

Investigating Implicit Inverse Kinematics in LGTM: A Text-Driven Motion Diffusion Model

組別: 第30組

組員: 林彥亨、林庭瀛

Abstract

This report details our efforts to re-implement the LGTM (Local-to-Global Text-Driven Human Motion Diffusion Model) introduced by Sun et al. (2024). Emphasis is placed on critical implementation steps, specifically preparing the HumanML3D dataset, training Text–Motion Retrieval (TMR) encoders, and accurately reproducing the LGTM diffusion model. We present a comprehensive pipeline, document encountered challenges, and propose solutions to ensure reproducibility.

1. Introduction

Our project aims to:

1. Faithfully reproduce the LGTM model and validate its results against the original implementation.
2. Establish a detailed pipeline encompassing dataset preparation (HumanML3D), TMR encoder pre-training, and inference.

2. Background

2.1 LGTM Overview

LGTM employs a hierarchical diffusion framework, decomposing textual motion descriptions into six localized narratives corresponding to distinct human body parts (head, torso, limbs). Each narrative is processed using specialized TMR encoders, which are subsequently combined through a global attention mechanism to synthesize cohesive human motions.

2.2 HumanML3D Dataset

HumanML3D is a publicly available benchmark dataset specifically designed for text-to-3D human motion generation and retrieval tasks. Introduced by [Guo et al., CVPR 2022], it serves as a cornerstone dataset for learning the mapping between natural language descriptions and semantically aligned 3D human motion sequences.

ASSAM stands for "A Semi-Automated Pipeline for Semantic Annotation of Human Motion", and it is the data collection and annotation framework used to create the HumanML3D dataset. ASSAM is a semi-automated pipeline developed by Guo et al. (2022) to facilitate the large-scale creation of text-motion pairs. Its goal is to generate natural language descriptions for 3D human motion sequences efficiently and accurately.

It addresses a key challenge in motion-language datasets: how to generate high-quality textual annotations for complex human motions at scale.

Feature	Description
Text Annotations	14,616 natural language descriptions written by humans that describe human actions and behaviors
Motion Sequences	SMPL-based 3D motions, about 42 hours of data
Motion Length	Each clip ranges from 2 to 10 seconds (20–200 frames at 20 FPS)
Skeleton	22-joint human skeleton extracted from SMPL format
Motion Representation	Includes root velocity, joint positions, joint rotations, and contact information
Splits	Predefined train/val/test sets for fair comparison
Alignment	Each motion sequence is tightly aligned with its text description in semantics and timing

The **HumanML3D** dataset pairs natural language descriptions with 3D human motion sequences using a structured format designed for text-to-motion tasks. Each motion clip is stored as a .npz file containing joint positions, 6D rotations, root velocities, contact flags, and optional features like joint velocities—based on the SMPL model with 22 joints and frame lengths ranging from 2 to 10 seconds. Corresponding textual descriptions, written by humans, are provided in .txt or .json files, often with multiple captions per motion. The dataset also includes metadata such as joint mappings and predefined train/validation/test splits, making it ready for use in training and evaluating multimodal models.

Types of Text Descriptions

- Human-written, not template-generated
- Includes both **simple** and **composite** actions
 - “*A person walks forward*”
 - “*The person jumps and waves their right hand*”
- Reflects **temporal order** and **spatial detail**
 - “*First turns to the left, then kneels*”
 - “*Raises both arms above the head*”
- Mentions **specific body parts** (e.g., arms, legs, head)

2.3 TMR Encoders

TMR (Text-to-Motion Representation) encoders are contrastive models that embed both textual and motion data into a shared semantic latent space. In LGTM, TMR transforms whole-body and body-part descriptions into embeddings that serve as conditioning signals for the diffusion model. By assigning a dedicated TMR encoder to each body part, LGTM ensures that both global and local textual semantics are accurately captured, providing robust semantic grounding for the motion generation process.

2.4 fmbvh Library

The **fmbvh** package is used extensively in LGTM for motion data processing. It provides tools for parsing **.bvh** motion files and extracting features such as joint offsets, translations, and quaternions. These features are further converted to 6D rotation matrices and used for differential forward kinematics.

