A. Implementation details

A.1. SMPL-X Joint-to-parameter conversion

This implementation provides a robust pipeline for recovering SMPL-X model parameters from 3D joint locations, a critical task in motion capture and human pose estimation workflows. The conversion process addresses the inverse kinematics problem of determining the underlying body model parameters that produce observed joint positions. The core challenge lies in finding the optimal SMPL-X parameters (global orientation, translation, and body pose) that minimize the discrepancy between the model's forward kinematics output and the target joint positions. This is formulated as an optimization problem where we iteratively refine parameter estimates to match the observed joints.

Initialization strategy The optimization begins with a carefully designed initialization phase to ensure convergence to meaningful solutions. The global orientation and translation are initialized using a batched Kabsch alignment algorithm, which finds the optimal rigid transformation between a template pose and the target joints. Specifically, we align the first three joints (typically root and hip joints) between a predefined template configuration and the input data. This provides a reasonable starting point for the global pose, accounting for the overall body orientation and position in space. The translation is further refined by compensating for the pelvis shift offset, which accounts for the difference between the SMPL-X model's origin and the actual pelvis location.

Table A1. Selected SMPL-X Body Joint IDs and Names

Joint ID	Joint Name
0	pelvis
1	left_hip
2	right_hip
3	spine1
4	left_knee
5	right_knee
6	spine2
7	left_ankle
8	right_ankle
9	spine3
10	left_foot
11	right_foot
12	neck
13	left_collar
14	right_collar
15	head
16	left_shoulder
17	right_shoulder
18	left_elbow
19	right_elbow
20	left_wrist
21	right_wrist
23	left_eye_smplhf
24	right_eye_smplhf
25	left_index1
34	left_ring1
40	right_index1
49	right_ring1

Optimization framework The parameter recovery employs gradient-based optimization using the Adam optimizer. Three sets of parameters are jointly optimized: the global orientation (represented as axis-angle vectors), the translation vectors, and the body pose parameters controlling the articulated joint rotations. The optimization runs for 300 iterations with a learning rate of 1e-3.

The objective function combines two key components. The primary term is the reconstruction loss, computed as the mean squared error between the predicted joint positions from the SMPL-X forward kinematics and the target joint locations. Only a subset of 28 semantically meaningful joints are used, listed in Tab. A1.

Temporal smoothness regularization To ensure physically plausible motion sequences, the implementation incorporates an angular velocity regularization term. This penalty discourages rapid changes in joint rotations between consecutive frames, promoting temporal smoothness in the recovered motion. The angular velocity is computed using rotation matrices and their relative transformations between frames. Specifically, the relative rotation between consecutive frames is extracted via matrix multiplication and converted back to axis-angle representation using logarithmic mapping from SO(3). The magnitude of these relative rotations, normalized by the time step, provides a measure of angular velocity that is penalized in the loss function.

Technical considerations The implementation leverages PyTorch3D's rotation utilities for efficient and numerically stable conversions between rotation representations. The SO(3) exponential and logarithmic maps ensure proper handling of rotation manifold geometry, avoiding singularities and discontinuities that can arise with naive Euler angle representations. The batched processing approach enables efficient optimization of entire motion sequences simultaneously, exploiting GPU parallelization while maintaining temporal consistency across frames.

The weighting between reconstruction accuracy and temporal smoothness (set to 0.03 for the angular velocity term) represents a careful balance between fitting the observed data and maintaining natural motion dynamics. This hyperparameter may require tuning based on the specific characteristics of the input data, such as noise levels and capture frequency.

A.2. Motion tokenizer

Architecture overview Our FSQ-based autoencoder (FSQAE) employs a hierarchical convolutional architec-

ture designed for efficient sequence compression and reconstruction. The model consists of three main components: an encoder network that progressively processes the input sequence, a finite scalar quantization (FSQ) module for discrete latent representation, and a decoder network that reconstructs the original sequence from the quantized codes.

