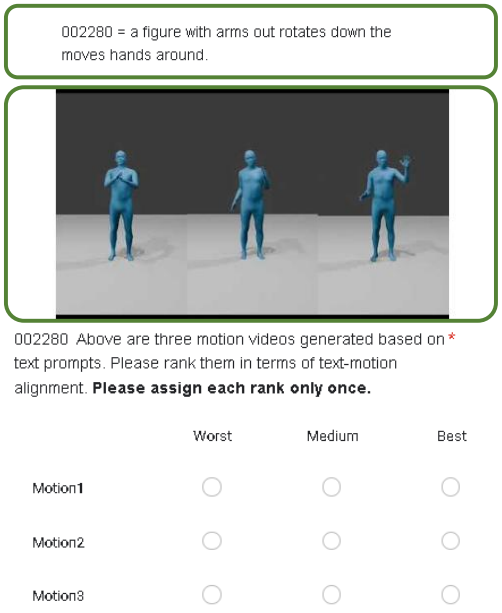


Figure 10. Illustration of the user study form.

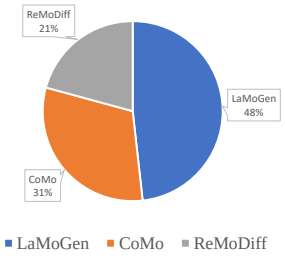


Figure 11. Illustration of the user study result.

which generates natural language instructional descriptions for each motion instance. As illustrated in Table 8 (Prompt #2), the LLM translates the annotated details into instructional text. For reproducibility and robustness evaluation, the LLM generates five instructional descriptions for each motion. All instructional texts produced by the LLM for the HumanML3D-Laban dataset are included in the supplementary material.

5. Experimental Results

5.1. User Study

We conducted a user study to evaluate the motion generation quality of LaMoGen (configured with GPT-4.1) in comparison with two state-of-the-art models for fine-grained text-to-motion generation: ReMoDiff [22] and CoMo [7]. Fifteen examples were randomly selected from the HumanML3D test set and rendered into video sequences. For each example, the results from LaMoGen, CoMo, and ReMoDiff were arranged side by side in a randomly determined order, forming a video triplet.

A total of 35 graduate students participated in the evaluation. Each participant was presented with all 15 video triplets and was asked to rank the three generated motions in each triplet as best, second-best, and worst, based on how well they matched the provided instructions, as shown in Fig. 10. Each ranking position was assigned to exactly one method, ensuring mutual exclusivity.

Statistical Analysis. For quantitative analysis, we assigned scores of 3, 2, and 1 to the best, second-best, and worst rankings, respectively. The average scores for each method across all motion sequences are presented in Fig. 11. Higher scores indicate stronger user preference. As shown in Fig.11, LaMoGen achieves the highest average user preference, followed by CoMo and ReMoDiff. Nearly half of the participants favoured motions generated by LaMoGen, indicating that our method provides superior text-motion alignment.

To determine whether the observed differences in user preferences among the three methods are statistically significant, we performed a Friedman test. The results indicate a significant difference in user preference ($\chi^2 = 387.380$, $p < 0.05$). Further analysis of the mean ranks demonstrates that LaMoGen achieves the highest mean rank (mean rank = 2.80), followed by CoMo (mean rank = 1.733) and ReMoDiff (mean rank = 1.457). These results suggest that participants consistently preferred LaMoGen over the other methods, with CoMo ranked second and ReMoDiff ranked last.

Table 2. Quantitative comparison for LLM usage cost. The approximate token count per motion sequence for generation and modification is reported.

Method	LLM-related Procedure	Objective	∼Token Count (k)
CoMo [7]	Decompose Texts	Generate	6.5
	Identify frames	Edit	51
	Identify body parts and edit	Edit	53
	Total		110.5
LaMoGen	Compose CDs	Generate	5
	Modify CDs	Edit	5
	Total		10

5.2. Quantitative Comparisons

5.2.1. LLM Usage Cost

We report the approximate token count utilized in LLM-based symbolic motion planning per motion sequence. We also compare our approach against CoMo [7], a related work that utilizes LLMs for motion generation, in terms of average token count. We reimplement the prompts described in the original paper of CoMo. We estimate the

count using OpenAI's token Calculator with GPT4.1. As shown in Table 2, our approach requires fewer tokens, indicating lower LLM demand and computational cost. This demonstrates more efficient LLM utilization.

Discussion. CoMo quantizes motion at only the pose level, resulting in a 392-code codebook. Each pose is encoded as a combination of body-part codes via the PoseScript [5] technique. In CoMo, there are two situations that require LLMs: (1) CoMo generates detailed textual descriptions of target motions; (2) CoMo enables users to edit motions using LLMs. During motion editing, CoMo requires LLM selection and modification from 392 pose codes, which consumes a large number of tokens, making LLM-based editing less efficient and cost-effective.

In contrast, our framework quantizes motions in both pose and temporal domains. Specifically, LabanLite defines 158 Laban codes based on Labanotation and further quantizes each sub-action in the temporal dimension, resulting in a more compact representation. Moreover, our proposed LaMoGen framework utilizes a two-level design that unifies high-level LLM-driven symbolic planning with low-level motion synthesis: LLMs are employed solely at the high-level composition stage, while a trained generator is used for low-level motion completion. This design substantially reduces LLM token consumption and operational cost.

Furthermore, during editing, unlike CoMo, which requires two separate LLM requests—one to identify frames that need modification and another to select and manipulate the corresponding body parts—LaMoGen enables direct modification in a single step. This is made possible by LabanLite's dual quantization in both the pose and temporal domains, allowing LLMs to directly edit sub-actions for each main body part rather than performing frame-wise and part-wise modifications as in CoMo. Therefore, our framework enables scalable, efficient motion planning while maintaining expressiveness and editability.

To summarize, this unique framework not only improves LLM token efficiency but also facilitates faster planning, reduces monetary cost, and makes large-scale deployment more feasible.