The library also supports simple motion visualization and video export. LGTM uses these functionalities to transform raw motion data into formats suitable for model input and output. TMR encoders are contrastive models trained to embed textual and motion data into a shared latent space. LGTM specifically leverages individual TMR encoders for each body part, providing a robust semantic grounding for the diffusion process.

3. Methodology

3.1 Environment Setup

- We conducted LGTM experiments on WSL2 using Python 3.10, PyTorch 2.2, and CUDA 12.1 with NVIDIA RTX 4060 GPU.
- We conducted HumanML3D datasets on WSL2 using Python 3.7.10, PyTorch

3.2 Data Preparation

The **HumanML3D** dataset required obtaining raw motion data from the **AMASS** dataset collection. We followed the instructions to download specific sub-datasets including CMU, HumanEva, MPI_HDM05, and KIT-ML from the [AMASS website](#). Each sub-dataset contains BVH-formatted motion sequences that are essential for constructing HumanML3D.

After downloading, we organized the files into the expected folder structure and ran the provided preprocessing notebooks (**raw_pose_processing.ipynb**, **motion_representation.ipynb**, and **cal_mean_variance.ipynb**). These notebooks convert raw motion sequences into a unified representation format, normalize motion lengths and joint definitions, and compute statistical parameters needed for model training. This step ensured data consistency and facilitated effective training.

3.3 TMR Pre-training

To prepare data for **TMR (Text-to-Motion Representation) encoder** training, we utilized the **prepare_data_model.sh** script provided in the LGTM repository. This automated script handled multiple preprocessing tasks:

1. Downloading Dependencies: GloVe embeddings and part-level annotations were downloaded using **gdown**.
2. Body Part Text Splitting: Text prompts were split into six body parts (head, torso, left/right arms, left/right legs) using a GPT-based annotation tool.
3. Motion Feature Extraction: 6D motion features were computed for both full-body and part-level motions using **compute_guoh3dfeats** and **compute_body_part_guoh3dfeats**.
4. Text Embedding Computation: CLIP-based embeddings were extracted from both global and part-level textual descriptions.
5. Statistical Normalization: Mean and standard deviation were computed for each set of motion features.
6. Pretrained Model Setup: Pretrained TMR encoders and LGTM model checkpoints were downloaded and extracted into the appropriate directories.

This script significantly reduced setup complexity and ensured consistent preparation across multiple stages of the TMR training pipeline.

3.4 LGTM Inference

Instead of training the LGTM model from scratch, we utilized the official pretrained model weights provided by the authors. These weights were downloaded and configured using the **prepare_data_model.sh** script.

To perform inference and generate motion outputs, we directly executed the **playground.ipynb** notebook. This notebook allowed us to input textual prompts, visualize generated motion sequences, and export results in **.bvh** and video formats. The LGTM inference pipeline automatically handled part-level text segmentation, embedding computation, diffusion denoising, and motion synthesis.

While some components in the notebook remain under development and may require additional adjustments, we were able to successfully produce sample motion sequences that align well with the given textual descriptions.

4. Results

4.1 Understanding the Nature of HumanML3D Data

Before presenting our inference results, it's important to clarify a key characteristic of the HumanML3D dataset: **it does not include inverse kinematics (IK) parameters by default**. The dataset provides 3D joint positions for each frame, which originate directly from existing motion capture (mocap) sequences. As such, it lacks explicit end-effector targets or per-skeleton bone lengths necessary for recalculating IK.

Therefore, while the data can be directly rendered using forward kinematics (FK), it is not intended to be regenerated via IK solvers. This distinction matters because LGTM does not attempt to perform inverse kinematics explicitly—it learns to generate motion sequences by aligning text semantics to joint-level outputs already formatted for FK.

4.2 Prompt Design for IK-like Control

Although LGTM does not implement explicit IK, users can still encourage IK-like behavior by carefully crafting textual prompts. The following table outlines several strategies for encoding spatial constraints or body-part targeting through natural language:

Strategy	Description	Example
Specify end-effector target	Clearly state which body part should move where	“Raise the right hand and touch the left shoulder”
Use spatial prepositions	Use words like above, below, in front of, to the side	“Lift the right foot above the ground and hold”
Mention fixed body parts	Explicitly tell which parts are static or supporting	“a man keeps both feet planted while reaching forward”
Describe orientation/angle	If possible, describe the direction or pose	“Bend the torso forward at a 45-degree angle”
Sequence actions clearly	Use clear temporal structure to disambiguate poses	“First, raise the left hand; then place it on the head”

4.3 Examples of IK-like Prompts

Below are examples comparing IK-like prompts with FK-style descriptions, showing how textual phrasing can shape motion intent.

