École des Ponts
ParisTech

PERCEIVING SYSTEMS
MAX PLANCK INSTITUTE FOR
INTELLIGENT SYSTEMS

THÈSE DE DOCTORAT
de l'École des Ponts ParisTech

Natural Language Control for 3D Human Motion Synthesis

École doctorale N°532, Mathématiques et Sciences et Technologies de l'Information et de la Communication (MSTIC)

Spécialité de doctorat: Informatique

Thèse préparée au sein du Laboratoire d'Informatique Gaspard-Monge et du Max Planck Institute for Intelligent Systems.

Thèse soutenue le 17 avril 2024, par
Mathis Petrovich

Composition du jury:

Edmond Boyer Inria/Meta Reality Labs	*Président du jury*
Francesc Moreno-Noguer Universitat Politècnica de Catalunya (UPC)	*Rapporteur*
Kris Kitani Carnegie Mellon University (CMU)	*Rapporteur*
Siyu Tang ETH Zürich	*Examinatrice*
Mathieu Aubry École des Ponts ParisTech (ENPC)	*Directeur de thèse*
Michael J. Black Max Planck Institute for Intelligent Systems	*Co-directeur de thèse*
Gül Varol École des Ponts ParisTech (ENPC)	*Co-directrice de thèse*

Ce manuscrit est dédié à ma famille.

Abstract

3D human motions are at the core of many applications in the film industry, healthcare, augmented reality, virtual reality and video games. However, these applications often rely on expensive and time-consuming motion capture data.

The goal of this thesis is to explore generative models as an alternative route to obtain 3D human motions. More specifically, our aim is to allow a natural language interface as a means to control the generation process. To this end, we develop a series of models that synthesize realistic and diverse motions following the semantic inputs.

In our first contribution, described in Chapter 3, we address the challenge of generating human motion sequences conditioned on specific action categories. We introduce `ACTOR`, a conditional variational autoencoder (VAE) that learns an action-aware latent representation for human motions. We show significant gains over existing methods thanks to our new Transformer-based VAE formulation, encoding and decoding SMPL pose sequences through a single motion-level embedding.

In our second contribution, described in Chapter 4, we go beyond categorical actions, and dive into the task of synthesizing diverse 3D human motions from *textual descriptions* allowing a larger vocabulary and potentially more fine-grained control. Our work stands out from previous research by not deterministically generating a single motion sequence, but by synthesizing multiple, varied sequences from a given text. We propose `TEMOS`, building on our VAE-based `ACTOR` architecture, but this time integrating a pretrained text encoder to handle large-vocabulary natural language inputs.

In our third contribution, described in Chapter 5, we address the adjacent task of text-to-3D human motion *retrieval*, where the goal is to search in a motion collection by querying via text. We introduce a simple yet effective approach, named `TMR`, building on our earlier model `TEMOS`, by integrating a contrastive loss to enhance the structure of the cross-modal latent space. Our findings emphasize the importance of retaining the motion generation loss in conjunction with contrastive training for improved results. We establish a new evaluation benchmark and conduct analyses on several protocols.

In our fourth contribution, described in Chapter 6, we introduce a new problem termed as "multi-track timeline control" for text-driven 3D human motion synthesis. Instead of a single textual prompt, users can organize multiple prompts in temporal intervals that may overlap. We introduce `STMC`, a test-time denoising method that can be integrated with any pre-trained motion diffusion model. Our evaluations demonstrate that our method generates motions that closely match the semantic and temporal aspects of the input timelines.

In summary, our contributions in this thesis are as follows: (i) we develop a generative variational autoencoder, `ACTOR`, for action-conditioned generation of human motion sequences, (ii) we introduce `TEMOS`, a text-conditioned generative model that synthesizes diverse human motions from textual descriptions, (iii) we present `TMR`, a new approach for text-to-3D human motion retrieval, (iv) we propose `STMC`, a method for timeline control in text-driven motion synthesis, enabling the generation of detailed and complex motions.

Résumé

Les mouvements humains 3D jouent un rôle clé dans divers domaines, tels que le cinéma, le secteur médical, la réalité augmentée, la réalité virtuelle et l'industrie du jeu vidéo. Toutefois, ces utilisations reposent souvent sur des données de capture de mouvement coûteuses et chronophages.

L'objectif de cette thèse est d'explorer les modèles génératifs en tant que voie alternative pour obtenir des mouvements humains 3D. Plus spécifiquement, notre objectif est de contrôler le processus génératif par le biais d'une interface en langage naturel. Pour cela, nous développons une série de modèles qui synthétisent des mouvements réalistes et variés en suivant des entrées sémantiques.

Dans notre première contribution, décrite dans le Chapitre 3, nous relevons le défi de générer des séquences de mouvements humains conditionnées par des catégories d'actions spécifiques. Nous présentons `ACTOR`, un autoencodeur variationnel conditionnel (VAE) qui apprend une représentation latente des mouvements humains tenant compte de l'action. Nous montrons des améliorations significatives par rapport aux méthodes existantes grâce à notre nouvelle formulation VAE basée sur un Transformer. Ce modèle code et décode des séquences de pose du corps humain paramétrisées par le modèle SMPL, en utilisant un vecteur de mouvement latent global.

Dans notre deuxième contribution, décrite dans le Chapitre 4, nous allons au-delà des actions catégorielles et nous nous intéressons à la synthèse de divers mouvements humains 3D à partir de *descriptions textuelles*. Cela permet d'élargir le vocabulaire et d'obtenir un contrôle potentiellement plus fin. Notre travail se distingue des recherches précédentes en ne générant pas de manière déterministe une séquence de mouvement unique, mais en synthétisant des séquences multiples et variées à partir d'un texte donné. Nous proposons `TEMOS`, qui repose sur notre architecture `ACTOR` basée sur un VAE, mais qui intègre cette fois un encodeur de texte pré-entraîné pour traiter les entrées en langage naturel à large vocabulaire.

Dans notre troisième contribution, décrite dans le Chapitre 5, nous abordons la tâche adjacente de la *recherche* de mouvements humains 3D à partir de texte, où l'objectif est, par le biais d'une requête textuelle, de rechercher à l'intérieur d'une collection de mouvements. Nous présentons une approche simple et efficace, appelée `TMR`, qui s'appuie sur notre modèle précédent `TEMOS`, en intégrant une fonction de coût contrastive pour améliorer la structure de l'espace latent multimodal. Nos résultats soulignent l'importance de conserver la génération de mouvement avec l'entraînement contrastif pour améliorer les résultats. Nous établissons un nouveau critère d'évaluation et effectuons des analyses sur plusieurs protocoles.

Dans notre quatrième contribution, décrite dans le Chapitre 6, nous présentons un nouveau problème appelé "contrôle par chronologie multi-pistes" pour la synthèse de mouvements humains 3D pilotée par le texte. Au lieu d'une seule description textuelle, les utilisateurs organisent plusieurs textes dans des intervalles temporels qui peuvent se chevaucher. Nous présentons `STMC`, une méthode

de débruitage en temps de test pouvant être intégrée à n'importe quel modèle de diffusion de mouvement humain pré-entraîné. Nos évaluations démontrent que notre méthode génère des mouvements qui correspondent étroitement aux aspects sémantiques et temporels de la chronologie d'entrée.

En résumé, les contributions de cette thèse sont les suivantes : (i) nous développons un autoencodeur variationnel génératif, `ACTOR`, pour la génération de séquences de mouvements humains conditionnée par l'action, (ii) nous présentons `TEMOS`, un modèle génératif conditionné par le texte qui synthétise des mouvements humains diversifiés, (iii) nous présentons `TMR`, une nouvelle approche pour la recherche de mouvements humains 3D à partir de texte, (iv) enfin, nous proposons `STMC`, une méthode pour la génération de mouvements humains controlés par une chronologie à plusieurs pistes.

Acknowledgements

I would like to express my deepest gratitude to my advisors, Gül Varol and Michael J. Black. I feel incredibly lucky to have benefited from their support and advice throughout my PhD journey.

Gül, your mentorship has been exceptional. I am very grateful for all the efforts you made in order to make me grow as a researcher. Your perfectionism and enthusiasm were a great inspiration and enriching me.

Michael, it has been a true privilege to collaborate with you. You taught me the importance of stepping back and critically evaluating my work. You have played a key role in shaping my approach to research.

Mathieu, thank you for getting me interested in computer vision research and for putting me in touch with Gül. It was also a real pleasure to take part in your team's research discussions.

I am deeply grateful to Francesc Moreno-Noguer, Kris Kitani, Siyu Tang, and Edmond Boyer for agreeing to be part of my thesis jury. Special thanks to Francesc and Kris for their insightful comments and feedback.

During my PhD, I had the privilege of being part of two prestigious institutions: the Imagine research group at ENPC and the Perceiving Systems department of the Max Planck Institute for Intelligent Systems. I would like to thank all my colleagues and the permanent researchers at ENPC for the enriching discussions in the office. I am equally grateful to all my colleagues at MPI for their support and collaboration.

I was blessed to be an intern at NVIDIA during my PhD. I sincerely thank the members of my team for their support and collaboration. I am excited and honoured to be back as a full-time member of the team.

I want to express my gratitude to Makoto Yamada. You have been an exceptional mentor during the beginning of my journey of researcher and provided invaluable guidance.

I wish to thank all my teachers at ENS who introduced me to the world of research and have played a significant role in shaping my academic path. I also really enjoyed the research internships I did during ENS at various research institutions. With Wolf Hauser at Dxo, Martial Hebert at CMU and Alexis Nasr at LIF.

My deepest thanks to my prépa teachers, who cultivated my love for mathematics, computer science and science in general.

I would like to express a special thank to my close friends. You are the best and you know it.

To my family, thank you for your unconditional love and support. I don't have enough words to tell you how grateful I am.

To my wife, Margaux. A whole page would not be enough to thank you. Your endless support, patience and love have been my anchor throughout this journey. Thank you for being by my side through every challenge and every victory.

Contents

Chapter 1

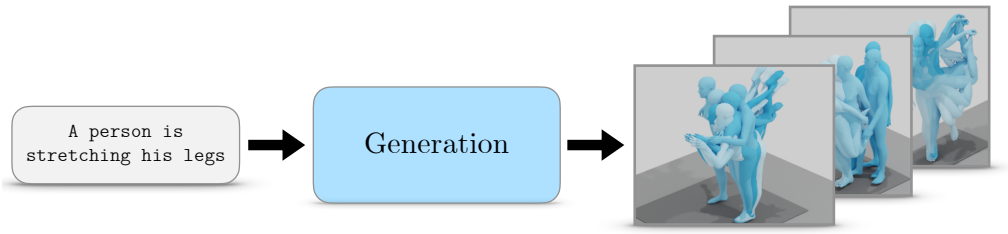
Introduction

This introductory chapter describes the goal (Section 1.1), motivations (Section 1.2), challenges (Section 1.3), contributions (Section 1.4) and the outline (Section 1.5) of the thesis.

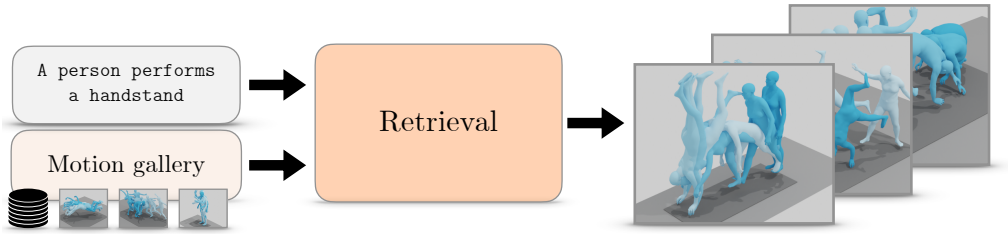
1.1 Goals

In this thesis, we are mainly interested in two tasks: to control the synthesis of new human motions and to retrieve pre-existing motion capture data.

3D human motion synthesis involves generating sequences of 3D human poses that represent the articulated human body movements, as shown in Figure 1.1a. Our aim is to explore the control of the generation process using semantic inputs, including categorical actions, free-form text, or more structured inputs (see Figure 1.2). Therefore, we develop models capable of being conditioned on such input signals to produce the desired human motions, using pretrained language models. Although 3D human motion generation is not new (see an early example in Figure 1.3), part of its complexity comes from the human sensitivity to realism. In this thesis, we build models capable of generating realistic human motions by producing meshes, thanks to the SMPL [Loper et al. 2015] body model (see Section 2.4.2 for more details). Finally, our goal is to make the motion generation process probabilistic, so that

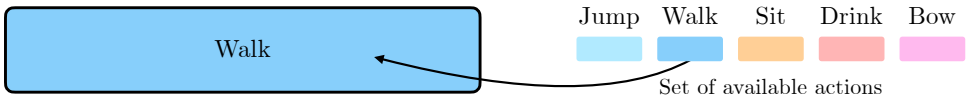


(a) 3D human motion generation from semantic input
(Chapters 3, 4, 6)

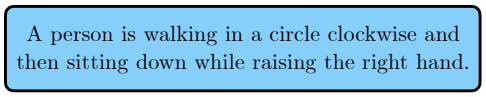


(b) Text-to-motion retrieval
(Chapter 5)

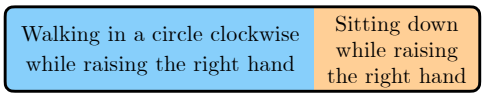
Figure 1.1: **Goal:** In this thesis, we solve two main goals (a) generating human motion from semantic inputs (e.g., action, text, timeline etc, see Figure 1.2) (b) retrieving a motion from a gallery of motions and a textual query.



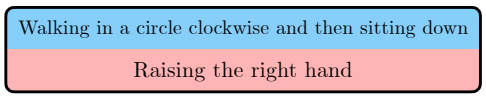
(a) Action label
(Chapter 3)



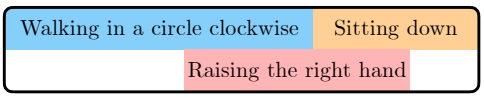
(b) Free-form text description
(Chapter 4)



(c) Temporal composition
(Annex A)



(d) Spatial composition
(Annex B)



(e) Multi-track timeline control
(Chapter 6)

Figure 1.2: **Semantic inputs:** We present several semantic inputs for generating 3D human motions. We show in (a) an action picked from a set of pre-defined actions, in (b) a free-form text description, in (c) a sequence of text prompts, in (d) a set of text prompts and in (e) a multi-track timeline which consists of precise time intervals with text descriptions. In this thesis and Annex, we will present methods for generating motions from all these inputs.

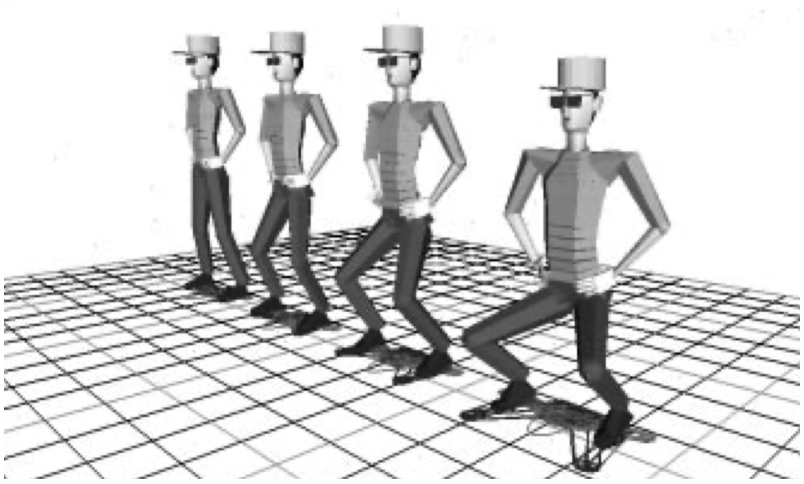


Figure 1.3: **Behavioral control in animation systems:** The field of 3D human motion synthesis has been explored extensively over the years. This figure illustrates the *Jack* animation system, as introduced by [Badler et al. 1993], and shows the lowering of the virtual character's centre of mass.

diverse motions can be synthesized from the same input. We design models we can sample from, such as variational autoencoders and diffusion (presented in Section 2.1).