5.2.2. Inference Time

We follow the previous work [7] to assess the inference time using the Average Inference Time per Sentence (AITS). AITS measures the time (in seconds) to generate one motion sequence (not including model loading, dataset preparation, or LLM composition time). We compare our framework against CoMo, where CoMo has a similar architecture, on the HumanML3D test set. As shown in Table 5, our approach achieves a lower time cost, demonstrating the compactness of our Laban codebook and the effectiveness of our LaMoGen framework.

5.2.3. Generation Performance on HumanML3D

Table 3 reports the performance results using Laban metrics as well as conventional metrics (R-precision at Top-3, R@3, and FID) on the HumanML3D test set. Consistent with the main paper, we evaluate our model under four high-level composer conditions: (1) *None*, where motion generation is conditioned only on the input text, serving as a baseline for text comprehension; (2) two LLM composers (*GPT4.1mini* and *GPT4.1*), where generation is conditioned on both the text and conceptual cues provided by the LLMs; and (3) *Human*, in which conceptual cues are derived from ground-truth annotations to simulate human composition. We also report the decoded results from ground truth, denoted as *LaMoGen (Dec)*, where both conceptual and detailed Laban codes are provided to the decoder to indicate the upper bound of decoding performance. *Please note: the composer composes conceptual symbols, and the generator generates details symbols. The decoder decodes the combination of conceptual and detail codes.*

Although the HumanML3D text descriptions are not strictly instructional—in other words, they are typically descriptive and do not precisely specify target motions in terms of step count or timing, the performance in the *None* condition is therefore lower than the baselines.

Nevertheless, the use of retrieval-augmented prompting effectively addresses this limitation. For instance, when a user provides a descriptive input such as "a person walks in a circle", even though this prompt lacks detailed instructions like the number of steps or timing, the Conceptual Description database is able to retrieve related motions such as "walk forward" and "turn". The LLM can then compose these motions to generate a "walk in a circle" sequence. Furthermore, if an exact example like "walks in a circle" exists in the Conceptual Description database, the corresponding conceptual description can be retrieved directly, enabling precise motion synthesis.

As a result, LaMoGen achieves comparable or superior performance on Laban metrics relative to other methods.

5.3. Qualitative Results

5.3.1. Fine-Grained Temporal Structure Modification

We further evaluate each method's ability to model fine-grained temporal structure by modifying the input text with both moving context and vague/precise temporal control. Specifically, we compare our approach with CoMo [7] and MotionGPT [8] using the following settings: (1) "Walk forward" as the baseline, (2) "Walk forward slowly" as the vague speed change, and (3) "Walk forward in 5 seconds" as the precise time control. For each case, we also report the conceptual description composed by the LLM that corresponds to our results (we omit the retrieved conceptual

Table 3. Quantitative comparisons on the HumanML3D test set, using the proposed Labanotation-based metrics: Semantic Alignment (SMT), Temporal Alignment (TMP), and Harmonious Alignment (HMN), along with Text-to-Motion metrics: R-precision Top-3 (R@3) and FID. **Bold** and underlined values indicate the best and the second-best performance, respectively.

	Method	SMT ↑				TMP ↑				HMN ↑			R@3 ↑	FID ↓
		supL	supR	armL	armR	supL	supR	armL	armR	arm-arm	arm-sup	sup-sup		
	Real data	-	-	-	-	-	-	-	-	-	-	-	0.797	0.002
	LaMoGen (Dec)	0.843	0.843	0.745	0.731	0.848	0.848	0.779	0.765	0.537	0.631	0.842	0.793	0.095
	MDM [15]	0.413	0.413	0.276	0.315	0.372	0.372	0.369	0.308	0.177	0.284	0.502	0.611	0.544
	ReMoDiff [22]	0.455	0.455	0.348	0.405	0.335	0.405	0.367	0.320	0.179	0.275	0.501	0.795	**0.103**
	MoDiff [23]	0.482	0.482	0.386	0.360	0.401	0.401	0.388	0.334	0.190	0.299	0.513	0.782	0.630
	CoMo [7]	0.463	0.463	0.369	0.401	0.372	0.372	0.393	0.346	0.222	0.321	0.535	0.790	0.262
LaMoGen Composer	None	0.467	0.467	0.353	0.381	0.374	0.374	0.338	0.363	0.218	0.313	0.529	0.755	1.091
LaMoGen Composer	GPT4.1mini	0.437	0.437	0.331	0.354	0.517	0.517	0.463	0.445	0.270	0.371	0.561	0.779	0.561
LaMoGen Composer	GPT4.1	0.563	0.563	0.433	0.456	0.615	0.615	0.527	0.544	0.302	0.411	0.588	0.796	0.252
LaMoGen Composer	Human	**0.690**	**0.690**	**0.554**	**0.534**	**0.712**	**0.712**	**0.623**	**0.608**	**0.332**	**0.451**	**0.617**	**0.813**	0.206

reference, other prompts, and left/right-hand conceptual descriptions due to page limitations).

Fig. 12 presents the generated motions of our framework alongside those from other compared methods. For our approach, we report the corresponding LLM-composed textual descriptions. We measured the total duration of all generated motions and manually selected the keyframes at foot contact to visually indicate the number of steps. This allows us to estimate the average duration of each step and thus assess the degree of motion speed change. It can be observed that CoMo and MotionGPT struggle to accurately modify the temporal structure: CoMo does not effectively adjust the motion speed, and both CoMo and MotionGPT are unable to strictly execute motions according to the user-specified duration.

In contrast, our framework leverages the compositional capabilities of LLMs to explicitly determine the type and order of sub-actions, as well as to set the duration for each sub-action. For example, in the "Walk forward in 5 seconds" scenario, our method generates ten steps, allocating 0.5 seconds to each step to support long-term motion. The LLM will adaptively increase the number of steps as needed in order to fulfill the specified duration requirement. These results demonstrate that, thanks to the integration of LabanLite and the compositional reasoning capabilities of LLMs, our framework offers superior modeling of fine-grained temporal structure compared to existing approaches.