Id	IK-like Prompt	FK-like Prompt	Result
1	a man raises the right hand and touches the left shoulder (video)	a man swings the right arm across the chest with elbow bent (video)	The FK-style prompt produced a more accurate motion result.
2	a man lifts the right foot above the ground and holds (video)	a man lifts the right thigh and bends the knee to raise the foot (video)	The IK-like prompt appeared less natural; the FK-like prompt generated more coherent motion.
3	a man keeps both feet planted while reaching forward (video)	a man leans forward and stretches both arms straight ahead. (video)	There was no significant difference between IK-like and FK-like prompts.

4	a man bends the torso forward at a 45-degree angle (video)	a man bends at the hips, tilting the upper body forward (video)	Body bending at a 90-degree when the input is FK-like prompt; Body bending at a 15-degree in IK-prompt.
5	a man first raises the left hand; then place it on the head (video)	a man lifts the left arm upward, then bends the elbow to rest the hand (video)	Both of IK-like and FK-like prompts can produce accurate and natural results.

5. Challenges and Solutions

5.1 HumanML3D Dataset Preparation

One of the first challenges we encountered was preparing the HumanML3D dataset. The dataset requires downloading and integrating multiple motion capture sub-datasets from the AMASS project. These datasets use varying skeleton structures and naming conventions, which can lead to compatibility issues during preprocessing.

We resolved this by strictly following the HumanML3D preprocessing pipeline and verifying all intermediate outputs, such as mean and variance files, pose representations, and mirrored motion augmentation.

5.2 TMR Encoder Setup

Setting up the TMR encoders involved multiple steps, including downloading GloVe embeddings, computing full-body and part-level motion features, and generating corresponding text embeddings. A common issue was incorrect file paths or broken symbolic links, especially when creating **pose_data** and **annotation folders**.

We addressed this by carefully inspecting the **prepare_data_model.sh** script and verifying symbolic links manually.

5.3 Executing playground.ipynb

The playground.ipynb notebook, although central to inference, is still under development and lacks robustness. Several cells require manual editing, and default paths are often incorrect or out of sync with current folder structures.

We solved this by adding path sanity checks and step-by-step validation of modules .

5.4 Designing IK-like and FK-like Prompts

Finally, designing effective prompts that simulate IK behavior was a conceptual challenge. Since the model does not support explicit inverse kinematics, we needed to embed spatial constraints naturally into the language. Crafting clear and unambiguous IK-like prompts required iterative testing and tuning. In several cases, FK-style descriptions

produced more accurate or plausible results. We documented this in Section 4.3, which compares paired prompts and their qualitative outcomes.

6. Conclusion

In this project, we successfully reproduced the LGTM (Local-to-Global Text-Driven Human Motion Diffusion Model) framework using the official pretrained weights and verified its functionality through controlled experiments. We built a complete and reproducible pipeline for data preparation, model setup, and inference execution. By exploring how textual prompts influence the resulting motion—especially with respect to implicit inverse kinematics (IK)—we identified key linguistic patterns that enhance spatial alignment in generated motion.

Through comparative experiments on IK-like versus FK-like prompts, we observed that FK-style descriptions often produced more accurate or natural results under the current LGTM architecture. These findings suggest that while LGTM does not perform explicit IK, it can be guided through careful textual design to mimic IK-driven behaviors.

References

- Sun, H. et al., LGTM: Local-to-Global Text-Driven Human Motion Diffusion Model, SIGGRAPH, 2024. arxiv.org/abs/2405.05149.
- LGTM, [GitHub](https://github.com/lllyasviel/LGTM).
- HumanML3D, [GitHub](https://github.com/lllyasviel/HumanML3D).
- TMR, [GitHub](https://github.com/lllyasviel/TMR).
- fmbvh, [Github](https://github.com/lllyasviel/fmbvh).