3D human motion retrieval focuses on searching for the most relevant 3D human motions from a gallery, based on a natural language query that specifies the desired motion (as illustrated in Figure 1.1b). The key advantage of retrieval over generation lies in the guarantee of obtaining motions that are both physically plausible and realistic because the search is based on real motion capture data. We investigate the creation of a cross-modal latent space between text and motion. As such, we leverage pretrained language models, contrastive training and generation losses which structure the latent space well.

1.2 Motivations

Human motions, with their multitude and ambiguity, are highly complex and can be subtle. Modeling a rich semantic space that represents this variety of human motions is therefore a difficult and interesting problem. Generating human motions from natural language description is a way to teach computers the complex language of human motions. Also, the synthesized motions could

be used to meet the growing demand for high-quality 3D human motion data across a range of sectors. According to a report by the Market Research Future [1], the market for motion capture data is expected to grow by a substantial 230% between 2021 and 2030. In the following, we first explain motion capture usage and cost, and then we present other applications of human motion generation and text-to-motion retrieval.

Motion capture technology plays a central role in creating human-centric special effects, serving as the foundation for realistic character movements in movies and video games. As illustrated in Figure 1.4, this technology is essential for bringing to life the dynamic actions seen on screen. In the video games sector, motion capture enables the design of a predefined set of motions — such as kicking, walking, and running — that are precisely captured and then mapped to gameplay controls. This allows players to execute complex movements at the press of a button, enhancing the interactive experience. Similarly, in virtual reality (VR) environments, there is a growing interest in populating these digital worlds with realistically moving humans. Since traditional methods of motion capture are expensive and time-consuming, cost-effective alternatives are highly advantageous. Human motion generation offers the ability to create new motions on demand, while retrieval methods allow motions to be indexed via text queries among existing large motion collections.

Motion assets for synthetic training data. The use of synthetic data in training data-hungry algorithms, such as deep neural networks, has become increasingly important, especially in scenarios where acquiring labeled training data is cumbersome. In the context of human motions, prior research has demonstrated the potential of generating synthetic videos using 3D motion assets [Varol et al. 2017; Cai et al. 2021; Varol et al. 2021; Black et al. 2023; Yang et al. 2023] to train models for a variety of tasks. Specifically, work in this area has focused on applications such as body part segmentation, depth estimation, action recognition and human pose estimation, leveraging synthetic datasets to overcome the limitations associated with the scarcity of labeled real-world data. Synthetic data generation is generally based on real motion capture. The transition to generated motion holds great promise for expanding

[1] https://www.marketresearchfuture.com/reports/3d-motion-capture-system-market-3026

(a) Movies (b) Video games

Figure 1.4: **Motion capture** has revolutionised the way we create and interact with digital characters, and it is essential to many industries. It allows filmmakers to translate human performances into lifelike animations for non-human creatures, as illustrated by the impressive transformation of actors into apes in "Planet of the Apes" (a). Similarly, motion capture is essential for creating deeply immersive experiences in video games. "Detroit: Become Human" showcases this by offering players the ability to control characters that move and express emotions like humans (b).

the diversity and range of motion assets available for synthetic data creation. This is particularly relevant for semantic tasks, such as action recognition or text-to-video retrieval, where it is necessary to accurately associate semantic labels with synthetic videos. An innovative approach in this context is the work presented in [Cascante-Bonilla et al. 2023], which explores the use of `TEACH` (Annex A) and the finetuning of vision-language models (VLMs) such as CLIP [Radford et al. 2021] on synthetic data. By automatically associating each video with a descriptive text, this methodology brings opportunities to improve synthetic training datasets in complex semantic tasks.

Application of retrieval. Text-to-motion retrieval ability offers a wide range of applications. It enables the automatic indexing of large collections of motion capture data, facilitating searches of these databases using natural language descriptions facilitating text-based searches in such motion databases (i.e., a search engine with a natural language interface). In addition, animators can use the retrieved motions as a basis or initialisation for creating new motion sequences. This approach simplifies the animation process by providing a starting point that is already close to the desired result, reducing the time and effort required to generate new animations from scratch. When used in the reverse direction of motion-to-text retrieval, it has the potential to simplify the labour-intensive text labeling process of newly captured motion data, by associating each motion with the closest text description. Furthermore, also

highlighted in [Guo et al. 2022a], a retrieval model can be used to assess the quality of human motion generation methods by checking whether a correct text label is successfully retrieved when queried with a *generated* motion.

Robotics and healthcare. While there are other challenges such as navigation and physically plausible generation, the capacity to control the generation of human motions offers substantial potential in the field of robotics. Imagine a scenario where humanoid robots can be controlled through voice commands. This advancement could have applications in the medical field, where robots could assist surgical procedures, enhancing the efficiency and precision of operations. Additionally, domestic robots guided by verbal instructions could simplify household tasks, greatly easing daily routines.

1.3 Challenges

The problems of generating 3D human motions from semantic signals and text-to-motion retrieval are very challenging. We provide an overview of the main technical difficulties below.

Temporal modeling. How to design neural network architectures that best model the temporal aspect of motions is an open problem. RNNs models with frame-level embeddings tend to jitter and drift. One contribution of this thesis is to propose a sequence-level embedding formulation with Transformers (Chapter 3). The advantage of sequence-level generations over autoregressive ones is that the generated motions are more coherent and smooth. However, this comes with the disadvantage that extending non-autoregressive models over long generations is not straightforward due to quadratic complexity.

Realistic motions. For human motion generations to be realistic, each body pose (within the temporal sequence) must be independently plausible, the overall motion must be consistent over time and appear natural to the human eye. One challenge is to ensure that the movements generated fall within the subspace of realistic movements. Some examples of unrealistic motion generations contain drifts, static poses, self-penetrations, or physically unplausible movements such as foot sliding. For tasks involving generation from

a sequence of text (Annex A) or a multi-track timeline (Chapter 6), maintaining continuity and smooth transitions between actions is also crucial for realism.

Diversity of generations. An ideal generative model should create multiple sequences that adhere to the semantic input while introducing natural variations. For instance, the action "kicking" can involve either foot (left or right) and can vary in height of the leg. The main challenge is to capture the degrees of freedom allowed by the specified action and to randomize the action-irrelevant body parts while still producing realistic motions. We address this challenge in this thesis by modeling the distribution of motions given a semantic input. Specifically, we use a Variational Autoencoder (VAE) approach in Chapter 3 (for actions) and in Chapter 4 (for texts), and a diffusion model in Chapter 6.

Limitations of training data. Annotated human motion datasets are rare and often contain limited data (less than 15,000 sequences). Training data-driven models with such low-data regime is known to be a challenge, leading to overfitting and lack of generalization. Moreover, typical motion datasets are dominated by common locomotive motions like "walking", while actions like "applauding" and "playing bowling" are underrepresented. Motion clips often do not align perfectly with the text descriptions, and existing datasets may lead to learning spurious correlations (see Annex B). For example, waving the arm is always associated with standing, without taking into account other possibilities such as sitting. In addition, the training data lacks complex compositions. To address these limitations, one approach we use, detailed in Annex B, involves creating synthetic data by blending different motions together by body parts.

Difficulty of evaluation. A well-known challenge in generative modeling is the difficulty of evaluation. Traditional evaluation metrics are often based on the comparison of joint positions, assessing how closely the generated motion follows the ground truth. More recently, retrieval-based metrics have been introduced, which involve using a pretrained model to perform retrieval tasks or compute measures like the Fréchet Inception Distance (FID). However, these metrics might not fully represent the perceptual quality of the motions. Therefore, conducting perceptual studies (as in Chapter 4, 6) is essential to evaluate the perceived quality of motions. In addition, in Chapter 5, we introduce a new evaluation method based on embedding similarity, which we use in Chapter 6.

1.4 Contributions

The following sections itemize the publications contributions, as well as the code that has been open-sourced during this thesis.

1.4.1 Publications

The work done during this PhD led to the following publications:

- Mathis Petrovich, Michael J. Black, Gül Varol "Action-Conditioned 3D Human Motion Synthesis with Transformer VAE"
  ICCV 2021 (Chapter 3: `ACTOR`).
- Mathis Petrovich, Michael J. Black, Gül Varol "TEMOS: Generating Diverse Human Motions from Textual Descriptions"
  ECCV (Oral) 2022 (Chapter 4: `TEMOS`).
- Mathis Petrovich, Michael J. Black, Gül Varol "TMR: Text-to-Motion Retrieval Using Contrastive 3D Human Motion Synthesis"
  ICCV 2023 (Chapter 5: `TMR`).
- Mathis Petrovich, Or Litany, Umar Iqbal, Michael J. Black, Gül Varol, Xue Bin Peng, Davis Rempe, "Multi-Track Timeline Control for Text-Driven 3D Human Motion Generation"
  CVPRW 2024 (Chapter 6: `STMC`).

I also played a major role in these other publications, which are included in the Annex to this manuscript:

- Nikos Athanasiou, Mathis Petrovich, Michael J. Black, Gül Varol "TEACH: Temporal Action Compositions for 3D Humans"
  3DV 2022 (Annex A: `TEACH`).
- Nikos Athanasiou*, Mathis Petrovich*, Michael J. Black, Gül Varol "SINC: Spatial Composition of 3D Human Motions for Simultaneous Action Generation" (*equal contribution)
  ICCV 2023 (Annex B: `SINC`).

1.4.2 Software contributions

The code for all the publications has been released on GitHub, along with the pre-trained models. Together, the projects currently have over a thousand stars on GitHub. We provide more information on each of them below, as well as links to the project websites where code, pre-trained models, videos and a demo (for some projects) can be found.

`ACTOR`. The code and pre-trained models for generating 3D human motions from an action category are released as part of the project presented in [Petrovich et al. 2021] (Chapter 3). Website: `https://mathis.petrovich.fr/actor`.

`TEMOS`. The code and pre-trained models for generating 3D human motions from a text description are released as part of the project presented in [Petrovich et al. 2022] (Chapter 4). Website: `https://mathis.petrovich.fr/temos`.

`TMR`. The code, the demo and pre-trained models for text-to-motion retrieval are released as part of the project presented in [Petrovich et al. 2023] (Chapter 5). Website: `https://mathis.petrovich.fr/tmr`.

`STMC`. The code and pre-trained models for generating 3D human motions from a multi-track timeline are released as part of the project presented in [Petrovich et al. 2024] (Chapter 6). Website: `https://mathis.petrovich.fr/stmc`.

The websites for the other two contributions are shown below.

`TEACH`. The code and pre-trained models for generating 3D human motions from a sequence of text description are released as part of the project presented in [Athanasiou et al. 2022] (Annex A). Website: `https://teach.is.tue.mpg.de`.

`SINC`. The code and pre-trained models for generating 3D human motions from a set of text description are released as part of the project presented in [Athanasiou et al. 2023] (Annex B). Website: `https://sinc.is.tue.mpg.de`.

1.5 Outline

Including this introduction, this thesis is organized into seven chapters. An Annex is also provided.

In Chapter 2 (background), we begin by presenting the generative models relevant to human motion synthesis. We then review the literature of human motion generation and text-to-motion retrieval. Finally, we present the different human body and motion representations and the motion datasets.

In Chapter 3 (`ACTOR`), we focus on the generation of realistic and diverse human motion sequences conditioned on specific action categories. We present a new VAE approach based on a conditional Transformer, designed and trained to generate action-conditioned human motions using the SMPL parametrization. One key novelty is that the latent space represents an entire motion within a single embedding, and a sequence-level embedding vector can be sampled from this space to non-autoregressively generate motions. To overcome the lack of training data, we use human motion estimation from monocular videos, and show that it is possible to learn our generator from these noisy estimates. We present an in-depth study of architecture ablation and loss components and show state-of-the-art results on several datasets. In addition, we highlight practical applications of our model in two areas: improving action recognition and denoising motion capture data.

In Chapter 4 (`TEMOS`), we address the problem of generating diverse 3D human motions from textual descriptions. We take inspiration from Chapter 3 and prior work on Language2Pose [Ahuja et al. 2019] and build a Transformer-based cross-modal variational approach. Instead of categorical action conditioning, we employ a text encoder to represent the free-form language condition. By using embedding losses, we ensure that the latent space is consistent between motion and text modalities. We provide a comprehensive ablation study of the model's components and show better performance than the state of the art by a large margin, on both qualitative metrics and perceptual studies. Besides our competitive performance, our model is able to generate multiple different motions per input text description, in contrast to previous deterministic works. We show that our model is compatible with different motion representations,

based on joint locations and with the parametric SMPL body model.

In Chapter 5 (TMR), we learn a model for text to 3D human motion retrieval. The previous Chapter 4 has already introduced a cross-modal latent space; we build on this model by incorporating a contrastive loss to further structure the latent space. We show that keeping the generation loss is crucial for obtaining good performance. Furthermore, given the similarity of text descriptions across motions in the dataset, we show that a simple negative filtering strategy can improve the model performance. As the retrieval task is not extensively studied in the literature, we introduce a series of benchmarks of varying difficulty. We provide extensive experiments to analyze the effects of each component in controlled environments, and show state-of-the-art performance. This high-performance retrieval model can now be used as an evaluator of 3D human motion generation methods and will be used to evaluate the method of Chapter 6.

In Chapter 6 (STMC), we design a new task of multi-track timeline control for text-driven 3D human motion generation. Instead of a single prompt, the timeline input gives users fine-grained control over the timing and duration of actions, while still maintaining the simplicity of natural language. To solve this new problem, we design a test-time denoising process that enables pre-trained diffusion models to handle the spatial and temporal compositions present in the timeline inputs (see Annex A, B). We show qualitatively, and with user studies, that our method works better than carefully crafted baselines. Also, we train a diffusion-based model to generate SMPL pose parameters directly to avoid the need for test-time optimization that traditional methods use.

In Chapter 7 (discussion), we present a summary of the contributions made in this work. We conclude by outlining limitations and suggesting potential avenues for future research.

In Annex, we include our work on generating motions from temporal compositions (TEACH in Annex A) and from spatial compositions (SINC in Annex B). Both these works build on (Chapter 4), adapting our individual text-to-motion TEMOS model to input a series or a set of textual descriptions, for sequential or simultaneous action generation, respectively.

Chapter 2

Background

In this chapter, we start by briefly reviewing some of the classical generative modeling approaches (Section 2.1). We will then present the literature on 3D human motion generation (Section 2.2) and text-to-motion retrieval (Section 2.3). Finally, we will discuss the different ways of representing 3D human bodies and motions (Section 2.4) and present 3D human motion datasets (Section 2.5).

2.1 Generative models

Generative modeling has recently attracted considerable interest in the research community, particularly in text-to-image synthesis [Ramesh et al. 2022; Rombach et al. 2022; Saharia et al. 2022], natural language processing [OpenAI 2022; Bard 2023] and human motion generation [Petrovich et al. 2021; Tevet et al. 2023; Xu et al. 2023]. The common objective is to synthesize new samples by learning the underlying distribution of the training data. In this section, we first present variational autoencoders (VAEs) and diffusion models – used throughout this thesis – and then we briefly introduce other popular generative models: generative adversarial networks (GANs) and normalizing flows. We show in Figure 2.1 an overview of the different methods and briefly explain the basic idea behind each method next.