5.3.2. Visual Examples of Motion Editing

We demonstrate the potential of our framework in terms of motion editing capability. Fig. 13 presents examples of edited motions alongside the corresponding conceptual descriptions refined by the LLM. The refined part is also highlighted. As shown, the powerful ability of LLMs to interpret ambiguous instructions and understand dialogue context enables our framework to support interactive motion modification through natural language conversation. This capability, which allows users to intuitively edit motions during dialogue, represents a significant advancement over existing works.

Table 4. Ablation study of different numbers of top-matched references and masking ratios on the HumanML3D test set.

Top ref.	R@3 ↑	FID ↓	MM-Dist ↓	Multi-Mod. ↑
1	0.755	0.595	3.840	0.954
3	0.796	0.252	3.087	1.131
5	0.782	0.371	3.191	1.030
7	0.774	0.324	3.225	0.971
Mask rat.	R@3 ↑	FID ↓	MM-Dist ↓	Multi-Mod. ↑
0.1	0.789	0.258	3.008	1.092
0.3	0.796	0.252	3.087	1.131
0.5	0.787	0.299	3.211	1.011
0.7	0.772	0.345	3.290	0.992
0.9	0.705	0.457	3.302	0.985

5.3.3. Additional Comparison Examples

Additional comparison examples are shown in Fig. 14. From this figure, we can observe that ReMoDiff and CoMo occasionally violate chronological order and struggle to specify movement timings, while our method generates text-aligned motions. Comprehensive video results, including both successful examples and failure cases, are provided in the supplementary materials. These video results were also utilised in the user study.

5.4. Ablation Studies

Due to page limitations, the numerical results of our ablation studies are included in the supplementary material, while analyses and conclusions are presented in the main paper. Below, we reiterate the ablation settings together

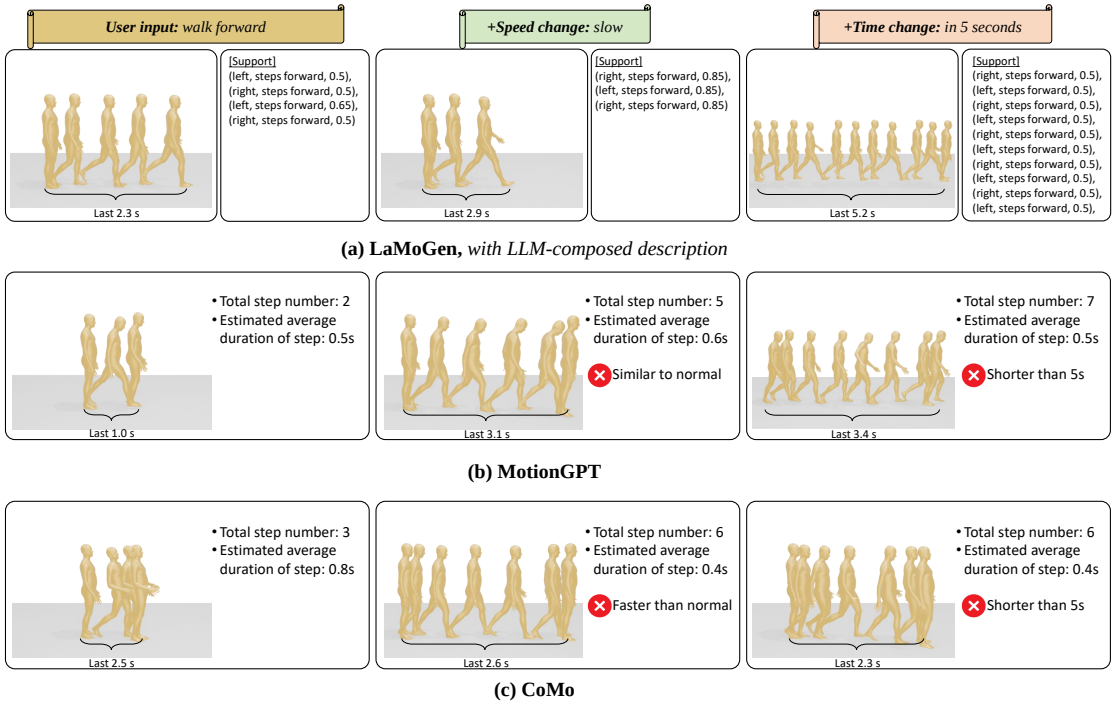


Figure 12. Qualitative comparison on fine-grained temporal structure modification. Left: the baseline motion, condition on "walk forward". Middle: the motion conditioned on "walk forward slow". Right: the motion conditioned on "walk forward in 5 seconds". We highlight the step number and the corresponding duration. We calculate the average step duration to assess the degree of the speed change.