Encoder architecture The encoder transforms input sequences of shape (B, N, D_in) into compact latent representations of shape (B, N, D_latent), where B denotes batch size, N represents sequence length, and D indicates dimensionality. We set the input motion to 85 channels, process through a hidden dimension of 256 channels, and compress to a 5-dimensional latent space.

The encoding pipeline begins with an initial feature projection using a one dimensional convolution with kernel size 3 and padding 1, mapping the 85 input channels to 256 hidden channels. The sequence then passes through two convolution stages. Each block consists of a 1D convolution with kernel size 7, stride 1, and padding 3, doubling the channel dimension while maintaining temporal resolution. The first convolution block expands from 256 to 512 channels, while the second expands from 512 to 1024 channels. Following the convolution stages, a residual block refines the features, and a final 1×1 convolution projects the 1024-channel representation to the 5-dimensional latent space for FSQ.

Residual convolutional blocks The core computational unit in both encoder and decoder is the residual convolutional block, which combines local convolution operations with position-wise MLPs. Each block maintains the input dimensions and consists of a 3×3 convolution with padding 1, followed by group normalization with 8 groups and GELU activation. The features then undergo per-position processing through an MLP module that includes layer normalization, expansion by a factor of 1 (maintaining the same width), GELU activation, and projection back to the original dimension. The block employs residual connections to facilitate gradient flow during training.

FSQ quantization The continuous latent representations are discretized using finite scalar quantization with levels [8, 8, 8, 6, 5] for each of the 5 latent dimensions. This configuration yields a codebook with 8×8×8×6×5 = 15,360 possible discrete codes. Each scalar component is independently quantized to its nearest level, enabling efficient indexing and lookup operations without requiring expensive nearest-neighbor searches typical of vector quantization approaches.

Decoder architecture The decoder mirrors the encoder structure, reconstructing the original sequence from the quantized latents. Starting from the 5-dimensional quantized representation, an initial 1×1 convolution projects to 1024 channels. After refinement through a residual block, two convolution stages progressively reduce the channel count while keeping the temporal dimension. The first convolution block uses transposed convolution with kernel size 7, stride 1, and padding 3 to transform from 1024 to 512 channels, while the second reduces from 512 to 256 channels. Each convolution block includes a subsequent residual block for feature refinement.

The final output projection consists of two stages: first, a convolution with GELU activation maintains the 256-channel representation, followed by a 1×1 convolution that projects back to the original 85-dimensional space. This two-stage projection helps ensure smooth reconstruction of the input sequences.

Training configuration The model processes variable-length sequences by maintaining the temporal dimension throughout the network, incorporating specific padding strategies in convolutional layers to strictly preserve the temporal resolution at every stage. This design enables the fully convolutional architecture to flexibly process arbitrary input durations during both training and inference, removing any requirements for specific length divisibility often associated with downsampling schemes. Group normalization is employed throughout the network to ensure stable training across different batch sizes, using 8 groups or fewer when channel dimensions are not divisible by 8.

B. Qualitative analysis of Gemini evaluation

To better understand the evaluation criteria and the discriminative power of our Gemini-based metric, we present concrete examples of how the model scores motion sequences. We focus on a complex composition task: *"Perform a cartwheel and then transition into a sword dance."*

The evaluation protocol assesses four key dimensions: semantic correctness, temporal order, transition smoothness, and physical plausibility. Below, we contrast a high-quality generation against a low-quality one to illustrate the scoring logic.

B.1. Case 1: High-quality generation

Observation: The character executes a clean cartwheel with proper hand placement, recovers balance naturally, and draws an imaginary sword to begin rhythmic slashing movements. The feet remain planted during the stance phase.

Gemini assessment:


```
{
    "correct": "all_correct",
    "order": "yes",
    "transition": "seamless",
    "physical": "good"
}
```

Score calculation: 10/10. Gemini correctly identifies that both actions are present (`all_correct`: +4), the sequence follows the text (`order`: +2), the recovery from the cartwheel into the dance is fluid (`transition`: +2), and there are no artifacts like foot sliding (`physical`: +2).