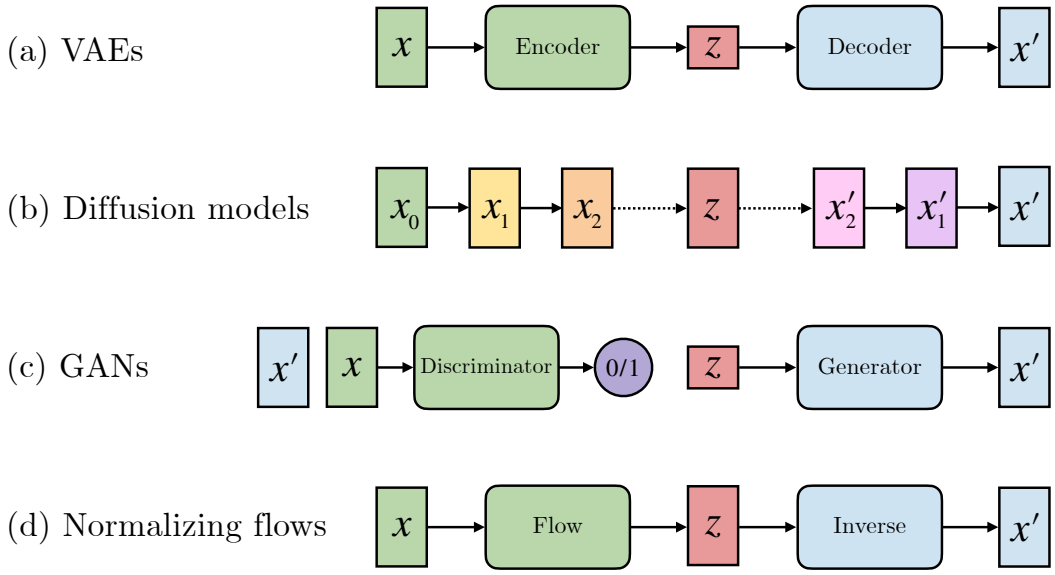


Figure 2.1: **Generative models:** We illustrate different generative models: (a) variational autoencoders (VAEs), (b) diffusion models, (c) generative adversarial networks (GANs) and (d) normalizing flows. x, in green, corresponds to samples from real data used during training. z, in red, corresponds to the latent vector: it is randomly sampled during generation and can be typically of a smaller dimension than x for (a) and (c). x', in blue, corresponds to the generated data. The figure was adapted from the blog post of [Weng 2021].[1]

Variational autoencoders (VAEs) [**Kingma et al. 2014**] are built with two neural networks: an encoder and a decoder. The encoder maps the input data into a probabilistic distribution (often Gaussian) from which we can sample a latent vector (usually of a smaller dimension). The decoder then aims to reconstruct the original input from the latent vector. VAEs are trained to minimize a weighted sum of two loss terms. The first is the reconstruction loss, which ensures that the decoder output is close to the input data. The second loss is the Kullback–Leibler (KL) divergence between the encoded distribution and a prior distribution (often a Gaussian distribution with zero mean and identity variance). This second term aims to regularize the latent space and is the key innovation in VAEs. This approach allows complex distributions to be approximated efficiently and guarantees a smooth and continuous latent space, which is essential for generating realistic and diverse samples.

Conditional variational autoencoders (CVAEs) [Sohn et al. 2015] extend VAEs by taking an additional condition as input (such as an action label in Chapter 3). The condition becomes an additional input for both the encoder and decoder

[1] https://lilianweng.github.io/posts/2021-07-11-diffusion-models

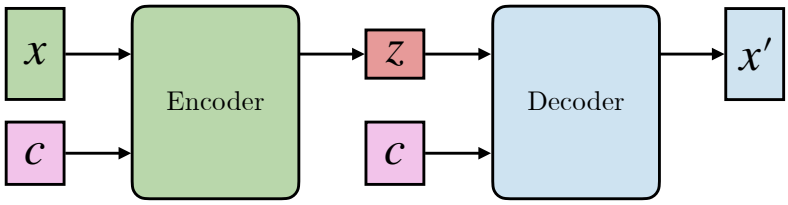


Figure 2.2: **Conditional variational autoencoders (CVAEs) [Sohn et al. 2015]:** The overall structure is similar to VAEs (see Figure 2.1) but the condition c is added as input to both the encoder and the decoder.

(see Figure 2.2). In our work, we instantiate encoder and decoder components with the Transformer architecture [Vaswani et al. 2017].

Diffusion models [**Sohl-Dickstein et al. 2015**; **Ho et al. 2020**] consist of a forward process and a reverse process. The forward process consists of gradually noising the input data for a large number of steps N, so that it could have been randomly sampled from a Gaussian distribution. In contrast, the reverse process is modeled by a neural network and is learned to denoise data. At test time, we generate new data by sampling noise from a Gaussian distribution and denoise it N times via the neural network. The denoising model takes as input the data to be denoised and the current time step and outputs the denoised data one step ahead. We can make the model conditional by adding an extra condition as input to the denoiser and, potentially, increase the adherence to the condition using techniques such as classifier guidance [Dhariwal et al. 2021] or classifier-free guidance [Ho et al. 2021]. Diffusion models are capable of generating high-quality samples with a stable training procedure, and can capture complex data distribution directly in the data space, or in a latent space [Rombach et al. 2022].

Generative adversarial networks (GANs) [**Goodfellow et al. 2014**] are built using two competing neural networks: a generator and a discriminator. The generator must generate data samples that closely resemble the distribution of the training data. On the other side, the discriminator checks these samples closely, identifying between artificially generated data from real data. Both networks are constantly improving as a direct consequence of this competition. As a result, the generator becomes increasingly capable of producing realistic samples, and the discriminator becomes more effective at spotting fakes. The main challenges of GANs are training stability during learning and mode

collapse, where the model generates limited varieties of outputs.

Normalizing flows [**Rezende et al. 2015**; **Dinh et al. 2017**] are a class of deep generative models that transform a simple distribution into a more complex one through a sequence of invertible mappings. The change of variables theorem guides this process, allowing the model to explicitly learn the distribution of the data. Normalizing flows, in contrast to GANs and VAEs, compute exact log-likelihoods, which makes them very useful for evaluation and interpretability. A fundamental component of their design is that each transformation must be invertible and have a computationally efficient Jacobian determinant. See Lilian Weng's post [Weng 2018] on flow-based deep generative models for an in-depth overview.

2.2 3D human motion generation

Research on human motion analysis has a long history dating back to 1980s [Futrelle et al. 1978; O'Rourke et al. 1980; Badler et al. 1993; Gavrila 1999] (see Figure 1.3 for an illustration from one of the earliest works). Recently, a large body of work in both vision and graphics has been dedicated to generating 3D human motions [Starke et al. 2019; Petrovich et al. 2021; Guo et al. 2022b; Li et al. 2022b; Zhang et al. 2022a; Athanasiou et al. 2023; Chen et al. 2023; Kulkarni et al. 2023; Tevet et al. 2023; Zhu et al. 2023b].

Some works focus on **unconditional generation**, where the goal is to synthesize motions in an unconstrained manner (i.e., without any start pose or a control input) and where a distribution covering the space of possible human motions are modeled [Yan et al. 2019; Zhang et al. 2020b; Zhao et al. 2020]. Given a random noise only, these models aim to generate a realistic and diverse sequence of human motions. Early work used PCA [Ormoneit et al. 2005] and GPLVMs [Urtasun et al. 2007] to learn statistical models of cyclic motions like walking and running. [Yan et al. 2019] presents a convolution-based generative model for realistic, unconstrained motions. Similarly, [Zhang et al. 2020b] synthesizes arbitrary sequences, focusing on unbounded motions in time. [Zhao et al. 2020] introduces a Bayesian approach, where Hidden semi-Markov Mod-

els are used for jointly training generative and discriminative models. More recently, some work focus particularly on generating long sequences [Habibie et al. 2017; Li et al. 2022b].

Unconstrained motion synthesis methods lack the ability to control the generation process. The goal of this thesis is to inject semantic control within motion generation. To this end, we focus the following sections on conditional motion generation. Specifically, we review prior work on motion synthesis, conditioned on an action label (Section 2.2.1), on a text description (Section 2.2.2) and on other types of conditions (Section 2.2.3).

2.2.1 Action-conditioned

An action label is defined as a symbolic category from a list of pre-defined actions such as ["walk", "jump", "sit"] (see Figure 1.2a). That is, action-conditioned motion generation methods do not treat the actions as textual data, but instead as categorical data. The goal of action-conditioned synthesis is to generate motions belonging to the specified action, allowing a form of semantic control.

Phase-functioned neural networks [Holden et al. 2017] and neural state machines [Starke et al. 2019] were introduced in the graphics literature. Both exploit the notion of actions being driven by the phase of a sinusoidal function. This is related to the idea of positional encoding (commonly used in transformer architectures); however, these methods require manual labor to segment actions and build these phase functions. In contrast, our action-conditioned motion synthesis method `ACTOR` [Petrovich et al. 2021], described in Chapter 3, does not require such phase segments. Most similar to `ACTOR` is Action2Motion [Guo et al. 2020], a *per-frame* VAE conditioned on actions, using a GRU-based architecture. In `ACTOR`, our *sequence-level* VAE latent space, in conjunction with the Transformer-based design provides significant advantages over Action2Motion both qualitative and quantitatively. PoseGPT [Lucas et al. 2022] proposes a method to output distributions on possible futures, with or without conditioning on past motion, using a quantization-based approach. Other recent works [Henter et al. 2020; Zanfir et al. 2020] use normalizing flows to

address human motion estimation and generation problems. INR [Cervantes et al. 2022] leverages variational implicit neural representations to obtain better variable-length sequence generation.

The main problem with action-conditioning is that it is limited to a predefined set of categories. The community has therefore shifted towards text conditioning, which is presented in the next section.

2.2.2 Text-conditioned

A **textual description** represents a written natural language sentence (e.g., in English) that describes what and how a human motion is performed. Unlike action labels, where the number of actions is fixed, text is free-form. The sentence can include various levels of detail: a precise sequence of actions such as *"A human slowly walks two steps with their arms crossed and then stops"* or a more ambiguous description such as *"A man walks"*. The data structure is a sequence of words $W_{1:N} = W_1, \ldots, W_N$ from the English vocabulary.

Recent work explores the advances in natural language modeling [Mikolov et al. 2013; Devlin et al. 2019] to design sequence-to-sequence approaches to cast the text-to-motion task as a machine translation problem [Ahn et al. 2018; Lin et al. 2018a; Plappert et al. 2018]. Others build joint cross-modal embeddings to map the text and motion to the same space [Yamada et al. 2018; Ahuja et al. 2019; Ghosh et al. 2021], which has been a success in other research areas such as text and images [Radford et al. 2021; Yuan et al. 2021; Yang et al. 2022].

In terms of the language encoding, [Ahuja et al. 2019] employs word2vec text embeddings [Mikolov et al. 2013], while [Ghosh et al. 2021] uses the more recent BERT model [Devlin et al. 2019]. In Chapter 4, we use the DistilBERT [Sanh et al. 2019] model while other recent works use the sentence-level CLIP [Radford et al. 2021] text embeddings [Tevet et al. 2022; Tevet et al. 2023].

In terms of body motion representations, several methods use an impoverished body motion representation. For example, some do not model the global trajectory [Plappert et al. 2018; Yamada et al. 2018], making the motions unrealistic and ignoring the global movement description in the input text.

Most other work uses 3D motion capture data [Lin et al. 2018a; Lin et al. 2018b; Ahuja et al. 2019; Ghosh et al. 2021]. [Lin et al. 2018a] proposes an RNN to map descriptions into a latent-space motion representation. DVGANs [Lin et al. 2018b] adapt the CMU MoCap database [*CMU Graphics Lab Motion Capture Database* 2003] and Human3.6M [Ionescu et al. 2011; Ionescu et al. 2014] for the task of motion generation and completion, and they use the action labels as text-conditioning instead of categorical supervision. Text2Action [Ahn et al. 2018] relies on an encoder-decoder RNN to learn the mapping between language and pose but only models the upper body motion. This is because Text2Action uses a semi-automatic approach to create training data from the MSR-VTT captioned video dataset [Xu et al. 2016], which contains frequently occluded lower bodies. They apply 2D pose estimation, lift the joints to 3D, and employ manual cleaning of the input text to make it generic. Language2Pose [Ahuja et al. 2019] instead learns a joint embedding space of sentences and poses, while [Ghosh et al. 2021] synthesize 3D skeletons using sentences from the KIT dataset, encoding the upper and lower body separately.

A key limitation of many text-conditioned motion generation models is that they are deterministic [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021]. We propose in Chapter 4 our `TEMOS` method [Petrovich et al. 2022] to generate diverse human motion from text description, following the probabilitic synthesis of Chapter 3. A concurrent work is [Guo et al. 2022a] which uses a temporal VAE along with quantization to generate diverse human motions. TM2T [Guo et al. 2022b] introduces a framework to jointly perform text-to-motion and motion-to-text, integrating a back-translation loss. Following the success of diffusion models [Sohl-Dickstein et al. 2015; Ho et al. 2020], very recently, MDM [Tevet et al. 2023], FLAME [Kim et al. 2023], MotionDiffuse [Zhang et al. 2022a], and MoFusion [Dabral et al. 2023] demonstrate diffusion-based motion synthesis conditioned on text. Recent work [Chen et al. 2023] shows the potential of latent diffusion to address the slow inference limitation and GraphMotion [Jin et al. 2023] uses hierarchical semantic graphs for fine-grained control over motion generation. A different line of work [Saunders et al. 2020; Huang et al. 2021; Saunders et al. 2021] focuses on generating sign language motion from text.

Structured inputs and compositional motions. A particular challenge for action- and text-conditioned motion generation is to synthesize compositional motions, that is, when the input text mentions multiple actions that should occur within one motion (e.g., one after the other, or at the same time). This is particularly difficult due to the lack of compositional training data.

Several studies, our work [Athanasiou et al. 2022] described in Annex A and others [Qian et al. 2023; Zhang et al. 2023b], focus on generating motions from a sequence of text prompts and durations, i.e., ***temporal* compositions**. In Annex A, we propose `TEACH`, a method which autoregressively generates one motion (per text prompt) at a time, conditioning the next motion in the sequence with the previous one. This extends `TEMOS` (Chapter 4) which is designed to handle one input text (typically describing a single action). ActionGPT [Kalakonda et al. 2023] improves `TEACH` by retraining it with text augmentations using language models. Concurrent to `TEACH`, [Mao et al. 2022] propose a method for weakly supervised action-driven motion prediction. Given a set of frames of one action, they generate the next action and the transition to it, using 20 action categories from BABEL that have clear transitions. In contrast, `TEACH` performs motion generation directly from free-form text, going beyond categorical actions. MultiAct [Lee et al. 2023] similarly aims to produce continuous transitions between generated actions. Recently, [Wang et al. 2022a] generate a sequence of actions in 3D scenes by synthesizing pose anchors that are then placed in the scene and refined by infilling. EMS [Qian et al. 2023] proposes a two-stage approach, by first generating each action separately and then merging them through a subsequent network. Diffusion models, EDGE [Tseng et al. 2023] and PriorMDM [Shafir et al. 2023], ensure consistency between adjacent motions by enforcing temporal constraints at transitions. Our approach in Chapter 6 to temporal composition is based on DiffCollage [Zhang et al. 2023b], which stitches motions (or images) together throughout the denoising process via score arithmetic at overlapping transitions.

Other work generates motions from a set of texts to be executed at the same time, i.e., ***spatial* (body-part) composition**. In this direction, MotionCLIP [Tevet et al. 2022] and MDM [Tevet et al. 2023] test the compositional capabilities of their methods, but only show preliminary analyses. MotionDiffuse [Zhang et al. 2022a] injects manually labeled body-part information

and performs noise interpolation to obtain spatial compositionality. In SINC (Annex B), we label ground truth motion capture sequences with corresponding body parts by prompting GPT-3 [Brown et al. 2020]. These labels are used to create a synthetic dataset of motions stitched together from MoCap sequences with compatible body parts, thereby improving performance of the VAE-based 3D motion generation method TEMOS (Chapter 4) for spatial composition.

In STMC (Chapter 6), we take inspiration from SINC by using body part labels to stitch motions together during test-time denoising of a text-to-motion diffusion model. We propose the new problem of ***timeline*-conditioned** generation which introduces a framework that generalizes both temporal and spatial compositions.

2.2.3 Other types of conditions

Besides action and text conditioning, the motion generation process can also be conditioned on other signals such as past frame(s), intermediate frame(s), audio, music, trajectory, objects or scenes. In the following, we review each of these conditions.