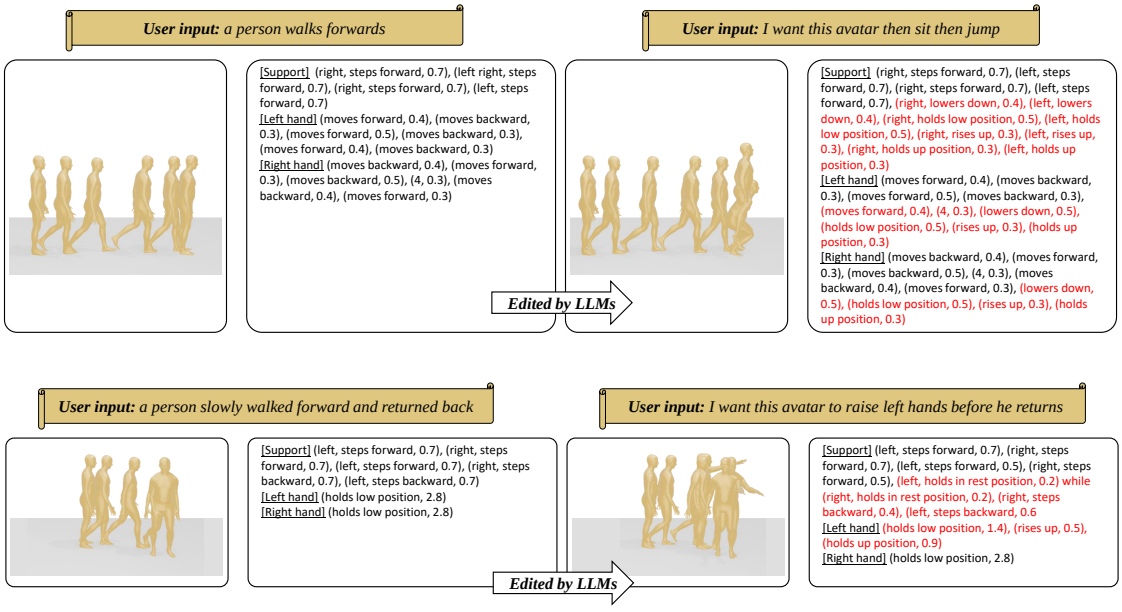


Figure 13. Visual examples for motion editing, with LLM-composed description. Left: the initial generated motion conditioned by the user input. Right: an ambiguous edit instruction is given to modify the motion. The corresponding LLM-composed description follows the LabanLite-defined format "<body-part group>, <moving semantic>, <time>". We highlight the refined part with red to show LLMs' dialogue context understanding capability.

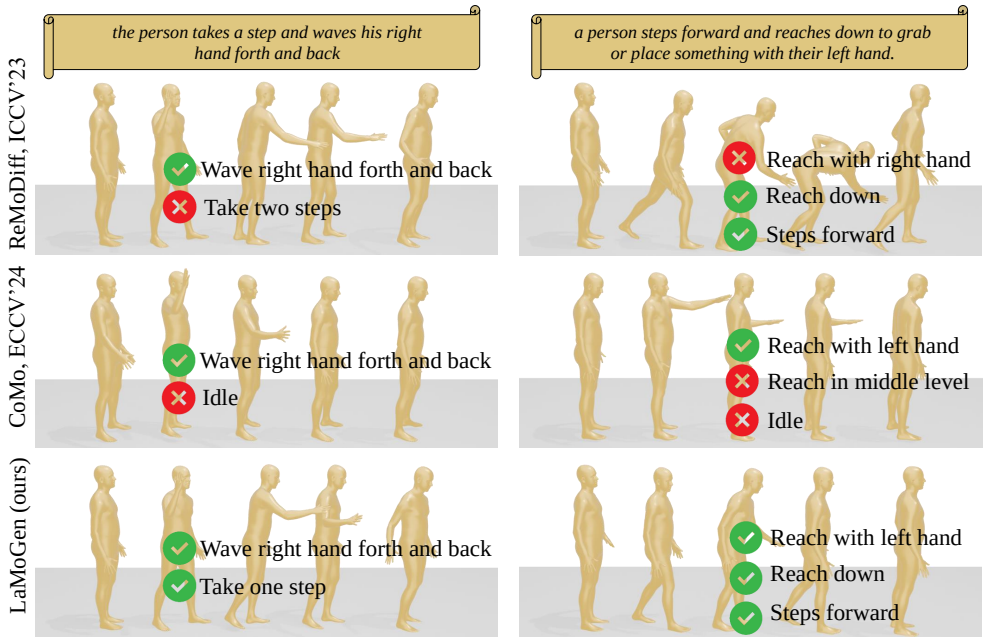


Figure 14. Qualitative comparisons on HumanML3D test sets, with motions progressing from left to right. Misalignments between text and generated motions are highlighted.

Table 5. Ablation study of different code dimensions on the HumanML3D test set. AITS measures the average inference time cost.

Code Dim	R@3 ↑	FID ↓	AITS ↓
CoMo [7]	0.790	0.262	0.58
256	0.759	0.310	0.28
384	0.783	0.295	0.32
512	0.796	0.252	0.45
768	0.802	0.241	0.57
1024	0.804	0.227	0.65

Table 6. Evaluation on Laban Benchmark. †: main paper setting.

Method	avg. SMT ↑	avg. TMP ↑	avg. HMN ↑	R@3 ↑	FID ↓
Real data	-	-	-	0.216	0.001
MDM	0.338	0.298	0.201	0.180	22.81
MDM_{lbn}	0.475	0.438	0.271	0.204	1.932
GPT4.1 †	0.533	0.501	0.392	0.208	1.861
$GPT4.1_{ip}$	0.496	0.467	0.328	0.200	3.408

with the corresponding quantitative findings. Additionally, we report the impact of the dimension of codebook entries.

Number of top-matched references. For LLM-guided conceptual composition, we provide top-matched conceptual examples to guide the composition of new conceptual codes via LLMs' retrieval augmentation prompting. We investigate how the number of retrieved examples affects performance on the HumanML3D test set using GPT-4.1. As shown in Table 4, increasing the number of examples from 1 to 3 consistently improves performance, indicating that LLMs benefit from sufficient references for accurate imitation. However, when increasing the number of examples further (e.g., from 3 to 5 or 7), we do not observe further improvement. This phenomenon is mainly caused by the limitation of the model's context window — the maximum number of tokens (words, symbols, etc.) that the LLM can process at once. When too many examples are provided, the beginning of the context may be truncated or, even within the window limit, earlier examples may be forgotten due to the model's recency bias. As a result, LLMs tend to focus more on the most recently presented cues and ignore in-

Table 7. Evaluation on HumanML3D. We examine the impact of varying the number of evenly spaced intervals for the Direction and Level symbols. †: main paper setting.

	Method	R@3 ↑	FID ↓	MM-Dist ↓	Div. →	Multi-Mod. ↑	Token Cost (k) ↓
	Real data	0.797	0.002	2.974	9.503	-	-
	MDM	0.611	0.544	5.566	9.559	2.799	-
	CoMo	0.790	0.262	3.032	9.936	1.013	6.5
Interval	3 †	0.796	0.252	3.087	9.124	1.131	5.0
Interval	5	0.796	0.198	3.026	9.859	1.360	7.7
Interval	7	0.800	0.184	2.992	9.840	1.413	10.3
	BERT	0.801	0.199	3.001	3.409	0.943	5.0
	Qwen3-8b	0.770	0.537	3.849	10.432	1.205	5.0

formation from previous examples, thereby diminishing the potential benefit of providing more reference cases.