B.2. Case 2: Low-quality generation

Observation: The character performs a rolling motion that resembles a cartwheel but lacks arm support. Immediately after, the character snaps to a standing pose without an intermediate recovery frame and begins waving arms erratically. Significant foot sliding is observed during the "dance" phase.

Gemini assessment:

```
{
    "correct": "partially_correct",
    "order": "yes",
    "transition": "abrupt",
    "physical": "minor_issues"
}
```

Score calculation: 5/10. The score is penalized because the sword dance is not clearly recognizable (`partially_correct`: +2), the cut between actions is unnatural (`transition`: +0), and the motion exhibits sliding artifacts (`physical`: +1). While the order is technically correct (`order`: +2), the overall quality is accurately reflected as poor.

C. Alignment validation between Gemini and human evaluators

C.1. Experimental setup

To validate the use of Gemini as both dataset annotator and performance evaluator, we conducted an alignment study comparing Gemini's assessments with human expert judgments. We recruited four well-trained human evaluators with extensive backgrounds in motion analysis and computer animation. Prior to evaluation, all participants underwent a standardized training session with calibration exercises to ensure inter-rater consistency.

The evaluation dataset comprised 120 generated motion sequences from various state-of-the-art methods, divided into two categories: (1) 75 single action sequences covering locomotion, object manipulation, gestures, and full-body movements (3-8 seconds each), and (2) 45 multi-action compositions involving sequential action combinations with transitions (e.g., "walk, pick up object, sit down").

C.2. Evaluation protocol

Both human evaluators and Gemini assessed the semantic accuracy and physical realism of each sequence using a 10-point Likert scale. Human evaluators used a web-based interface allowing multiple viewpoints, adjustable playback speed, and unlimited replays. Gemini received structured prompts with identical evaluation criteria and contextual information.

C.3. Correlation analysis

We computed the Pearson correlation coefficient between Gemini scores and averaged human scores. For each motion sequence i, the mean human score was calculated as the mean of four scores:

$$H_i = \frac{1}{4}\sum_{j=1}^{4} h_{ij} \tag{A1}$$

where h_{ij} represents evaluator j's score for motion i.

The Pearson correlation coefficient was then computed as:

$$r = \frac{\sum_{i=1}^{n}(G_i - \bar{G})(H_i - \bar{H})}{\sqrt{\sum_{i=1}^{n}(G_i - \bar{G})^2 \sum_{i=1}^{n}(H_i - \bar{H})^2}} \tag{A2}$$

where G_i is Gemini's score, $\bar{G}$ and $\bar{H}$ are the respective means, and $n = 120$. Statistical significance was assessed using the t-statistic.

C.4. Results

The analysis yielded an overall Pearson correlation of $r = 0.89$ between Gemini and human evaluators, indicating strong statistical significance. Single action generations showed slightly higher correlation ($r = 0.91$) compared to multi-action compositions ($r = 0.83$), reflecting the increased complexity in evaluating transitions and long-term coherence.

Inter-rater reliability among human evaluators, measured using Krippendorff's alpha, was $\alpha = 0.82$, validating our human baseline quality. Individual evaluator correlations with Gemini ranged from 0.81 to 0.92, demonstrating consistent alignment across different evaluation styles. The correlation was strongest for locomotion-based actions ($r = 0.93$) and slightly lower for fine-grained manipulative actions ($r = 0.80$).

This strong alignment validates our methodology of leveraging Gemini throughout the motion generation pipeline, confirming that Gemini can serve as a reliable proxy for human judgment while enabling scalable and consistent evaluation.

D. Effect of loss function on tokenization

We conduct ablation studies on the local and global loss functions used for training motion tokenizer. We compare three different loss configurations: (1) global loss only, (2) local loss only, and (3) a combination of local and global losses. Surprisingly, the model trained with only global loss

Table A2. **Effect of loss function on tokenization.** Lower values indicate better performance for all metrics.