Future motion synthesis. Given past motion or an initial pose, predicting future frames has been referred as motion prediction [Bowden 2000; Galata et al. 2001; Martinez et al. 2017; Liu et al. 2022; Salzmann et al. 2022; Zhong et al. 2022]. Formally, it consists of generating $x_{t+1}, \dots, x_T$ given the past frames $x_0, \dots, x_t$. In earlier studies [Bowden 2000; Galata et al. 2001] statistical models have been employed for this task. Recently, several works show promising results following progress in generative models with neural networks, such as GANs [Goodfellow et al. 2014], VAEs [Kingma et al. 2014] or diffusion models [Sohl-Dickstein et al. 2015; Ho et al. 2020]. Examples include HP-GAN [Barsoum et al. 2018], recurrent VAE [Habibie et al. 2017] and diffusion-based models [Gu et al. 2022; Rempe et al. 2023] for future motion prediction. Most work relies on autoregressive neural models [Fragkiadaki et al. 2015; Martinez et al. 2017; Pavllo et al. 2018; Gopalakrishnan et al. 2019; Pavllo et al. 2019] and can process variable sequence lengths. The body is commonly represented as a skeleton, though recent work exploits full 3D body shape

models [Aksan et al. 2019; Zhang et al. 2021]. Similar to [Zhang et al. 2021], we also go beyond sparse joints and incorporate vertices on the body surface by outputing SMPL [Loper et al. 2015] bodies. DLow [Yuan et al. 2020] focuses on diversifying the sampling of future motions from a pretrained model, while MOJO extends this to exploit the full 3D body [Zhang et al. 2021]. [Corona et al. 2020] performs conditional future prediction using contextual cues about the object interaction. [Li et al. 2021] present a Transformer-based method for dance generation conditioned on music and past motion.

Motion completion. Instead of knowing the past and predicting the future, a more general task is motion completion. There is a related line of work on motion "in-betweening" that takes both past and future poses and "inpaints" plausible motions between them; see [Harvey et al. 2020] for more. If only random frames are missing, it is called motion in-filling. Finally, if we want to transition from one motion to another, it is called motion blending or transition generation. Initial attempts at transition generation based on keyframes relied on inverse kinematics (IK) and time constraints to maintain physically-plausible motion [Witkin et al. 1988; Rose et al. 1996]. Recent approaches learn more expressive transitions from data [Harvey et al. 2018; Zhang et al. 2018; Ruiz et al. 2019; Harvey et al. 2020; Zhou et al. 2020; Duan et al. 2021]. [Zhang et al. 2018] use an RNN to learn jumping motions of a 2D lamp, while [Ruiz et al. 2019] and [Kaufmann et al. 2020], inspired by image inpainting, tackle motion infilling with CNNs, which work well for such tasks in 2D. [Harvey et al. 2018] propose Recurrent Transition Networks, which are limited to fixed-length transitions, and later extend this to a stochastic model along with a benchmark for motion inbetweening [Harvey et al. 2020]. [Zhou et al. 2020] suggest that convolutions are better suited to the task of transition generation and, thus, propose a purely convolutional architecture with separate components for the path predictor, motion generator and discriminator. Their method can interpolate over long time periods, but the maximum interval is limited by the receptive field of the generator. [Duan et al. 2021; Duan et al. 2022] use Transformers for motion completion and tackle the tree tasks. [Tang et al. 2022] propose an online framework for real-time in-between motion generation. [Kim et al. 2022] propose the novel task of pose-conditioned in-betweening and semantic-conditioned in-betweening. Most of this prior work on infilling and

synthesis is focused on locomotion (walking, running, etc.). While such tasks are important, human behavior includes a wider range of actions and their complex combinations.

Conditioned on audio. Another interesting task is audio-conditioned motion generation. Some work tackles the problem of music-to-dance generation [Tang et al. 2018; Lee et al. 2019; Li et al. 2020; Li et al. 2021; Valle-Pérez et al. 2021; Moltisanti et al. 2022; Sun et al. 2022a] where the generated motions should be realistic, and the dance should be in rhythm with the music. Another line of work is to generate gestures given a speech signal [Ahn et al. 2018; Ginosar et al. 2019; Bhattacharya et al. 2021; Habibie et al. 2022; Alexanderson et al. 2023; Zhu et al. 2023a] or to generate facial motions [Karras et al. 2017; Cudeiro et al. 2019; Richard et al. 2021; Fan et al. 2022].

Spatial control signal. A recently studied task is spatial control signal conditioning [Habibie et al. 2017; Holden et al. 2017; Pavllo et al. 2018; Kania et al. 2021; Zhang et al. 2022c; Karunratanakul et al. 2023; Rempe et al. 2023; Xie et al. 2024]. This involves generating human motions that are conditioned to follow a trajectory, or for a part of the body to be in a given location at a given time. For example, the work of GAMMA [Zhang et al. 2022c] consists of generating perpetual motions to reach goals for populating digital environments. QuaterNet [Pavllo et al. 2018] focuses on generating locomotion actions such as walking and running given a ground trajectory and average speed. With the rise of diffusion models, GMD [Karunratanakul et al. 2023] allows spatial guidance without re-training the diffusion model (in the same spirit as in Chapter 6) while OmniControl [Xie et al. 2024] uses an approach similar to ControlNet [Zhang et al. 2023a] by finetuning a model to adjust the output. There is also a significant graphics literature on the topic, see for example, [Holden et al. 2020] on learning motion matching, [Lee et al. 2018] on character animation and MotionVAE [Ling et al. 2020] which uses deep reinforcement learning to produce goal-directed motion.

Conditioned on a scene or objects. Other work focuses on the generation of human motions in an environment [Starke et al. 2019; Hassan et al. 2021; Zhang et al. 2022b; Zhang et al. 2022c]. [Hassan et al. 2021] propose SAMP, a method for generating motions in a scene with a few object interactions.

COUCH [Zhang et al. 2022b] releases a dataset and a method to tackle the problem of sitting realistically on a chair. [Wang et al. 2022a] use the guidance of action sequences to generate scene-aware motions, while [Wang et al. 2022c] use linguistic descriptions. [Taheri et al. 2022] propose a model to generate 3D human motions for grasping objects. NIFTY [Kulkarni et al. 2023] uses object interaction fields to generate human motions interacting with objects in a scene. [Starke et al. 2020] propose a character control framework for generating motions of people playing basketball or interacting with objects.

2.3 Text-to-motion retrieval

We present an overview of the most closely related work on text-to-motion retrieval, as well as a brief discussion on cross-modal retrieval works.

Text and human motion. The research on human motion modeling has recently witnessed an increasing interest in bridging the gap between semantics and 3D human body motions, in particular for text-conditioned motion synthesis as discussed in Section 2.2.2. However, despite remarkable progress in text-to-image synthesis [Ramesh et al. 2021; Rombach et al. 2022], text-to-motion synthesis remains at a nascent stage. The realism of the synthesized motions is limited, e.g., foot sliding artifacts as discussed in Chapter 4. We turn to text-to-motion retrieval as another alternative in Chapter 5, and perhaps complementary approach to obtain motions for a given textual description. This focus is therefore different from the work on synthesis, but it can potentially be used as part of a retrieval augmentation generation (RAG) pipeline. However, we do make use of a motion synthesis branch to aid the retrieval task.

Motion retrieval is relatively less explored. As briefly mentioned in Section 5.1, motion-to-motion retrieval methods exist (e.g., motion matching [Sidenbladh et al. 2002; Büttner et al. 2015; Holden et al. 2020]). However, the *text*-to-motion retrieval task is more challenging due to being cross-modal, i.e., nearest neighbor search across text and motion modalities. Within this category, the very recent work of Guo et al. [Guo et al. 2022a] trains a retrieval model purely for evaluation purposes, and applies a margin-based contrastive

loss [Hadsell et al. 2006], using the Euclidean distance between all pairs within a batch. `TEMOS` (Chapter 4) contains a common latent space between text and motion and can therefore be used for text-to-motion retrieval. In Chapter 5, we introduce `TMR` [Petrovich et al. 2023], a dedicated model for text-motion retrieval, inspired by image-text models such as CLIP [Radford et al. 2021].

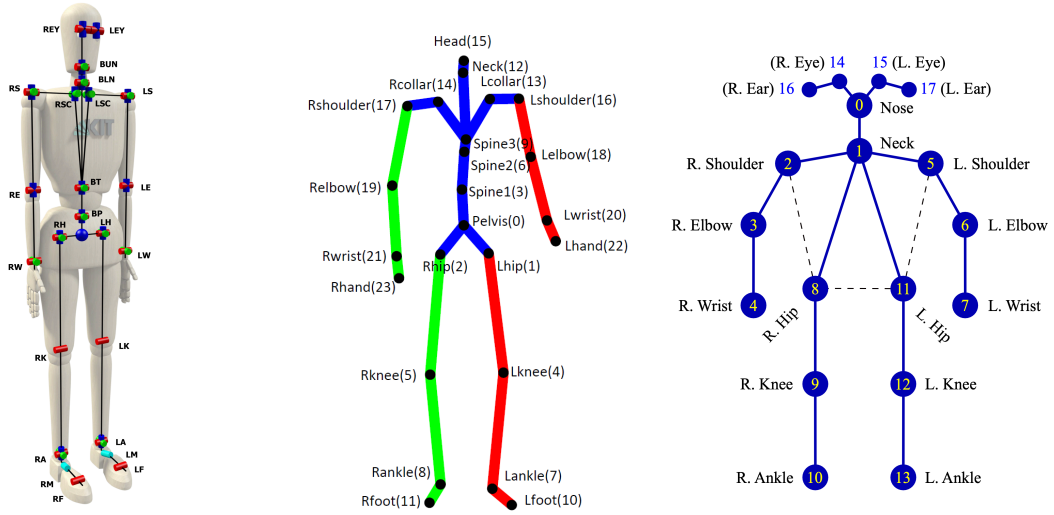
Cross-modal retrieval. Among widely adopted vision & language retrieval models, some successful examples include CLIP [Radford et al. 2021], BLIP [Li et al. 2022a], and CoCa [Yu et al. 2022] for images, and MIL-NCE [Miech et al. 2020], Frozen [Bain et al. 2021], and CLIP4Clip [Luo et al. 2022] for videos. They all use variants of cross-modal contrastive learning techniques, such as InfoNCE [Oord et al. 2018], which we also employ in Chapter 5. As discussed in Section 5.1, we draw inspiration from BLIP [Li et al. 2022a] and CoCa [Yu et al. 2022], which add synthesis branches to standard retrieval frameworks. Our approach in Chapter 5 is similar in spirit to these works in that we perform a cross-modal vision & language retrieval task, but differ in focusing on 3D human motion retrieval, which, to the best of our knowledge, has not been benchmarked.

2.4 Human body and motion representations

A **3D human motion** can be defined as a sequence of articulated human bodies. We review in this section, skeleton-based (Section 2.4.1) and parametric body representations (Section 2.4.2) — used in this thesis — but other representations such as voxels [Varol et al. 2018] or implicit representations [Rempe et al. 2021; He et al. 2022; Chen et al. 2023] have been explored in the literature in various contexts. As will be discussed, some motion representations integrate temporal information by incorporating velocities or adapting the coordinate system (such as formulating the body pose relative to the previous frame).

2.4.1 Skeleton-based representations

In the study of human motion, skeleton-based representations play a crucial role in capturing the structure and dynamics of the human body. One way is to define a kinematic tree and define a set of joints articulation. It can be determined by the joint locations of several keypoints (head, hands, knees, feet, etc), but the exact location can vary from one representation to another (see Figure 2.3).



(a) (b) (c)

Figure 2.3: **Skeleton-based:** A human body can be represented with joint positions or rotations with an underlying kinematic tree. We show several examples: (a) 21 joints defined in the MMM framework [Terlemez et al. 2014], (b) 24 joints defined in the SMPL body model [Loper et al. 2015] and (c) 18 joints used in the COCO-Pose dataset [Lin et al. 2014].

Positions. The most straight forward way to model body pose is to explicitly express the 3D joints positions (xyz) in the global coordinate system. Using joint positions in the Euclidean space is interpretable and is convenient for easy visualization.

Local rotations. Another way to model pose, which is commonly used in the animation industry, is to record the rotations of the limbs relative to their parents (local rotations). It is possible to compute a transformation from local rotations to joint positions via forward kinematics (and vice versa via inverse kinematics). The main advantage of local rotations over joint positions, is that the length of the limbs is constant and cannot vary throughout the motion.

Rotation Invariant Forward Kinematics (RIFKE). This representation was first introduced in [Holden et al. 2016]. The main observation is that the representation of a human motion should not depend on the origin rotation. This leads to the proposal of recording the 3D joint locations relative to a body-centric coordinate system. In addition, it exploits temporal information by encoding velocities instead of positions for the root joint. More precisely, for each frame, the data representation contains:

i) the **3D human joints** in a coordinate system local to the body (the definition of the 3 axes and the origin is explained in Figure 2.4,

ii) the **angles** between the local X-axis and the global X-axis (by storing them as differences between two frames),

iii) the **translation**, as the velocity of the root joint in the body's local coordinate system for X and Y axes and the position of the root joint for the Z axis.

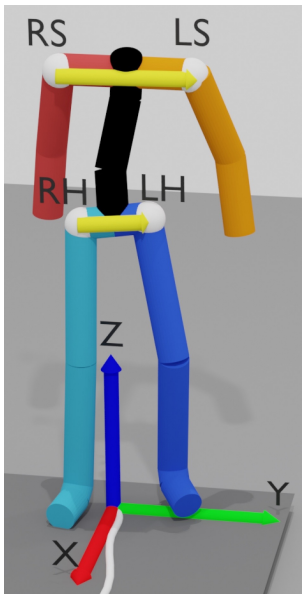
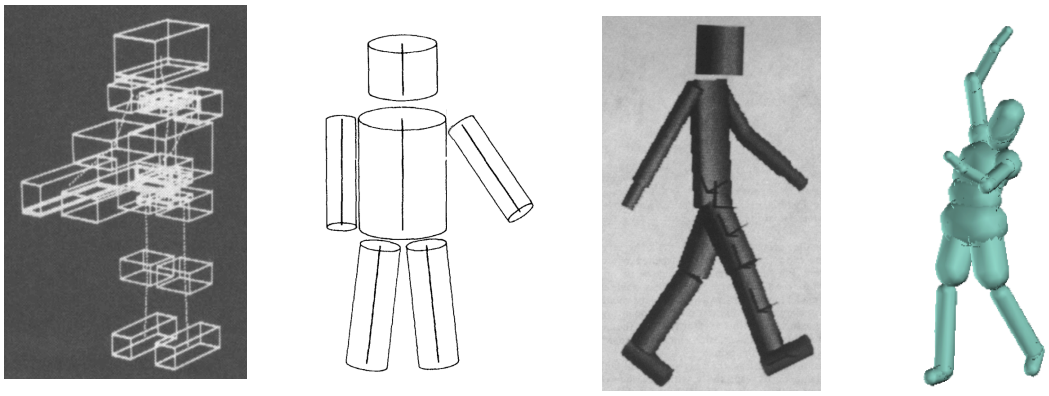


Figure 2.4: **Body's local coordinate system:** The origin is defined as the projection of the root joint into the ground. The X-axis direction is computed by taking the cross product between the average of the two yellow vectors: (left hip (LH) - right hip (RH)) and (left shoulder (LS) - right shoulder (RS)). The Z-axis (gravity axis) remains the same, and the Y-axis is the cross product of Z and X.

2.4.2 Parametric body models

Representing humans using joints is useful and simple, but lacks realism without the surface. Modeling the outside of a human body in 3D provides access to the entire volume, which can be useful for modeling contact with objects or collisions, especially when integrating it into simulation environments using physics.