Masking ratio on Laban codes. We also examine how the masking ratio on Laban codes affects performance during code generation. A higher masking ratio reduces the influence of conceptual cues, making motion generation rely more on text and less on Laban symbols, reducing the influence of conceptual guidance. We conduct this study on the HumanML3D test set, with GPT4.1 and top-3 retrieval settings. Comparing results in Table 4, a masking ratio of 0.3 offers the best balance and achieves optimal performance.

Impact of Code Dimension. We further investigate the impact of varying dimensions of the Laban code on the HumanML3D test set. Following the ablation findings above, the number of retrieved references is set to 3, and the masking ratio to 0.3, with GPT-4.1 as the LLM. We assess the generation quality (R@3, FID) and Average Inference Time per Sentence (AITS) across different code dimensions, and compare our approach to CoMo [7]. As shown in Table 5, increasing the code dimension improves generation performance but also increases inference time. Considering both factors, a code dimension of 512 is selected as the optimal setting.

Impact of Symbolic Guidance. To investigate the effect of symbolic guidance, we integrate LLM-composed symbolic codes into the baseline MDM model [15] by summing the activated code with the initial noise through the diffusion process. As reported in Table. 6 ("MDM$_{lbn}$"), this symbolic conditioning results in a substantial improvement: FID is reduced by a factor of 20 compared to the vanilla MDM. This dramatic gain demonstrates that leveraging high-level symbolic structure from language models can provide strong guidance for motion generation, significantly enhancing both overall fidelity and alignment between text and generated motion.

Impact on Instruction Prompting. Our LLM composition process relies on two components: Retrieval-Augmented Generation (RAG), which supplies templates for sub-action representations and temporal durations, and Instruction Prompting (IP), which defines the rules and definition on Conceptual Symbols. To evaluate the specific contribution of RAG, we conduct an ablation in which RAG is removed and the LLM is tasked with composing concepts solely based on IP instructions. Table. 6 "GPT4.1$_{ip}$" shows that this configuration results in a slight drop across most evaluation metrics. Importantly, we observe a much larger FID gap; without the concrete sub-action durations provided by RAG, the generated duration distributions diverge significantly from ground truth.

Impact of Discrete Intervals. To further examine the effect of discretization granularity in LabanLite, we conducted an ablation study varying the number of discrete intervals in our encoding scheme. As shown in Table. 7, the "Interval-7" configuration improves FID by 26%, but also doubles the LLM cost. It shows that, increasing the number of intervals can improve the reconstruction fidelity, but poses challenges for computational cost. Moreover, excessive granularity necessitates the introduction of nonstandard symbols, which diminishes the interpretability for human readers trying to understand the notation. Overall, our results suggest that a moderate number of intervals yields a favorable balance between fidelity, interpretability, and computational efficiency.

Choice of GPT-style vs. BERT-style Generation. In our framework, the availability of global information allows for both GPT-style (autoregressive) and BERT-style (infilling) generation strategies. As observed in Table. 7, adopting BERT-style generation ("BERT") improves the FID metric by 21%, reflecting enhanced accuracy in reconstruction. However, this comes at the expense of diversity, which decreases significantly by 62.6%. In contrast, GPT-style generation ("3 †") offers superior diversity, which is crucial for the downstream requirements of our task. Balancing these factors, we prioritize generation diversity and thus adopt GPT's autoregressive approach as our default method.

Performance on Weaker Language Models. To evaluate the robustness of our method with less powerful language models, we assess performance using the Qwen3-8b model, as reported in Table. 7. As expected, reducing LLM size to 8B leads to slightly lower overall generation metrics. Nevertheless, by incorporating RAG and IP constraints, our method successfully maintains valid Laban symbols, ensuring strong alignment between textual inputs and motion outputs. Notably, the Qwen3-8b configuration achieves a 26% improvement in R@3 compared to the MDM baseline. These results indicate our approach remains effective—even when deployed with weaker LLMs—highlighting its versatility for resource-constrained scenarios.

6. Discussion

To address the challenge posed by ultra-long text inputs (e.g., "walk slowly for 10 steps, then jump twice while waving both hands"), we can employ the LLM itself to segment complex user instructions into manageable sub-actions (e.g., "walk slowly for 10 steps" and "jump twice while waving both hands"). This decomposition is well within the capabilities of modern LLMs, and crucially, the semantic integrity of the original motion intent is preserved

during splitting. Once the ultra-long instruction is divided into discrete sub-actions, each segment can be processed independently, enabling LaMoGen to handle extended or sequential behaviors in a scalable manner. The concatenation of generated sub-actions naturally forms an ultra-long motion sequence, overcoming the constraints imposed by limited LLM context windows.

7. Limitations and Future Work

7.1. Need for a User Interface

Due to constraints in research funding and computational resources, the development of a user interface for editing Laban symbol sequences was beyond the scope of the present work. Nevertheless, we recognise that such an interface would be highly beneficial for promoting the broader adoption of the proposed LaMoGen framework and the LabanLite motion representation. As part of our future work, we plan to design and implement a user-friendly LaMoGen interface, which could be integrated as an add-on for 3D creation tools such as Blender.

7.2. Learning Cost of Labanotation

Learning Labanotation does entail a certain learning cost. While incorporating a larger number of Laban symbols allows for the expression of more complex motions, it also inevitably introduces more intricate notation rules. In the main paper, we address this challenge by proposing a two-stage LLM-Guided Text-Laban-Motion Generation Module: in the first stage, either human users or LLMs compose conceptual Laban instances to represent high-level motion; in the second stage, a Kinematic Detail Augmentor further refines these conceptual instances into detailed low-level motion. This two-stage approach simplifies the notation by requiring only conceptual symbols—approximately one-fifth of the full symbol set—and thus helps to reduce the learning effort. Although a certain amount of learning is still necessary, we believe that our approach is more accessible and convenient for human editing compared to existing methods based on key frames [3, 11, 19] or key points [18].