Method	Local Accuracy		Global Coherence		
	Rot. (°)	Pos. (cm)	Rot. (°)	Pos. (cm)	Vel. (cm/s)
Ours w/ local	**7.37**	9.22	18.70	15.82	17.10
Ours w/ local&global	7.61	9.18	10.97	9.72	16.06
Ours	7.55	**9.14**	**10.13**	**9.53**	**15.30**

demonstrates superior performance compared to other configurations.

The results are shown in Tab. A2. Training with global loss alone achieves the best reconstruction quality. This suggests that optimizing the absolute positions of joints in the global coordinate system is more effective for preserving motion characteristics than considering local joint relationships. The local-loss-only configuration shows degraded performance. This degradation likely occurs because local joint relationships alone cannot effectively capture the cumulative effects of joint rotations along kinematic chains, leading to error accumulation during motion reconstruction.

Contrary to our initial hypothesis, combining both local and global losses does not yield better results than using global loss alone. The combined loss configuration achieves performance metrics similar to the local-loss-only setting, suggesting that the addition of local constraints may actually interfere with the model's ability to optimize for global motion consistency.

E. Impact of model and data scale

We investigate the scaling properties of our method regarding model size and training data volume. As shown in Figure A1, the larger 7B model consistently achieves higher semantic scores across T2M, Concatenation, and Editing tasks compared to the 3B version. Figure A2 demonstrates that increasing the training data from 1/8 to the full dataset yields continuous performance gains, highlighting the data-driven nature of the approach.

F. MotionGB dataset statistics

We provide a comprehensive statistical analysis of the MotionGB dataset, focusing on motion quality, temporal distribution, and semantic diversity.

Motion quality High-quality motion generation requires training data with physical plausibility and minimal noise. We utilize the Mean Jerk metric (m/s^3) to quantify motion smoothness. As illustrated in Figure A3, MotionGB achieves the lowest mean jerk of **38.13**, significantly outperforming existing large-scale datasets like Motion-X (237.51) and MotionMillion (68.16). This indicates that our filtering pipeline effectively removes jitter and artifacts, resulting in smoother and more realistic motion primitives.

Sequence length distribution To ensure the model generalizes across different temporal scales, MotionGB

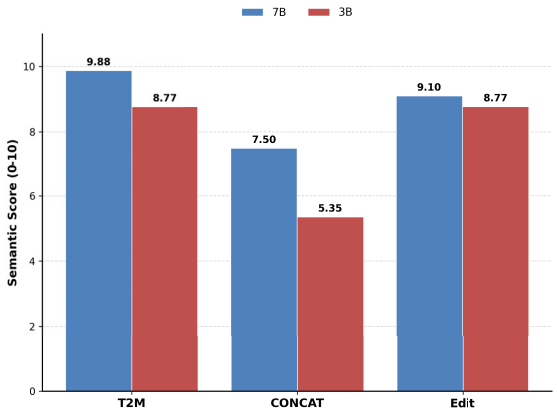

Figure A1. **Effect of model size.** Comparison of semantic scores between 7B and 3B parameter models. The 7B model shows superior performance across all three tasks (T2M, CONCAT, Edit).

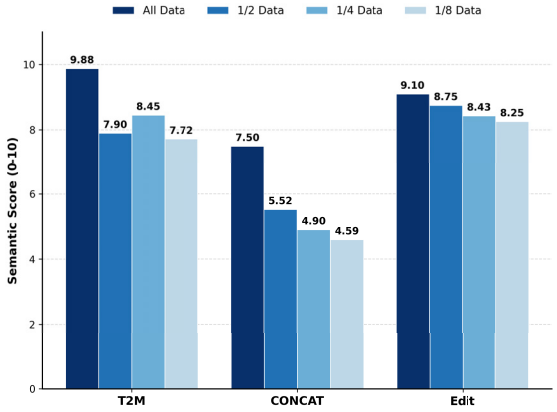

Figure A2. **Effect of data scaling.** Performance evaluation using varying fractions of training data (All, 1/2, 1/4, 1/8). The model benefits significantly from larger-scale data training.