(a) (b) (c) (d)

Figure 2.5: **Early surface models** represent the human body surface by (a) rectangular paralepipeds [O'Rourke et al. 1980], (b) cylinders [Marr et al. 1978], (c) elliptical cylinders [Hogg 1983] or more recently, (d) capsules [Bogo et al. 2016]. The figures are taken from the original papers.

3D primitive shapes are extensively used in early research, and can be composed with 3D polyhedra [O'Rourke et al. 1980], cylinders [Marr et al. 1978] or elliptical cylinders [Hogg 1983] which are positioned precisely in space (see Figure 2.5). Recent research uses a similar representation with capsules [Bogo et al. 2016] for its simplicity for optimization.

Skinned 3D human models. A common data structure for representing a 3D surface is a mesh, which consists of vertices and triangles. A mesh can represent humans realistically and can be easily integrated into any animation including movies or video games. Recently, many efforts have been made to develop a 3D parametric model of the human body [Anguelov et al. 2005; Loper et al. 2015; Romero et al. 2017; Pavlakos et al. 2019; Osman et al. 2020; Xu et al. 2020; Osman et al. 2022]. These models take a human pose as input (joint rotations) and a shape vector (which represents the morphology of a body) and output a human mesh with a fixed topology (see Figure 2.6a).

SCAPE [Anguelov et al. 2005] constructs a human body model learned from data, which covers variation in subject's pose and shape using triangular deformations. SMPL [Loper et al. 2015] instead uses vertex deformation with linear blend skining, is compatible with existing animation software, and is learned from thousands of 3D body scans. SMPL has become a reference model and is used in many works (including all the chapters in this thesis). More recently, several models have followed (see Figure 2.6b,c for visual examples): SMPL-H [Romero et al. 2017] integrates the MANO [Romero et al. 2017] hand model, SMPL-X [Pavlakos et al. 2019] adds facial expression, STAR [Osman et al. 2020] learns sparse spatially local corrective blend shapes and is more expressive than SMPL, and finally SUPR [Osman et al. 2022] integrates feet through a part-based human representation. GHUM&GHUML [Xu et al. 2020] models the body, hands and face and trains all model parameters based on variational auto-encoders. Other interesting research make it possible to regress the bio-mechanical skeleton from the SMPL meshes [Keller et al. 2022; Keller et al. 2023], or create a body model for infants [Hesse et al. 2018].

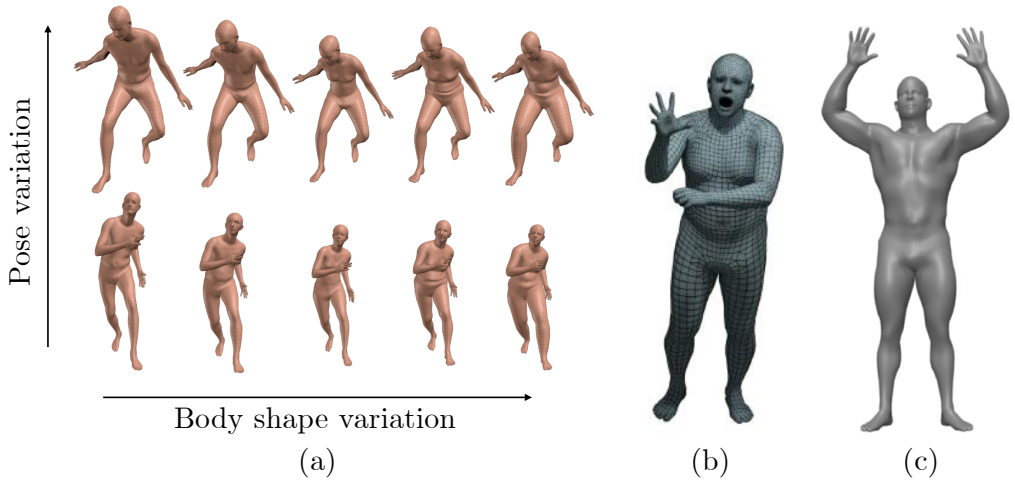


Figure 2.6: **Skinned 3D Human Body Models** can represent a variety of poses and shapes variations (SMPL [Loper et al. 2015] in (a)). A lot of effort have been made to increase the realism: (b) SMPL-X [Pavlakos et al. 2019] adds hands and face, (c) STAR [Osman et al. 2020] generalize better to new body shapes.

2.5 Motion datasets

In this section, we focus on 3D human motion datasets. Section 2.5.1 outlines widely used data sources for 3D motions. Section 2.5.2 presents specific data collections focusing on semantic annotations associated to motions.

2.5.1 Motions

We present two types of motion datasets: from motion capture studios or motion estimated from monocular videos.

Motion capture (MoCap) data consists of collecting 3D data from markers placed on actors. MoCap has the property to being high quality but also expensive. The AMASS (Archive of Motion Capture As Surface Shapes) [Mahmood et al. 2019] dataset consists of more than 40 hours of motion data, spanning over 300 subjects, with more than 11,000 motions. AMASS is collected by combining several sources into a common format [Troje 2002; Müller et al. 2007; Sigal et al. 2010; Hoyet et al. 2012; Loper et al. 2014; Akhter et al. 2015; Mandery et al. 2015; Bogo et al. 2017; Trumble et al. 2017; Aristidou et al. 2019; Chatzitofis et al. 2020; Ghorbani et al. 2021; Advanced Computing Center for the Arts and Design n.d.; Lab n.d.; Ltd. n.d.; University et al. n.d.; *CMU Graphics Lab Motion Capture Database* 2003]. The motions are processed into the same SMPL [Loper et al. 2015] body format via the proposed MoSh++ algorithm [Loper et al. 2014; Mahmood et al. 2019]. The 100STYLE dataset [Mason et al. 2022] comprises an extensive collection of more than 4 million motion capture frames and contains a wide range of locomotion styles. All styles are performed by the same actor. Also Human3.6M [Ionescu et al. 2011; Ionescu et al. 2014] contains 3.6 million different 3D articulated poses captured from a set of professional actors.

Motion estimation from videos has recently made significant progress [Kanazawa et al. 2019; Kocabas et al. 2020; Luo et al. 2020; Kocabas et al. 2021; Kocabas et al. 2024]. It is now possible to obtain reasonably good quality motions from videos. This is a more cost-effective approach but can be error-prone as opposed to MoCap . VIBE [Kocabas et al. 2020] is based on the

single-image human shape estimation method SPIN [Kolotouros et al. 2019]. Similarly, HMMR [Kanazawa et al. 2019] extends the single-image method HMR [Kanazawa et al. 2018]. In Chapter 3, we adopt VIBE [Kocabas et al. 2020] to obtain training motion sequences from action-labeled video datasets. The Polarization Human Shape and Pose Dataset (PHSPD) [Zou et al. 2020] consists of 1061 motions. The authors fit SMPL pose parameters from an input stream of four cameras, one polarization camera and three Kinects. See the next section for more video-based datasets.

2.5.2 Semantic annotations

We present two types of annotations: with action labels and with text descriptions.

Action categories. The NTU RGB+D 120 dataset [Shahroudy et al. 2016; Liu et al. 2019a] is a video action recognition dataset. Each video sample is associated with depth map sequences, 3D skeletal data, and infrared (IR) sequences. Overall, there are about 114k videos. In the work of [Guo et al. 2020], the authors use a subset of 13 action categories (NTU-13) with SMPL parameters obtained with VIBE [Kocabas et al. 2020]. Similarly, the UESTC dataset [Ji et al. 2018] consists of 25K video sequences across 40 action categories (mostly exercises, and some represent cyclic movements). The HumanAct12 dataset [Guo et al. 2020] consists of action annotations of the PHSPD dataset [Zou et al. 2020]. HumanAct12 temporally trims the videos, annotates them into 12 action categories, and only provides their joint coordinates in a canonical frame. These three datasets (NTU-13, HumanAct12, UESTC) have been used for action-conditioned motion generation in Chapter 3. The BABEL [Punnakkal et al. 2021] dataset densely annotates AMASS [Mahmood et al. 2019] with text and action categories. See more details on BABEL in the next paragraph.

Text annotations. The KIT Motion-Language [Plappert et al. 2016] dataset (KIT-ML) provides text annotations from subsets of the KIT Whole-Body Human Motion Database [Mandery et al. 2015] and of the CMU Graphics Lab Motion Capture Database [*CMU Graphics Lab Motion Capture Database* 2003]. The dataset consists of 3,911 motion sequences with 6,353 sequence-level

text annotations, with 9.5 words per description on average. The motion capture data are originally processed with the Master Motor Map (MMM) framework [Terlemez et al. 2014] (see Figure 2.3a). Since AMASS combines multiple MoCap collections including KIT and CMU, in Chapter 4 we provide a mapping from KIT-ML to AMASS motions to be able to use SMPL pose parameters.

The BABEL dataset [Punnakkal et al. 2021] is a much larger collection than KIT-ML and contains sequence-level annotations as well frame-level annotations for AMASS [Mahmood et al. 2019] (i.e., breaking the full motion sequence into smaller segments). The unique property of BABEL is that it has multiple annotated segments within a sequence some of which also overlap, which allows us to investigate generation of a *sequential actions* (Annex A), as well as *simultaneous actions* (Annex B). In contrast, a textual label in KIT-ML [Plappert et al. 2016] covers the entire sequence.

Another widely used text-motion dataset is HumanML3D [Guo et al. 2022a], which provides textual descriptions for a subset of AMASS [Mahmood et al. 2019] and HumanAct12 [Guo et al. 2020] motion capture collections. It consists of 44,970 text annotations for 14,616 motions.

More recently, Motion-X [Lin et al. 2023] proposes a dataset of 3D whole-body motion (body+hands+face) with the SMPL-X [Pavlakos et al. 2019] representation. It consists of 81.1K motions from diverse sources (motion capture and estimation from videos). In this thesis, we use the KIT-ML dataset in Chapters 4 and 5, HumanML3D in Chapters 5 and 6 and finally, BABEL in Annex A, B.

Chapter 3

Action-Conditioned 3D Human Motion Synthesis with Transformer VAE

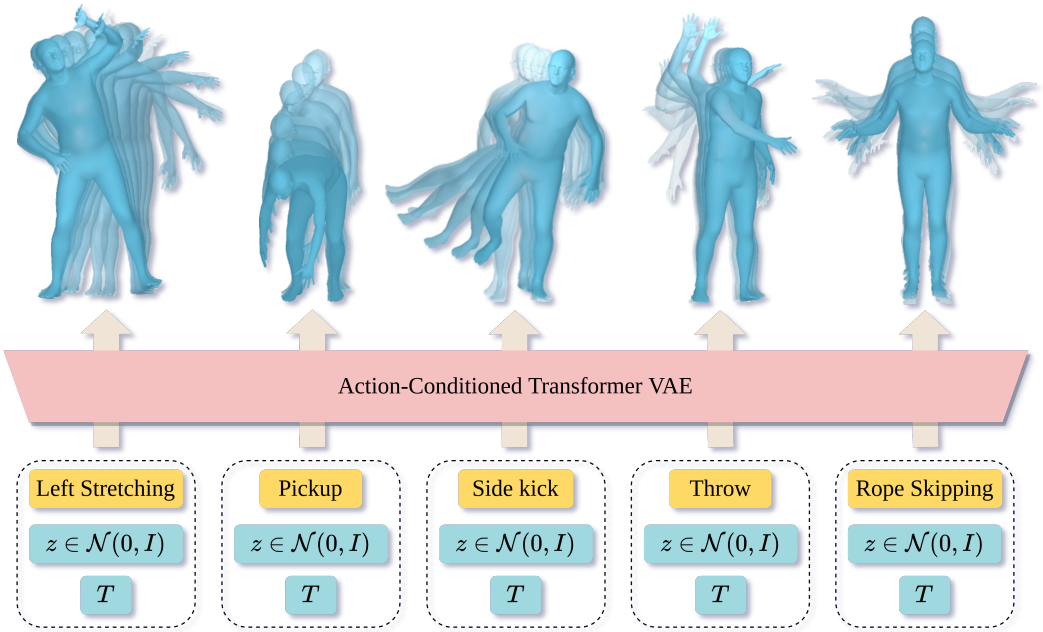


Figure 3.1: **Action-Conditioned TransfORmer VAE (ACTOR)** learns to synthesize human motion sequences conditioned on a categorical action and a duration, T. Sequences are generated by sampling from a single motion representation latent vector, z, as opposed to the frame-level embedding space in prior work.

This chapter presents our first contribution on the generation of 3D human motions, with the focus on the problem of action-conditioned generation of realistic and diverse human motion sequences.

In contrast to methods that complete, or extend, motion sequences, this task does not require an initial pose or sequence. Here we learn an action-aware latent representation for human motions by training a generative variational autoencoder (VAE). By sampling from this latent space and querying a certain duration through a series of positional encodings, we synthesize variable-length motion sequences conditioned on a categorical action. Specifically, we design a Transformer-based architecture, `ACTOR`, for encoding and decoding a sequence of parametric SMPL human body models estimated from action recognition datasets. We evaluate our approach on the NTU RGB+D, HumanAct12 and UESTC datasets and show improvements over the state of the art. Furthermore, we present two use cases: improving action recognition through adding our synthesized data to training, and motion denoising. Code, models and an illustrative video are available on our project page: https://mathis.petrovich.fr/actor.

3.1 Introduction

Despite decades of research on modeling human motions [Badler 1975; Badler et al. 1993], synthesizing realistic and controllable sequences remains extremely challenging. In this work, our goal is to take a semantic action label like "Throw" and generate an infinite number of realistic 3D human motion sequences, of varying length, that look like realistic throwing (Figure 3.1). A significant amount of prior work has focused on taking one pose, or a sequence of poses, and then predicting future motions [Habibie et al. 2017; Barsoum et al. 2018; Aksan et al. 2019; Yuan et al. 2020; Zhang et al. 2021]. This is an overly constrained scenario because it assumes that one already has a motion sequence and just needs more of it. On the other hand, many applications such as virtual reality and character control [Holden et al. 2017; Starke et al. 2019] require generating motions of a given type (semantic action label) with a specified duration.

We address this problem by training an action-conditioned generative model with 3D human motion data that has corresponding action labels. In particular, we construct a Transformer-based encoder-decoder architecture and train it with the VAE objective.

Transformer VAEs. The recent successes of Transformers in language tasks has increased interest in attention-based neural network models. Several works use Transformers in conjunction with generative VAE training. Particular examples include story generation [Fang et al. 2021], sentiment analysis [Cheng et al. 2019], response generation [Lin et al. 2020], and music generation [Jiang et al. 2020]. The work of [Jiang et al. 2020] learns latent embeddings per timeframe, while [Cheng et al. 2019] averages the hidden states to obtain a single latent code. On the other hand, [Fang et al. 2021] performs attention averaging to pool over time. In contrast to these works, we adopt learnable tokens as in [Devlin et al. 2019; Dosovitskiy et al. 2021] to summarize the input into a sequence-level embedding.

We parameterize the human body using SMPL [Loper et al. 2015] as it can output joint locations or the body surface. This paves the way for better modeling of interaction with the environment, as the surface is necessary to model contact. Moreover, such a representation allows the use of several reconstruction losses: constraining part rotations in the kinematic tree, joint locations, or surface points. The literature [Lee et al. 2018] and our results suggest that a combination of losses gives the most realistic generated motions. Furthermore, the possibility of obtaining a human mesh paves the way for better interaction modeling with the environment as it proves useful in dealing with contacts.

The key challenge of motion synthesis is to generate sequences that are perceptually realistic while being diverse. Many approaches for motion generation have taken an autoregressive approach such as LSTMs [Fragkiadaki et al. 2015] and GRUs [Martinez et al. 2017]. However, these methods typically regress to the mean pose after some time [Martinez et al. 2017] and are subject to drift. The key novelty in our Transformer model is to provide positional encodings to the decoder and to output the full sequence at once. Positional encoding has been popularized by recent work on neural radiance fields [Mildenhall et al.