7.3. Potential for Disentangling Motion and Style

It is important to note that Labanotation applies quantization simultaneously at both the spatial and temporal levels. As a result, distinct performances—specifically, different individuals' personalized interpretations of the same contextual prompt—may be encoded using identical Laban symbols. This inherent abstraction implies that LabanLite, which is built upon Labanotation, does not explicitly represent fine-grained individual differences.

For instance, consider the context of "raising the hand." Younger individuals may execute this action energetically, characterized by a rapid initial lift followed by a deceleration as the hand approaches and stabilizes at the target height. In contrast, older individuals may perform the same movement with a composed steadiness, gradually lifting the hand at a constant speed from a lower to a higher position. These two performances exhibit distinctly different profiles in terms of acceleration and deceleration, with the former manifesting a vigorous style and the latter reflecting a calm, controlled demeanor. Notably, despite these pronounced differences in movement dynamics—each indicative of a unique personal style—the resulting motion sequences are encoded identically within the Laban symbol sequence.

Rather than viewing this as a limitation, we recognize it as an opportunity: by abstracting away such personalized nuances, LabanLite naturally separates the essential motion content from individual stylistic expression. Therefore, LabanLite holds significant potential for disentangling motion and style. In future work, we aim to further investigate this property by treating LabanLite-based motion descriptions as representations of neutral, canonical movements and subsequently modeling style-specific residuals to enable flexible motion restyling.

7.4. Potential for Flexible Extension Across Body Parts and Domains

The proposed LabanLite representation and LaMoGen framework possess significant flexibility in modeling human movement. While our current implementation focuses on the major body parts, LabanLite's codebook and design can be readily extended to model additional components, such as finger and toe gestures, simply by incorporating the corresponding Laban symbols. This adaptability is supported by recent studies, which have demonstrated the feasibility of Laban-based fine-grained motion reconstruction in sub-domains such as finger gesture estimation [9] and dance movement reconstruction [10]. Therefore, although our present work primarily targets daily motion, the methodology can be straightforwardly generalized to facilitate applications in a wide range of domains, including detailed human–object interactions and expressive dance analysis. We anticipate that future work will broaden the scope of LabanLite, establishing a unified motion representation applicable to diverse areas of human movement science and interactive systems.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 7, 8

[2] Nikos Athanasiou, Mathis Petrovich, Michael J Black, and Gül Varol. Teach: Temporal action composition for 3d humans. In *3DV*, pages 414–423. IEEE, 2022. 7

[3] Ho Yin Au, Junkun Jiang, and Jie Chen. Soscontrol: Enhancing human motion generation through saliency-aware symbolic orientation and timing control. *arXiv preprint arXiv:2601.14258*, 2025. 15

[4] DeepSeek-AI. Deepseek-v3 technical report, 2024. 8

[5] Ginger Delmas, Philippe Weinzaepfel, Thomas Lucas, Francesc Moreno-Noguer, and Grégory Rogez. Posescript: Linking 3d human poses and natural language. *IEEE transactions on pattern analysis and machine intelligence*, 2024. 10

[6] 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 *CVPR*, pages 5152–5161, 2022. 7, 8

[7] Yiming Huang, Weilin Wan, Yue Yang, Chris Callison-Burch, Mark Yatskar, and Lingjie Liu. Como: Controllable motion generation through language guided pose code editing. In *ECCV*, pages 180–196. Springer, 2024. 7, 9, 10, 11, 13, 14

[8] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. Motiongpt: Human motion as a foreign language. *Advances in Neural Information Processing Systems*, 36, 2024. 10

[9] Ling Li, WenRui Yang, Xinchun Yu, Junliang Xing, and Xiao-Ping Zhang. Translating motion to notation: Hand labanotation for intuitive and comprehensive hand movement documentation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 4092–4100, 2024. 15

[10] Min Li, Zhenjiang Miao, and Yuanyao Lu. Labanformer: Multi-scale graph attention network and transformer with gated recurrent positional encoding for labanotation generation. *Neurocomputing*, 539:126203, 2023. 15

[11] Jinpeng Liu, Wenxun Dai, Chunyu Wang, Yiji Cheng, Yansong Tang, and Xin Tong. Plan, posture and go: Towards open-world text-to-motion generation. *arXiv preprint arXiv:2312.14828*, 2023. 15

[12] Andreas Mueller. wordcloud: A little word cloud generator in python. https://github.com/amueller/word_cloud, 2015. Accessed: 2025-06-30. 6

[13] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer Vision and Pattern Recognition*, pages 10975–10985, 2019. 4

[14] 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. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 8

[15] 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*, 2023. 11, 14

[16] Muriel Topaz, Odette Blum, and Jessica Segall. *Elementary Labanotation: A study guide*. Dance Notation Bureau, 1996. 1

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 7

[18] Weilin Wan, Zhiyang Dou, Taku Komura, Wenping Wang, Dinesh Jayaraman, and Lingjie Liu. Tlcontrol: Trajectory and language control for human motion synthesis. In *ECCV*, pages 37–54. Springer, 2024. 15

[19] Tao Wang, Zhihua Wu, Qiaozhi He, Jiaming Chu, Ling Qian, Yu Cheng, Junliang Xing, Jian Zhao, and Lei Jin. Stickmotion: Generating 3d human motions by drawing a stickman. In *CVPR*, pages 12370–12379, 2025. 15

[20] 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. 8

[21] Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *CVPR*, pages 14730–14740, 2023. 7, 8

[22] Mingyuan Zhang, Xinying Guo, Liang Pan, Zhongang Cai, Fangzhou Hong, Huirong Li, Lei Yang, and Ziwei Liu. Remodiffuse: Retrieval-augmented motion diffusion model. In *ICCV*, pages 364–373, 2023. 9, 11

[23] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 11

Table 8. Prompts used in our LaMoGen to process natural language via Large Language Models.