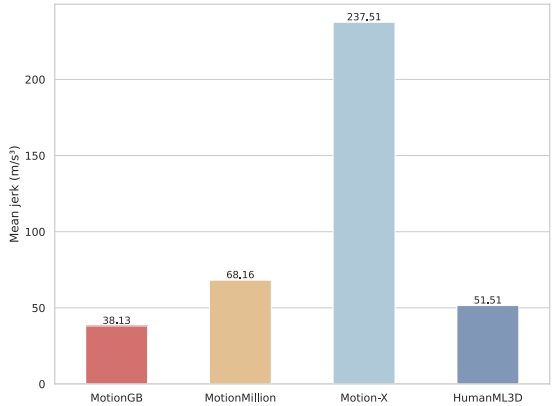

Figure A3. **Mean jerk comparison.**

comprises sequences with highly variable durations. Figure A4 presents the length distribution, ranging from short atomic actions ($< 2s$) to long-term sequences extending up to **20 seconds**. The cumulative distribution curve shows a

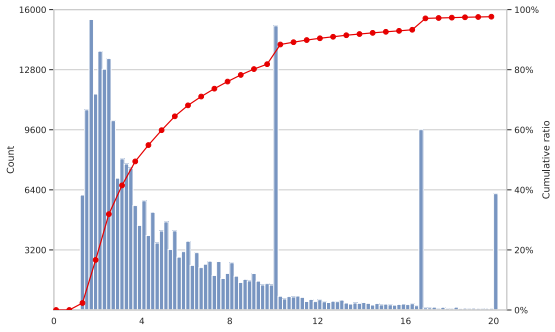

Figure A4. **Motion length distribution of MotionGB.**

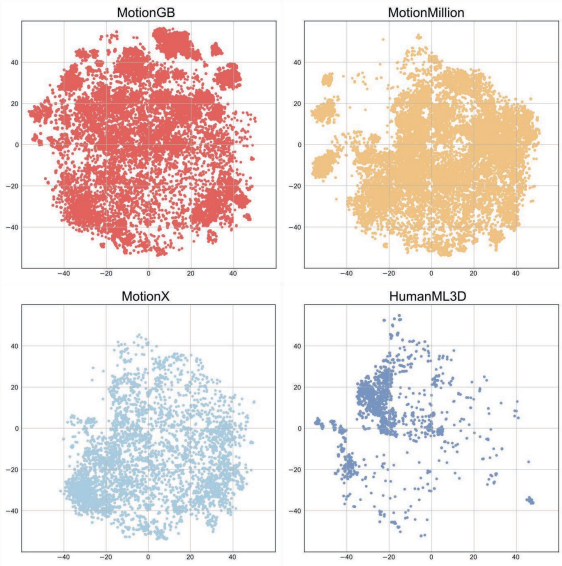

Figure A5. **Motion semantic comparison using t-SNE.**

healthy balance, with specific peaks (e.g., at 10s and 20s) indicating the inclusion of extended, uninterrupted motion clips, which are crucial for learning long-term temporal dependencies.

Pose diversity We visualize the pose diversity of MotionGB by performing t-SNE on body part poses, comparing it with MotionMillion, Motion-X, and HumanML3D. As shown in Figure A5, MotionGB exhibits a dense and expansive pose distribution (shown in red), covering a significantly broader action space than HumanML3D (blue). The cluster density and spread are comparable to the largest available datasets (MotionMillion), confirming that MotionGB captures a rich variety of action categories required for open-vocabulary generation.

G. More qualitative results

Please refer to the **supplementary video** for qualitative results demonstrating our method's performance on both motion generation and editing tasks, as well as visual comparisons with baseline approaches.