2020]; we have not seen it used for motion generation as we do. This allows the generation of variable length sequences without the problem of the motions regressing to the mean pose. Moreover, our approach is, to our knowledge, the first to create an action-conditioned *sequence*-level embedding. The closest work is Action2Motion [Guo et al. 2020], which, in contrast, presents an autoregressive approach where the latent representation is at the *frame*-level. Getting a *sequence*-level embedding requires pooling the time dimension: we introduce a new way of combining Transformers and VAEs for this purpose, which also significantly improves performance over baselines.

A challenge specific to our action-condition generation problem is that there exists limited motion capture (MoCap) data paired with distinct action labels, typically on the order of 10 categories [Ionescu et al. 2014; *CMU Graphics Lab Motion Capture Database* 2003]. We instead rely on monocular motion estimation methods [Kocabas et al. 2020] to obtain 3D sequences for actions and present promising results on 40 fine-grained categories of the UESTC action recognition dataset [Ji et al. 2018]. In contrast to [Guo et al. 2020], we do not require multi-view cameras to process monocular trajectory estimates, which makes our model potentially applicable to larger scales. Despite being noisy, monocular estimates prove sufficient for training and, as a side benefit of our model, we are able to denoise the estimated sequences by encoding-decoding through our learned motion representation.

An action-conditioned generative model can augment existing motion capture datasets, which are expensive and limited in size [Mahmood et al. 2019; *CMU Graphics Lab Motion Capture Database* 2003]. Recent work, which renders synthetic human action videos for training action recognition models [Varol et al. 2021], shows the importance of motion diversity and large amounts of data per action. Such approaches can benefit from an infinite source of action-conditioned motion synthesis. We explore this through our experiments on action recognition. We observe that, despite a domain gap, the generated motions can serve as additional training data, specially in low-data regimes. This is a similar conclusion to [Zhao et al. 2020], but in our case the conditional model can synthesise motions in a controlled way (balanced training set), instead of having only pseudo-labels. Finally, a compact action-aware latent space for human motions can be used as a prior in other tasks such as human

motion estimation from videos.

Our contributions are fourfold: (i) We introduce ACTOR, a novel Transformer-based conditional VAE, and train it to generate action-conditioned human motions by sampling from a sequence-level latent vector. (ii) We demonstrate that it is possible to learn to generate realistic 3D human motions using noisy 3D body poses estimated from monocular video; (iii) We present a comprehensive ablation study of the architecture and loss components, obtaining state-of-the-art performance on multiple datasets; (iv) We illustrate two use cases for our model on action recognition and MoCap denoising.

3.2 Method

Problem definition. Actions defined by body-motions can be characterized by the rotations of body parts, independent of identity-specific body shape. To be able to generate motions with actors of different morphology, it is desirable to disentangle the pose and the shape. Consequently, without loss of generality, we employ the SMPL body model [Loper et al. 2015], which is a disentangled body representation (similar to recent models [Romero et al. 2017; Pavlakos et al. 2019; Osman et al. 2020; Xu et al. 2020]). Ignoring shape, our goal, is then to generate a sequence of *pose* parameters. More formally, given an action label a (from a set of predefined action categories $a \in A$) and a duration T, we generate a sequence of body poses $R_1, \ldots, R_T$ and a sequence of translations of the root joint represented as displacements, $D_1, \ldots, D_T$ (with $D_t \in \mathbb{R}^3, \forall t \in \{1, \ldots, T\}$).

Motion representation. SMPL pose parameters represent 23 joint rotations in the kinematic tree and one global rotation. We adopt the continuous 6D rotation representation for training [Zhou et al. 2019], making $R_t \in \mathbb{R}^{24\times 6}$. Let P_t be the combination of R_t and D_t, representing the pose and location of the body in a single frame, t. The full motion is the sequence $P_1, \ldots, P_T$. Given a generator output pose P_t and any shape parameter, we can obtain body mesh vertices (V_t) and body joint coordinates (J_t) differentiably using [Loper et al. 2015].

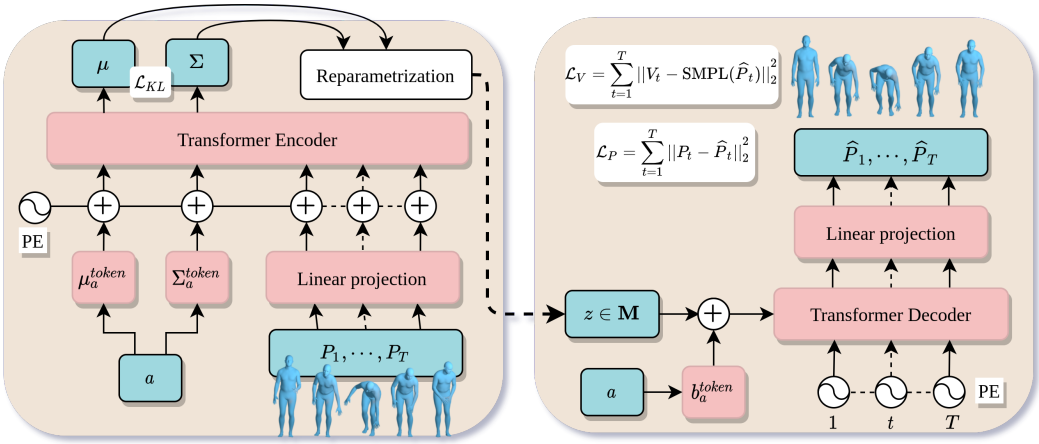


Figure 3.2: **Method overview:** We illustrate the encoder (left) and the decoder (right) of our Transformer-based VAE model that generates action-conditioned motions. Given a sequence of body poses $P_1, \ldots, P_T$ and an action label a, the encoder outputs distribution parameters on which we define a KL loss ($\mathcal{L}_{KL}$). We use extra learnable tokens per action (μ_a^{token} and Σ_a^{token}) as a way to obtain μ and Σ from the Transformer encoder. Using μ and Σ, we sample the motion latent representation $z \in \mathbf{M}$. The decoder takes the latent vector z, an action label a, and a duration T as input. The action determines the learnable b_a^{token} additive token, and the duration determines the number of positional encodings (PE) to input to the decoder. The decoder outputs the whole sequence $\widehat{P}_1, \ldots, \widehat{P}_T$ against which the reconstruction loss $\mathcal{L}_P$ is computed. In addition, we compute vertices with a differentiable SMPL layer to define a vertex loss ($\mathcal{L}_V$). For training z is obtained as the output of the encoder; for generation it is randomly sampled from a Gaussian distribution.

3.2.1 Conditional Transformer VAE for motions

We employ a conditional variational autoencoder (CVAE) model [Sohn et al. 2015] and input the action category information to both the encoder and the decoder. More specifically, our model is an action-conditioned Transformer VAE (`ACTOR`), whose encoder and decoder consist of Transformer layers (see Figure 3.2 for an overview).

Encoder. The encoder takes an arbitrary-length sequence of poses, and an action label a as input, and outputs distribution parameters μ and Σ of the motion latent space. Using the reparameterization trick [Kingma et al. 2014], we sample from this distribution a latent vector $z \in \mathbf{M}$ with $\mathbf{M} \subset \mathbb{R}^d$. All the input pose parameters (R) and translations (D) are first linearly embedded into a $\mathbb{R}^d$ space. As we embed arbitrary-length sequences into one latent space

(sequence-level embedding), we need to pool the temporal dimension. In other domains, a `[class]` token has been introduced for pooling purposes, e.g., in NLP with BERT [Devlin et al. 2019] and more recently in computer vision with ViT [Dosovitskiy et al. 2021]. Inspired by this approach, we similarly prepend the inputs with learnable tokens, and only use the corresponding encoder outputs as a way to pool the time dimension. To this end, we include two extra learnable parameters per action, μ_a^{token} and Σ_a^{token}, which we called "distribution parameter tokens". We append the embedded pose sequences to these tokens. The resulting Transformer encoder input is the summation with the positional encodings in the form of sinusoidal functions. We obtain the distribution parameters μ and Σ by taking the first two outputs of the encoder corresponding to the distribution parameter tokens (i.e., discarding the rest).

Decoder. Given a single latent vector z and an action label a, the decoder generates a realistic human motion for a given duration in one shot (i.e., not autoregressive).

We use a Transformer decoder model where we feed time information as a query (in the form of T sinusoidal positional encodings), and the latent vector combined with action information, as key and value. To incorporate the action information, we simply add a learnable bias b_a^{token} to shift the latent representation to an action-dependent space. The Transformer decoder outputs a sequence of T vectors in $\mathbb{R}^d$ from which we obtain the final poses $\widehat{P}_1, \ldots, \widehat{P}_T$ following a linear projection. A differentiable SMPL layer is used to obtain vertices and joints given the pose parameters as output by the decoder.

3.2.2 Training

We define several loss terms to train our model and present an ablation study in Section 3.3.2.

Reconstruction loss on pose parameters ($\mathcal{L}_P$). We use an L2 loss between the ground-truth poses $P_1, \ldots, P_T$, and our predictions $\widehat{P}_1, \ldots, \widehat{P}_T$ as $\mathcal{L}_P = \sum_{t=1}^{T} \|P_t - \widehat{P}_t\|_2^2$. Note that this loss contains both the SMPL rotations and the root translations. When we experiment by discarding the translations, we break

this term into two: $\mathcal{L}_R$ and $\mathcal{L}_D$, for rotations and translations, respectively.

Reconstruction loss on vertex coordinates ($\mathcal{L}_V$). We feed the SMPL poses P_t and $\widehat{P}_t$ to a differentiable SMPL layer (without learnable parameters) with a mean shape (i.e., $\beta = \vec{0}$) to obtain the root-centered vertices of the mesh V_t and $\widehat{V}_t$. We define an L2 loss by comparing to the ground-truth vertices V_t as $\mathcal{L}_V = \sum_{t=1}^{T} \|V_t - \widehat{V}_t\|_2^2$. We further experiment with a loss $\mathcal{L}_J$ on a more sparse set of points such as joint locations $\widehat{J}_t$ obtained through the SMPL joint regressor. However, as will be shown in Section 3.3.2, we do not include this term in the final model.

KL loss ($\mathcal{L}_{KL}$). As in a standard VAE, we regularize the latent space by encouraging it to be similar to a Gaussian distribution with μ the null vector and Σ the identity matrix. We minimize the Kullback–Leibler (KL) divergence between the encoder distribution and this target distribution.

The resulting total loss is defined as the summation of different terms: $\mathcal{L} = \mathcal{L}_P + \mathcal{L}_V + \lambda_{KL}\mathcal{L}_{KL}$. We empirically show the importance of weighting with λ_{KL} (equivalent to the β term in β-VAE [Higgins et al. 2017]) in our experiments to obtain a good trade-off between diversity and realism (see Section 3.3.2). The remaining loss terms are simply equally weighed, further improvements are potentially possible with tuning.

Implementation details. We use the AdamW optimizer with a fixed learning rate of 0.0001. The minibatch size is set to 20 and we found that the performance is sensitive to this hyperparameter (see Section 3.3.2). We train our model for 2000, 5000 and 1000 epochs on NTU-13, HumanAct12 and UESTC datasets, respectively. Overall, more epochs produce improved performance, but we stop training to retain a low computational cost. Note that to allow faster iterations, for ablations on loss and architecture, we train our models for 1000 epochs on NTU-13 and 500 epochs on UESTC. We finetune our model with variable-durations after pretraining on fixed-durations. For this, we restore the model weights from the fixed-duration pretraining and finetune for 100 additional epochs, with the same training hyperparameters. For all our experiments, we set the embedding dimensionality to 256. In the Transformer, we set the number of layers to 8, the number of heads in multi-head attention to 4, the

dropout rate to 0.1 and the dimension of the intermediate feedforward network to 1024. As in GPT-3 [Brown et al. 2020] and BERT [Devlin et al. 2019], we use Gaussian Linear Error Units (GELU) [Hendrycks et al. 2016] in our Transformer architecture. Our models are implemented with PyTorch [Paszke et al. 2019], and we use PyTorch3D [Johnson et al. 2020] to perform differentiable conversion between rotation representations. We integrate the differentiable SMPL layer using the PyTorch implementation of SMPL-X [Pavlakos et al. 2019]. For the evaluation metrics, we use the implementations provided by Action2Motion [Guo et al. 2020].

Runtime. Training takes 24 hours for 2K epochs on NTU, 19h hours for 5K epochs on HumanAct12, and 33 hours for 1K epochs on UESTC on a single Tesla V100 GPU, using 4GB GPU memory with batch size 20.

3.3 Experiments

We first introduce the datasets and performance measures used in our experiments (Section 3.3.1). Next, we present an ablation study (Section 3.3.2) and compare to previous work (Section 3.3.3). Then, we illustrate use cases in action recognition (Sections 3.3.4). Finally, we provide qualitative results and discuss limitations (Section 3.3.5).

3.3.1 Datasets and evaluation metrics

We use three datasets originally proposed for action recognition, mainly for skeleton-based inputs. Each dataset is temporally trimmed around one action per sequence. Next, we briefly describe them.

NTU RGB+D dataset [Shahroudy et al. 2016; Liu et al. 2019a]. To be able to compare to the work of [Guo et al. 2020], we use their subset of 13 action categories. [Guo et al. 2020] provide SMPL parameters obtained through VIBE estimations [Kocabas et al. 2020]. Their 3D root translations, obtained through multi-view constraints, are not publicly available, therefore we use their approximately origin-centered version. We refer to this data as

NTU-13 and use it for training.

HumanAct12 dataset [Guo et al. 2020]. Similarly, we use this data for state-of-the-art comparison. HumanAct12 is adapted from the PHSPD dataset [Zou et al. 2020] that releases SMPL pose parameters and root translations in camera coordinates for 1191 videos. HumanAct12 temporally trims the videos, annotates them into 12 action categories, and only provides their joint coordinates in a canonical frame. We also process the SMPL poses to align them to the frontal view.

UESTC dataset [Ji et al. 2018]. This recent dataset consists of 25K sequences across 40 action categories (mostly exercises, and some represent cyclic movements). To obtain SMPL sequences, we apply VIBE on each video and select the person track that corresponds best to the Kinect skeleton provided in case there are multiple people. We use all 8 static viewpoints (we discard the rotating camera) and canonicalize all bodies to the frontal view. We use the official cross-subject protocol to separate train and test splits, instead of the cross-view protocols since generating different viewpoints is trivial for our model. This results in 10650 training sequences that we use for learning the generative model, as well as the recognition model: the effective diversity of this set can be seen as 33 sequences per action on average (10K divided by 8 views, 40 actions). The remaining 13350 sequences are used for testing. Since the protocols on NTU-13 and HumanAct12 do not provide test splits, we rely on UESTC for recognition experiments.

Evaluation metrics. We follow the performance measures employed in [Guo et al. 2020] for quantitative evaluations. We measure FID, action recognition accuracy, overall diversity, and per-action diversity (referred to as multimodality in [Guo et al. 2020]). For all these metrics, a pretrained action recognition model is used, either for extracting motion features to compute FID, diversity, and multimodality; or directly the accuracy of recognition. For experiments on NTU-13 and HumanAct12, we directly use the provided recognition models of [Guo et al. 2020] that operate on joint coordinates. For UESTC, we train our own recognition model based on pose parameters expressed as 6D rotations (we observed that the joint-based models of [Guo et al. 2020] are sensitive to global viewpoint changes). We generate sets of sequences 20 times with

different random seeds and report the average together with the confidence interval at 95%. We refer to [Guo et al. 2020] for further details. One difference in our evaluation is the use of the average body shape parameter ($\beta = \vec{0}$) when obtaining joint coordinates from the mesh for both real and generated sequences. Note also that [Guo et al. 2020] only reports FID score comparing to the training split (FID_{tr}), since NTU-13 and HumanAct12 datasets do not provide test splits. On UESTC, we additionally provide an FID score on the test split as FID_{test}, which we rely most on to make conclusions.

3.3.2 Ablation study

We first ablate several components of our approach in a controlled setup, studying the loss and the architecture.