Prompt #1: Conceptual cues composer
There are five digit collections describing movements, where each line consists of: [number] [Caption] - a general description of the motion sequence. [Support] - detailed descriptions of the movements of the supporting body parts, specifically the left and right feet, using a series of triplets. [Left hand] - detailed descriptions of the movements of the left hand, using a series of tuples. [Right hand] - detailed descriptions of the movements of the right hand, using a series of tuples. In the detailed descriptions, we specify the movement details for each body part and their duration in seconds. For the support movements, the details must be selected from these 54 categories: [1: steps to rest position, ..., 54: holds in knee-flexed backward diagonally to left position]. For the hand movements, the details must be selected from these 81 categories: [1: moves close to shoulder, ..., 81: moves relatively low backward diagonally to left]. For example, for the [Support] line, the triplet list would be like: (left, 1, 0.25), (right, 2, 0.25), (left, 1, 0.25) while (right, 2, 0.25). This means that the first movement is "left foot steps to rest position in 0.25 seconds". The second movement is "right foot steps forward in 0.25 seconds". The third movement is "left foot steps to rest position in 0.25 seconds while right foot steps forward in 0.25 seconds". For the [Left hand] line, the tuple list would be like: (1, 0.5), (2, 0.2). This means that the first movement is "left hand moves close to shoulder in 0.5 seconds" and the second movement is "left hand moves forward in 0.2 seconds". For the [Right hand] line, the structure and definition are similar to [Left hand] lines. Below is the main body of the digit collection describing the movements. You should strictly imitate the following content and create only one digit collection of *YOUR_INPUT*. Reply without explanation.
Prompt #2: Rephraser for generating motion descriptions
Rephrase the sentence creatively *YOUR_INPUT*. the step number is *YOUR_INPUT*, with the step order: *YOUR_INPUT*.

Table 9. Illustration of the Direction symbols and their corresponding partial semantics.

Name	Appearance	Semantics
Direction L/F		Move body part left forward
Direction M/F		Move body part to front
Direction R/F		Move body part right forward
Direction L/M		Move body part to left
Direction M/M		Move body part to middle
Direction R/M		Move body part to right
Direction L/B		Move body part left backward
Direction M/B		Move body part to back
Direction R/B		Move body part right backward

Table 10. Illustration of the Level symbols and their corresponding partial semantics.

Name	Appearance	Semantics
Level Hi.		Move body part to high level
Level Mi.		Move body part to mid-level
Level Lo.		Move body part to low level

Table 11. Illustration of the Hold symbols and their corresponding partial semantics. Note that if the attribute field for a symbol is left empty, it indicates that this body part is dynamic.

Name	Appearance	Semantics
Hold		Body part is stationary

Table 12. Illustration of the Orientation symbols and their corresponding partial semantics. Note that according to Labanotation, each symbol does not have a specific name. In LabanLite, we simply assign them sequential names for convenience.

Name	Appearance	Semantics
Orient 0		Body part orients at around $0°$
Orient 1		Body part orients at around $45°$
Orient 2		Body part orients at around $90°$
Orient 3		Body part orients at around $135°$
Orient 4		Body part orients at around $180°$
Orient 5		Body part orients at around $225°$
Orient 6		Body part orients at around $270°$
Orient 7		Body part orients at around $315°$

Table 13. Illustration of the Bend symbols and their corresponding partial semantics. Note that according to Labanotation, each symbol does not have a specific name. In LabanLite, we simply assign them sequential names for convenience.

Name	Appearance	Semantics
Bend 0		Body part bends at around 0°
Bend 1		Body part bends at around 30°
Bend 2		Body part bends at around 60°
Bend 3		Body part bends at around 90°
Bend 4		Body part bends at around 120°
Bend 5		Body part bends at around 150°

Table 14. Illustration of the Moving-effort symbols and their corresponding partial semantics. Note that according to Labanotation, each symbol does not have a specific name. In LabanLite, we simply assign them sequential names for convenience. If the attribute field for a symbol is left empty, it indicates "Moving-effort 1".

Name	Appearance	Semantics
Moving-effort 0		Body part moves very slow
Moving-effort 1	None	Body part moves in normal speed
Moving-effort 2		Body part moves fast
Moving-effort 3		Body part moves very fast

Table 15. Definition of the Laban codebook. Each codebook entry is described by its index (Code Idx.), associated Body-Part Group, corresponding SMPL key joint name (SMPL Joint), relevant attribute (Attribute), a marker indicating whether the attribute is conceptual (is Concpt.), and the Laban staff column name (Staff Col.).

Code Idx.	Body-Part Group	SMPL Joint	Attribute	is Concpt.	Staff Col.
$1 \sim 3$	Support-L	Left foot	Direction (L/M/R)	✓	Left support
$4 \sim 6$		Left foot	Direction (B/M/F)	✓	Left support
$7 \sim 9$		Left foot	Level (Lo./Mi./Hi.)	✓	Left support
$10 \sim 15$		Left knee	Bend	✗	Left leg gesture
$16 \sim 21$		Left hip	Bend	✗	Left leg gesture
$22 \sim 23$		Left foot & knee & hip	Hold	✓	Left support
$23 \sim 26$	Support-R	Right foot	Direction (L/M/R)	✓	Right support
$27 \sim 29$		Right foot	Direction (B/M/F)	✓	Right support
$30 \sim 32$		Right foot	Level (Lo./Mi./Hi.)	✓	Right support
$33 \sim 38$		Right knee	Bend	✗	Right leg gesture
$39 \sim 44$		Right hip	Bend	✗	Right leg gesture
$45 \sim 46$		Right foot & knee & hip	Hold	✓	Right support
$47 \sim 54$	Support-Both	Pelvis	Orient. Horiz.	✗	Body (Whole)
$55 \sim 62$		Pelvis	Orient. Vert.	✗	Body (Whole)
$63 \sim 67$		Pelvis	Moving effort Horiz.	✗	Body (Whole)
$68 \sim 72$		Pelvis	Moving effort Vert.	✗	Body (Whole)
$73 \sim 75$	Upper-L	Left hand	Direction (L/M/R)	✓	Left hand
$76 \sim 78$		Left hand	Direction (B/M/F)	✓	Left hand
$79 \sim 81$		Left hand	Level (Lo./Mi./Hi.)	✓	Left hand
$82 \sim 87$		Left elbow	Elbow Bend	✗	Left arm
$88 \sim 93$		Left shoulder	Shoulder Bend	✗	Left arm
$94 \sim 95$		Left hand & elbow & shoulder	Hold	✓	Left hand
$96 \sim 98$	Upper-R	Right hand	Direction (L/M/R)	✓	Right hand
$99 \sim 101$		Right hand	Direction (B/M/F)	✓	Right hand
$102 \sim 104$		Right hand	Level (Lo./Mi./Hi.)	✓	Right hand
$105 \sim 110$		Right elbow	Elbow Bend	✗	Right arm
$111 \sim 116$		Right shoulder	Shoulder Bend	✗	Right arm
$117 \sim 118$		Right hand & elbow & shoulder	Hold	✓	Right hand
$119 \sim 126$	Torso	Head	Orient. Horiz.	✗	Head
$127 \sim 134$		Head	Orient. Vert.	✗	Head
$135 \sim 140$		Spine2	Bend	✗	Body (Whole)
$141 \sim 143$	Upper-L	Left elbow	Direction (L/M/R)	✗	Left arm
$144 \sim 146$		Left elbow	Direction (B/M/F)	✗	Left arm
$147 \sim 149$		Left elbow	Level (Lo./Mi./Hi.)	✗	Left arm
$150 \sim 152$	Upper-R	Right elbow	Direction (L/M/R)	✗	Right arm
$153 \sim 155$		Right elbow	Direction (B/M/F)	✗	Right arm
$156 \sim 158$		Right elbow	Level (Lo./Mi./Hi.)	✗	Right arm

Table 16. Support body-part's moving semantic lookup table.

Index	Semantics	Index	Semantics
1	steps to rest position	28	holds in rest position
2	steps forward	29	holds in forward position
3	steps backward	30	holds in backward position
4	steps to right	31	holds in right position
5	steps to left	32	holds in left position
6	steps forward diagonally to right	33	holds in forward diagonally to right position
7	steps forward diagonally to left	34	holds in forward diagonally to left position
8	steps backward diagonally to right	35	holds in backward diagonally to right position
9	steps backward diagonally to left	36	holds in backward diagonally to left position
10	rises	37	holds in the raised position
11	rises to forward	38	holds in the raised forward position
12	rises to backward	39	holds in the raised backward position
13	rises to right	40	holds in the raised right position
14	rises to left	41	holds in the raised left position
15	rises forward diagonally to right	42	holds in the raised forward diagonally to right position
16	rises forward diagonally to left	43	holds in the raised forward diagonally to left position
17	rises backward diagonally to right	44	holds in the raised backward diagonally to right position
18	rises backward diagonally to left	45	holds in the raised backward diagonally to left position
19	knee flex	46	holds in knee-flexed position
20	knee flex forward	47	holds in knee-flexed forward position
21	knee flex backward	48	holds in knee-flexed backward position
22	knee flex right	49	holds in knee-flexed right position
23	knee flex left	50	holds in knee-flexed left position
24	knee flex forward diagonally to right	51	holds in knee-flexed forward diagonally to right position
25	knee flex forward diagonally to left	52	holds in knee-flexed forward diagonally to left position
26	knee flex backward diagonally to right	53	holds in knee-flexed backward diagonally to right position
27	knee flex backward diagonally to left	54	holds in knee-flexed backward diagonally to left position

Table 17. Arm body-part's moving semantic lookup table.

Index	Semantics	Index	Semantics	Index	Semantics
1	moves close to shoulder	28	holds close to shoulder position	55	moves relatively to previous position
2	moves forward	29	holds forward position	56	moves relatively forward
3	moves backward	30	holds backward position	57	moves relatively backward
4	moves to right	31	holds right position	58	moves to relatively right
5	moves to left	32	holds left position	59	moves to relatively left
6	moves forward diagonally to right	33	holds forward diagonally to right position	60	moves relatively forward diagonally to right
7	moves forward diagonally to left	34	holds forward diagonally to left position	61	moves relatively forward diagonally to left
8	moves backward diagonally to right	35	holds backward diagonally to right position	62	moves relatively backward diagonally to right
9	moves backward diagonally to left	36	holds backward diagonally to left position	63	moves relatively backward diagonally to left
10	rises up	37	holds up position	64	moves relatively up
11	rises to up forward	38	holds up forward position	65	moves relatively up forward
12	rises to up backward	39	holds up backward position	66	moves relatively up backward
13	rises to up right	40	holds up right position	67	moves relatively up right
14	rises to up left	41	holds up left position	68	moves relatively up left
15	rises up forward diagonally to right	42	holds up forward diagonally to right position	69	moves relatively up forward diagonally to right
16	rises up forward diagonally to left	43	holds up forward diagonally to left position	70	moves relatively up forward diagonally to left
17	rises up backward diagonally to right	44	holds up backward diagonally to right position	71	moves relatively up backward diagonally to right
18	rises up backward diagonally to left	45	holds up backward diagonally to left position	72	moves relatively up backward diagonally to left
19	lowers down	46	holds low position	73	moves relatively low
20	lowers to down forward	47	holds low forward position	74	moves relatively low forward
21	lowers to down backward	48	holds low backward position	75	moves relatively low backward
22	lowers to down right	49	holds low right position	76	moves relatively low right
23	lowers to down left	50	holds low left position	77	moves relatively low left
24	lowers down forward diagonally to right	51	holds low forward diagonally to right position	78	moves relatively low forward diagonally to right
25	lowers down forward diagonally to left	52	holds low forward diagonally to left position	79	moves relatively low forward diagonally to left
26	lowers down backward diagonally to right	53	holds low backward diagonally to right position	80	moves relatively low backward diagonally to right
27	lowers down backward diagonally to left	54	holds low backward diagonally to left position	81	moves relatively low backward diagonally to left