Loss study. Here, we investigate the influence of the reconstruction loss formulation when using the parametric SMPL body model in our VAE. We first experiment with using (i) only the rotation parameters $\mathcal{L}_R$, (ii) only the joint coordinates $\mathcal{L}_J$, (iii) only the vertex coordinates $\mathcal{L}_V$, and (iv) the combination $\mathcal{L}_R + \mathcal{L}_V$. Here, we initially discard the root translation to only assess the pose representation. For representing the rotation parameters, we use the 6D representation from [Zhou et al. 2019]. We explore other rotation representations in Table 3.1 and find that an axis-angle representation is difficult to train due to discontinuities, while others, such as quaternions, rotation matrices and 6D continuous representations are similar in performance on NTU-13. On UESTC, we obtain the best performance with the 6D representation and use this in all our experiments. In Table 3.2, we observe that a single loss is not sufficient to constrain the problem, especially losses on the coordinates do not converge to a meaningful solution on UESTC. On NTU-13, qualitatively, we also observe invalid body shapes since joint locations alone do not fully constrain the rotations along limb axes. We provide examples in our qualitative analysis. We conclude that using a combined loss significantly improves the results, constraining the pose space more effectively. We further provide an experiment on the influence of the weight parameter λ_{KL} controlling the KL divergence loss term $\mathcal{L}_{KL}$ in Table 3.3. We find that there is a trade-off between

diversity and realism that is best balanced at $\lambda_{KL} = 1e{-}5$. We use this value in all our experiments.

	UESTC					NTU-13			
	FID$_{tr}$↓	FID$_{test}$↓	Acc.↑	Div.→	Multimod.→	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm 0.26}$	$2.79^{\pm 0.29}$	$98.8^{\pm 0.1}$	$33.34^{\pm 0.32}$	$14.16^{\pm 0.06}$	$0.02^{\pm 0.00}$	$99.8^{\pm 0.0}$	$7.07^{\pm 0.02}$	$2.27^{\pm 0.01}$
Axis-angle	$513.39^{\pm 98.35}$	$531.88^{\pm 43.41}$	$16.4^{\pm 0.4}$	$19.75^{\pm 0.44}$	$1.81^{\pm 0.00}$	$14.98^{\pm 0.03}$	$41.7^{\pm 0.7}$	$5.29^{\pm 0.02}$	$1.96^{\pm 0.01}$
Quaternion	$281.9^{\pm 87.5}$	$305.02^{\pm 21.97}$	$41.2^{\pm 1.0}$	$23.48^{\pm 0.39}$	$14.57^{\pm 0.06}$	$0.20^{\pm 0.00}$	$95.6^{\pm 0.3}$	$7.08^{\pm 0.04}$	$2.23^{\pm 0.01}$
Rotation matrix	$277.14^{\pm 76.59}$	$300.29^{\pm 29.53}$	$41.6^{\pm 1.9}$	$22.25^{\pm 0.30}$	$14.56^{\pm 0.10}$	$0.17^{\pm 0.00}$	$95.9^{\pm 0.2}$	$7.08^{\pm 0.04}$	$2.19^{\pm 0.01}$
6D continuous	$20.02^{\pm 1.79}$	$23.64^{\pm 3.59}$	$90.5^{\pm 0.4}$	$32.77^{\pm 0.48}$	$14.64^{\pm 0.07}$	$0.17^{\pm 0.00}$	$96.4^{\pm 0.2}$	$7.08^{\pm 0.03}$	$2.12^{\pm 0.01}$

Table 3.1: **SMPL pose parameter representation:** We investigate different rotation representations for the SMPL pose parameters. While on NTU-13, all except axis-angle representations perform similarly, the best performing representation on UESTC is the 6D continuous representation [Zhou et al. 2019]. Note that the action recognition model which is used for evaluation is based on 6D rotations on UESTC and joint coordinates on NTU-13. Therefore, we convert each generation to these representations before evaluation.

	UESTC					NTU-13			
Loss	FID$_{tr}$↓	FID$_{test}$↓	Acc.↑	Div.→	Multimod.→	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm 0.26}$	$2.79^{\pm 0.29}$	$98.8^{\pm 0.1}$	$33.34^{\pm 0.32}$	$14.16^{\pm 0.06}$	$0.02^{\pm 0.00}$	$99.8^{\pm 0.0}$	$7.07^{\pm 0.02}$	$2.27^{\pm 0.01}$
$\mathcal{L}_J$	3M*	3M*	$3.3^{\pm 0.2}$	$267.68^{\pm 346.06}$	$153.62^{\pm 50.62}$	$0.49^{\pm 0.00}$	$93.6^{\pm 0.2}$	$7.04^{\pm 0.04}$	$2.12^{\pm 0.01}$
$\mathcal{L}_R$	$292.54^{\pm 113.35}$	$316.29^{\pm 26.05}$	$42.4^{\pm 1.7}$	$23.16^{\pm 0.47}$	$14.37^{\pm 0.08}$	$0.23^{\pm 0.00}$	$95.4^{\pm 0.2}$	$7.08^{\pm 0}$ .04	$2.18^{\pm 0.02}$
$\mathcal{L}_V$	4M*	4M*	$2.7^{\pm 0.2}$	$314.66^{\pm 476.18}$	$169.49^{\pm 27.90}$	$0.25^{\pm 0.00}$	$95.8^{\pm 0.3}$	$7.08^{\pm 0.04}$	$2.07^{\pm 0.01}$
$\mathcal{L}_R + \mathcal{L}_V$	$\mathbf{20.49}^{\pm 2.31}$	$\mathbf{23.43}^{\pm 2.20}$	$\mathbf{91.1}^{\pm 0.3}$	$31.96^{\pm 0.36}$	$14.66^{\pm 0.03}$	$\mathbf{0.19}^{\pm 0.00}$	$\mathbf{96.2}^{\pm 0.2}$	$7.09^{\pm 0.04}$	$2.08^{\pm 0.01}$

Table 3.2: **Reconstruction loss:** We define the loss on the SMPL pose parameters which represent the rotations in the kinematic tree ($\mathcal{L}_R$), their joint coordinates ($\mathcal{L}_J$), as well as vertex coordinates ($\mathcal{L}_V$). We show that constraining both rotations and vertex coordinates is critical to obtain smooth motions. In particular, coordinate-based losses alone do not converge to a meaningful solution on UESTC (*). → means motions are better when the metric is closer to real.

Root translation. Since we estimate the 3D human body motion from a monocular camera, obtaining the 3D trajectory of the root joint is not trivial for real training sequences, and is subject to depth ambiguity. We assume a fixed focal length and approximate the distance from the camera based on the ratio between the 3D body height and the 2D projected height. Similar to [Varol et al. 2021], we observe reliable translation in the xy image plane, but considerable noise in depth. Nevertheless, we still train with this type of data and visualize generated examples in Figure 3.3 with and without the loss on translation $\mathcal{L}_D$. Certain actions are defined by their trajectory (e.g., 'Left Stretching') and we are able to generate the semantically relevant translations

	UESTC					NTU-13			
	FID$_{tr}$↓	FID$_{test}$↓	Acc.↑	Div.→	Multimod.→	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm0.26}$	$2.79^{\pm0.29}$	$98.8^{\pm0.1}$	$33.34^{\pm0.32}$	$14.16^{\pm0.06}$	$0.02^{\pm0.00}$	$99.8^{\pm0.0}$	$7.07^{\pm0.02}$	$2.27^{\pm0.01}$
λ_{KL} = 1e−3	$460.72^{\pm90.36}$	$490.12^{\pm36.10}$	$34.4^{\pm1.4}$	$20.69^{\pm0.60}$	$1.25^{\pm0.00}$	$13.79^{\pm0.03}$	$46.6^{\pm0.7}$	$5.79^{\pm0.04}$	$1.53^{\pm0.01}$
λ_{KL} = 1e−4	$367.95^{\pm94.07}$	$390.68^{\pm41.02}$	$38.1^{\pm0.9}$	$20.91^{\pm0.38}$	$9.19^{\pm0.08}$	$9.90^{\pm0.02}$	$50.3^{\pm1.0}$	$6.15^{\pm0.04}$	$2.86^{\pm0.02}$
λ_{KL} = 1e−5	$20.02^{\pm1.79}$	$23.64^{\pm3.59}$	$90.5^{\pm0.4}$	$32.77^{\pm0.48}$	$14.64^{\pm0.07}$	$0.17^{\pm0.00}$	$96.4^{\pm0.2}$	$7.08^{\pm0.03}$	$2.12^{\pm0.01}$
λ_{KL} = 1e−6	$34.13^{\pm5.52}$	$39.74^{\pm3.57}$	$77.4^{\pm0.8}$	$29.60^{\pm0.35}$	$18.08^{\pm0.08}$	$13.83^{\pm0.03}$	$46.6^{\pm0.7}$	$5.78^{\pm0.04}$	$1.54^{\pm0.01}$
λ_{KL} = 1e−7	$80.05^{\pm7.71}$	$83.68^{\pm12.55}$	$47.1^{\pm2.1}$	$25.06^{\pm0.15}$	$19.96^{\pm0.08}$	$7.04^{\pm0.03}$	$43.0^{\pm2.1}$	$6.17^{\pm0.03}$	$4.18^{\pm0.01}$

Table 3.3: **Weighting the KL loss term:** To obtain a good trade-off between diversity and realism, it is important to find the balance between the reconstruction loss term and the KL loss term in training. We set the weight λ_{KL} to 1e−5 in our training.

despite noisy data. Compared to the real sequences, we observe much less noise in our generated sequences (see the supplemental video on our website).

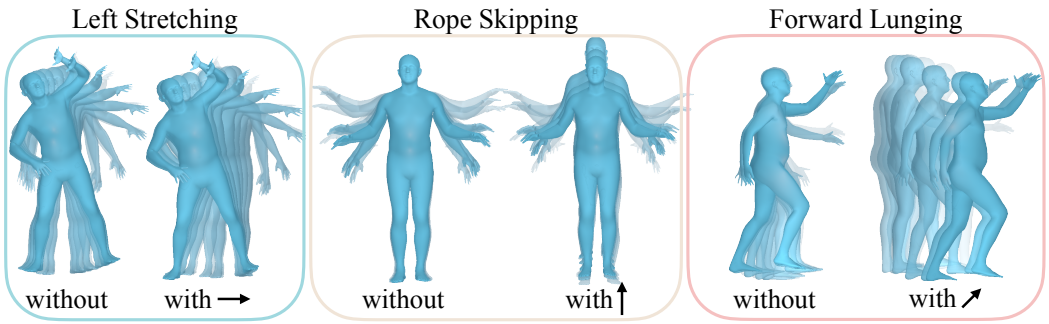


Figure 3.3: **Generating the 3D root translation:** Despite our model learning from noisy 3D trajectories, we show that our generations are smooth and they capture the semantics of the action. Examples are provided from the UESTC dataset for translations in x ('Left Stretching'), y (Rope Skipping), and z ('Forward Lugging') with and without the loss on the root displacement $\mathcal{L}_D$.

Architecture design. Next, we ablate several architectural choices. The first question is whether an attention-based design (i.e., Transformer) has advantages over the more widely used alternatives such as a simple fully-connected autoencoder or a GRU-based recurrent neural network. In Table 3.4, we see that our Transformer model outperforms both fully-connected and GRU encoder-decoder architectures on two datasets by a large margin. In contrast to the fully-connected architecture, we are also able to handle variable-length sequences. We further note that our sequence-level decoding strategy is key to obtain an improvement with Transformers, as opposed to an autoregressive Transformer decoder as in [Vaswani et al. 2017] (Table 3.4, a). At training time, the autoregressive model uses teacher forcing, i.e., using the ground-

	UESTC					NTU-13			
Architecture	$FID_{tr}\downarrow$	$FID_{test}\downarrow$	Acc.↑	Div.→	Multimod.→	$FID_{tr}\downarrow$	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm 0.26}$	$2.79^{\pm 0.29}$	$98.8^{\pm 0.1}$	$33.34^{\pm 0.32}$	$14.16^{\pm 0.06}$	$0.02^{\pm 0.00}$	$99.8^{\pm 0.0}$	$7.07^{\pm 0.02}$	$2.27^{\pm 0.01}$
Fully connected	$562.09^{\pm 48.12}$	$548.13^{\pm 38.34}$	$10.5^{\pm 0.5}$	$12.96^{\pm 0.11}$	$10.87^{\pm 0.05}$	$0.47^{\pm 0.00}$	$88.7^{\pm 0.6}$	$6.93^{\pm 0.03}$	$3.05^{\pm 0.01}$
GRU	$25.96^{\pm 3.02}$	$27.08^{\pm 2.98}$	$87.3^{\pm 0.4}$	$30.66^{\pm 0.33}$	$15.24^{\pm 0.08}$	$0.28^{\pm 0.00}$	$94.8^{\pm 0.2}$	$7.08^{\pm 0.04}$	$2.20^{\pm 0.01}$
Transformer	$\mathbf{20.49}^{\pm 2.31}$	$\mathbf{23.43}^{\pm 2.20}$	$\mathbf{91.1}^{\pm 0.3}$	$31.96^{\pm 0.36}$	$14.66^{\pm 0.03}$	$0.19^{\pm 0.00}$	$\mathbf{96.2}^{\pm 0.2}$	$7.09^{\pm 0.04}$	$2.08^{\pm 0.01}$
a) w/ autoreg. decoder	$55.75^{\pm 2.62}$	$60.10^{\pm 4.87}$	$88.4^{\pm 0.6}$	$33.46^{\pm 0.69}$	$10.62^{\pm 0.10}$	$2.62^{\pm 0.01}$	$88.0^{\pm 0.5}$	$6.80^{\pm 0.03}$	$1.76^{\pm 0.01}$
b) w/out $\mu_a^{token}, \Sigma_a^{token}$	$27.46^{\pm 3.43}$	$31.37^{\pm 3.04}$	$86.2^{\pm 0.4}$	$31.82^{\pm 0.38}$	$15.71^{\pm 0.12}$	$0.26^{\pm 0.00}$	$94.7^{\pm 0.2}$	$7.09^{\pm 0.03}$	$2.15^{\pm 0.01}$
c) w/out b_a^{token}	$24.38^{\pm 2.37}$	$28.52^{\pm 2.55}$	$89.4^{\pm 0.7}$	$32.11^{\pm 0.33}$	$14.52^{\pm 0.09}$	$\mathbf{0.16}^{\pm 0.00}$	$\mathbf{96.2}^{\pm 0.2}$	$7.08^{\pm 0.04}$	$2.19^{\pm 0.02}$

Table 3.4: **Architecture:** We compare various architectural designs, such as the encoder and the decoder of the VAE, and different components of the Transformer model, on both NTU-13 and UESTC datasets.

truth pose for the previous frame. This creates a gap with test time, where we observed poor autoencoding reconstructions such as decoding a left-hand waving encoding into a right-hand waving.

We also provide a controlled experiment by changing certain blocks of our Transformer VAE. Specifically, we remove the μ_a^{token} and Σ_a^{token} distribution parameter tokens and instead obtain μ and Σ by averaging the outputs of the encoder, followed by two linear layers (Table 3.4, b). This results in considerable drop in performance. Moreover, we investigate the additive b_a^{token} token and replace it with a one-hot encoding of the action label concatenated to the latent vector, followed by a linear projection (Table 3.4, c). Although this improves a bit the results on the NTU-13 dataset, we observe a large decrease in performance on the UESTC dataset which has a larger number of action classes.

We experiment with the number of Transformer layers in both of our encoder and decoder architectures. Table 3.5 summarizes the results. While 2 and 4 layers are sub-optimal, the performance difference between 6 and 8 layers is minimal. We use 8 layers in all our experiments.

Also, we find that the batch size significantly influences the performance. In Table 3.6, we report results with batch sizes of 10, 20, 30, 40 for a fixed learning rate. The best performance is obtained at 20, which is used in all our experiments.

Training with sequences of variable durations. A key advantage of sequence-modeling with architectures such as Transformers is to be able to

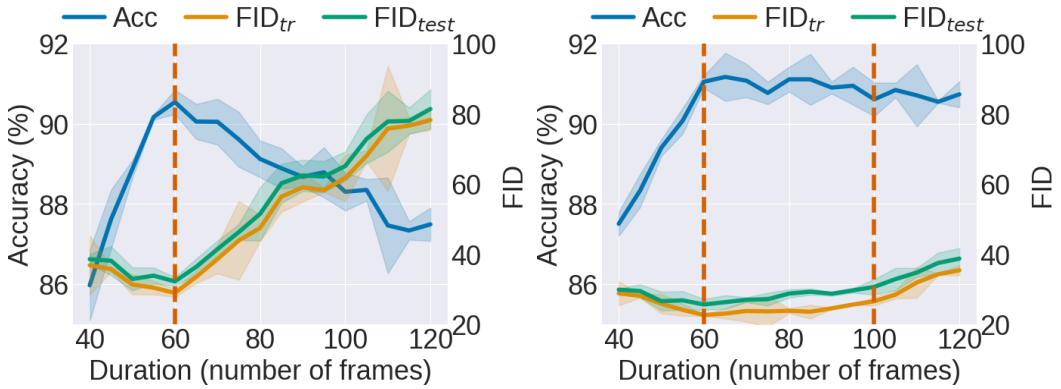


Figure 3.4: **Generating variable-length sequences:** We evaluate the capability of the models trained on UESTC with **(left)** fixed-size 60 frames and **(right)** variable-size between [60, 100] frames on generating various durations. We report accuracy and FID metrics. For the fixed model, we observe that the best performance is when tested at the seen duration of 60, but over 85% accuracy is retained even at ranges between [40, 120] frames. The performance is overall improved when the model has previously seen duration variations in training; there is a smaller drop in performance beyond the seen range (denoted with dashed lines).

	UESTC					NTU-13			
	FID$_{tr}$↓	FID$_{test}$↓	Acc.↑	Div.→	Multimod.→	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm0.26}$	$2.79^{\pm0.29}$	$98.8^{\pm0.1}$	$33.34^{\pm0.32}$	$14.16^{\pm0.06}$	$0.02^{\pm0.00}$	$99.8^{\pm0.0}$	$7.07^{\pm0.02}$	$2.27^{\pm0.01}$
2-layers	$34.66^{\pm2.58}$	$37.17^{\pm3.53}$	$84.9^{\pm0.6}$	$30.87^{\pm0.36}$	$15.83^{\pm0.08}$	$0.24^{\pm0.00}$	$94.6^{\pm0.2}$	$7.07^{\pm0.03}$	$2.22^{\pm0.01}$
4-layers	$23.93^{\pm1.50}$	$26.75^{\pm1.99}$	$88.9^{\pm0.5}$	$32.24^{\pm0.76}$	$15.06^{\pm0.06}$	$0.19^{\pm0.00}$	$96.1^{\pm0.2}$	$7.09^{\pm0.04}$	$2.10^{\pm0.01}$
6-layers	$21.68^{\pm1.78}$	$24.92^{\pm2.09}$	$89.0^{\pm0.6}$	$32.61^{\pm0.41}$	$15.31^{\pm0.05}$	$0.16^{\pm0.00}$	$96.6^{\pm0.1}$	$7.09^{\pm0.04}$	$2.11^{\pm0.01}$
8-layers	$20.02^{\pm1.79}$	$23.64^{\pm3.59}$	$90.5^{\pm0.4}$	$32.77^{\pm0.48}$	$14.64^{\pm0.07}$	$0.17^{\pm0.00}$	$96.4^{\pm0.2}$	$7.08^{\pm0.03}$	$2.12^{\pm0.01}$

Table 3.5: **Number of layers:** We use 8 layers in both the encoder and the decoder of the Transformer VAE. While the performance degrades at 2 or 4 layers, we see marginal gains after 6 layers.

handle variable-length motions. At generation time, we control how long the model should synthesize by specifying a sequence of positional encodings to the decoder. We can trivially generate more diversity by synthesizing sequences of different durations. However, so far we have trained our models with fixed-size inputs, i.e., 60 frames. Here, we first analyze whether a fixed-size trained model can directly generate variable output sizes. This is presented in Figure 3.4 (left). We plot the performance over several sets of generations of different lengths between 40 and 120 frames (with a step size of 5). Since our recognition model used for evaluation metrics is trained on fixed-size 60-frame inputs, we naturally observe performance decrease outside of this length. However, the

	UESTC					NTU-13			
	$FID_{tr}\downarrow$	$FID_{test}\downarrow$	Acc.↑	Div.→	Multimod.→	$FID_{tr}\downarrow$	Acc.↑	Div.→	Multimod.→
Real	$2.93^{\pm0.26}$	$2.79^{\pm0.29}$	$98.8^{\pm0.1}$	$33.34^{\pm0.32}$	$14.16^{\pm0.06}$	$0.02^{\pm0.00}$	$99.8^{\pm0.0}$	$7.07^{\pm0.02}$	$2.27^{\pm0.01}$
Batch size = 10	$283.28^{\pm94.40}$	$309.15^{\pm33.90}$	$39.7^{\pm1.5}$	$23.24^{\pm0.43}$	$15.73^{\pm0.11}$	$13.95^{\pm0.03}$	$46.2^{\pm0.6}$	$5.77^{\pm0.05}$	$1.56^{\pm0.01}$
Batch size = 20	$20.02^{\pm1.79}$	$23.64^{\pm3.59}$	$90.5^{\pm0.4}$	$32.77^{\pm0.48}$	$14.64^{\pm0.07}$	$0.17^{\pm0.00}$	$96.4^{\pm0.2}$	$7.08^{\pm0.03}$	$2.12^{\pm0.01}$
Batch size = 30	$23.37^{\pm2.95}$	$26.06^{\pm1.28}$	$89.7^{\pm0.5}$	$32.07^{\pm0.58}$	$14.59^{\pm0.05}$	$0.18^{\pm0.00}$	$96.2^{\pm0.2}$	$7.07^{\pm0.04}$	$2.13^{\pm0.01}$
Batch size = 40	$25.36^{\pm1.82}$	$28.22^{\pm2.16}$	$89.2^{\pm0.7}$	$32.22^{\pm0.44}$	$14.52^{\pm0.10}$	$0.26^{\pm0.00}$	$95.4^{\pm0.1}$	$7.06^{\pm0.05}$	$2.10^{\pm0.01}$

Table 3.6: **Batch size:** We observe sensitivity of the Transformer VAE training to different batch sizes and report the performance at several batch size values. We set this hyperparameter to 20 in our training.

accuracy still remains high which indicates that our model is already capable of generating diverse durations.

Next, we *train* our generative model with variable-length inputs by randomly sampling a sequence between 60 and 100 frames. However, simply training this way from random weight initialization converges to a poor solution, leading all generated motions to be frozen in time. We address this by pretraining at 60-frame fixed size and finetuning at variable sizes. We see in Figure 3.4 (right) that the performance is greatly improved with this model.

Furthermore, we investigate how the generations longer or shorter than their average durations behave. We observe qualitatively that shorter generations produce partial actions e.g., picking up without reaching the floor, and longer generations slow down somewhat non-uniformly in time. We refer to the supplemental video on our website for qualitative results.

3.3.3 Comparison to the state of the art

Action2Motion [Guo et al. 2020] is the only prior work that generates action-conditioned motions. We compare to them in Table 3.7 on their NTU-13 and HumanAct12 datasets. On both datasets, we obtain significant improvements over this prior work that uses autoregressive GRU blocks, as well as other baselines implemented by [Guo et al. 2020] by adapting other works [Cai et al. 2018; Tulyakov et al. 2018]. The improvements over [Guo et al. 2020] can be explained mainly by removing autoregression and adding the proposed learnable tokens (Table 3.4). Note that our GRU implementation obtains

	NTU-13				HumanAct12			
Method	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→	FID$_{tr}$↓	Acc.↑	Div.→	Multimod.→
Real [Guo et al. 2020]	$0.03^{\pm 0.00}$	$99.9^{\pm 0.1}$	$7.11^{\pm 0.05}$	$2.19^{\pm 0.03}$	$0.09^{\pm 0.01}$	$99.7^{\pm 0.1}$	$6.85^{\pm 0.05}$	$2.45^{\pm 0.04}$
Real*	$0.02^{\pm 0.00}$	$99.8^{\pm 0.0}$	$7.07^{\pm 0.02}$	$2.25^{\pm 0.01}$	$0.02^{\pm 0.00}$	$99.4^{\pm 0.0}$	$6.86^{\pm 0.03}$	$2.60^{\pm 0.01}$
CondGRU [Guo et al. 2020]†	$28.31^{\pm 0.14}$	$7.8^{\pm 0.1}$	$3.66^{\pm 0.02}$	$3.58^{\pm 0.03}$	$40.61^{\pm 0.14}$	$8.0^{\pm 0.2}$	$2.38^{\pm 0.02}$	$2.34^{\pm 0.04}$
Two-stage GAN [Cai et al. 2018]†	$13.86^{\pm 0.09}$	$20.2^{\pm 0.3}$	$5.33^{\pm 0.04}$	$3.49^{\pm 0.03}$	$10.48^{\pm 0.09}$	$42.1^{\pm 0.6}$	$5.96^{\pm 0.05}$	$2.81^{\pm 0.04}$
Act-MoCoGAN [Tulyakov et al. 2018]†	$2.72^{\pm 0.02}$	$\mathbf{99.7}^{\pm 0.1}$	$6.92^{\pm 0.06}$	$0.91^{\pm 0.01}$	$5.61^{\pm 0.11}$	$79.3^{\pm 0.4}$	$6.75^{\pm 0.07}$	$1.06^{\pm 0.02}$
Action2Motion [Guo et al. 2020]	$0.33^{\pm 0.01}$	$94.9^{\pm 0.1}$	$7.07^{\pm 0.04}$	$2.05^{\pm 0.03}$	$2.46^{\pm 0.08}$	$92.3^{\pm 0.2}$	$7.03^{\pm 0.04}$	$2.87^{\pm 0.04}$
ACTOR (ours)	$\mathbf{0.11}^{\pm 0.00}$	$97.1^{\pm 0.2}$	$7.08^{\pm 0.04}$	$2.08^{\pm 0.01}$	$\mathbf{0.12}^{\pm 0.00}$	$\mathbf{95.5}^{\pm 0.8}$	$6.84^{\pm 0.03}$	$2.53^{\pm 0.02}$

Table 3.7: **State-of-the-art comparison:** We compare to the recent work of [Guo et al. 2020] on the NTU-13 and HumanAct12 datasets. Note that due to differences in implementation (e.g., random sampling, using zero shape parameter), our metrics for the ground truth real data (Real*) are slightly different than the ones reported in their paper. The performance improvement with our Transformer-based model shows a clear gap from Action2Motion. † Baselines implemented by [Guo et al. 2020].

similar performance as [Guo et al. 2020], while using the same hyperparameters as the Transformer. In addition to the quantitative performance improvement, measured with recognition models based on joint coordinates, our model can directly output human meshes, which can further be diversified with varying the shape parameters. [Guo et al. 2020] instead applies an optimization step to fit SMPL models on their generated joint coordinates, which is typically substantially slower than a neural network forward pass.

Jitter removal for Action2Motion [Guo et al. 2020]. Besides the quantitative improvement of ACTOR over Action2Motion, we observe qualitatively that Action2Motion generations have significant temporal jitter. To investigate whether our improvement stems from this difference, we removed jitter (using 1€ filter) from Action2Motion generations (that we obtained with their code). The result becomes worse (FID: 0.41 → 0.63, Acc: 94.3% → 93.0%)[1], perhaps because the real data also has considerable jitter. This suggests that our significant quantitative improvement can be attributed to other factors such as more distinguishable actions.

[1]These two values for Action2Motion does not match Table 3 of the paper because we use our own evaluation script and normalized the ground truth shape by taking the average shape of SMPL.

3.3.4 Use cases in action recognition

In this section, we test the limits of our approach by illustrating the benefits of our generative model and our learned latent representation for the skeleton-based action recognition task. We adopt a standard architecture, ST-GCN [Yan et al. 2018], that employs spatio-temporal graph convolutions to classify actions. We show that we can use our latent encoding for denoising motion estimates and our generated sequences as data augmentation to action recognition models.

Use case I: Human motion denoising. In the case when our motion data source relies on monocular motion estimation such as [Kocabas et al. 2020], the training motions remain noisy. We observe that by simply encoding-decoding the real motions through our learned embedding space, we obtain much cleaner motions. Since it is difficult to show motion quality results on static figures, we refer to our supplemental video on our website to see this effect. We measure the denoising capability of our model through an action recognition experiment in Table 3.8. We change both the training and test set motions with the encoded-decoded versions. We show improved performance when trained and tested on $\text{Real}_{denoised}$ motions (97.0%) compared to Real_{orig} (91.8). Note that this result on its own is not sufficient for this claim, but is only an indication since our decoder might produce less diversity than real motions. Moreover, the action label is given at denoising time. We believe that such denoising can be beneficial in certain scenarios where the action is known, e.g., occlusion or missing markers during MoCap collection.

Use case II: Augmentation for action recognition. Next, we augment the real training data (Real_{orig}), by adding generated motions to the training. We first measure the action recognition performance without using real sequences. We consider interpolating existing Real_{orig} motions that fall within the same action category in our embedding space to create intra-class variations ($\text{Real}_{interpolated}$). We then synthesize motions by sampling noise vectors conditioned on each action category (Generated). Table 3.8 summarizes the results. Training only on synthetic data performs 80.7% on the real test set, which is promising. However, there is a domain gap between the noisy real motions and our smooth generations. Consequently, adding generated motions to real training only marginally improves the performance. In Figure 3.5,

	Test accuracy (%)	
	Real_{orig}	$\text{Real}_{denoised}$
Real_{orig}	91.8	93.2
$\text{Real}_{denoised}$	83.8	97.0
$\text{Real}_{interpolated}$	77.6	93.9
Generated	80.7	97.0
Real_{orig} + Generated	**91.9**	**98.3**

Table 3.8: **Action recognition:** We employ a standard architecture (ST-GCN [Yan et al. 2018]) and perform action recognition experiments using several sets of training data on the UESTC cross-subject protocol [Ji et al. 2018]. Training only with generated samples obtains 80% accuracy on the real test set which is another indication our action-conditioning performs well. Nevertheless, we observe a domain gap between generated and real samples, mainly due to the noise present in the real data. We show that simply by encoding-decoding the test sequences, we observe a denoising effect, which in turn shows better performance. However, one should note that the last-column experiments are not meant to improve performance in the benchmark since it uses the action label information.

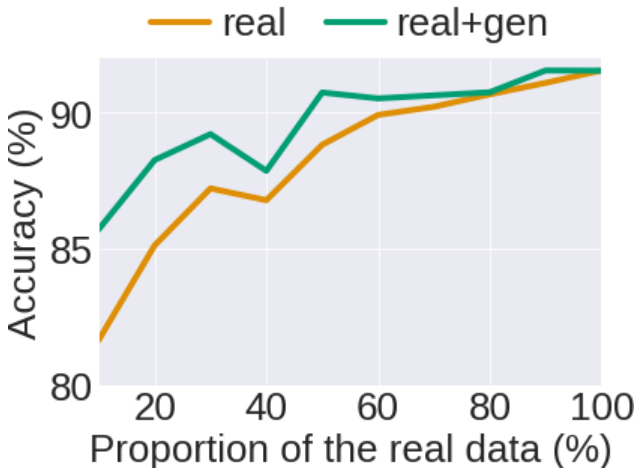


Figure 3.5: **Data augmentation:** We show the benefit of augmenting the real data with our generative model (real+gen), especially at low-data regime. We have limited gains when the real data is sufficiently large.

we investigate whether the augmented training helps for low-data regimes by training at several fractions of the data. In each minibatch we equally sample real and generated motions. However, in theory we have access to infinitely many generations. We see that the improvement is more visible at low-data regime.

3.3.5 Qualitative results

In Figure 3.6, we visualize several examples from our generations. We observe a great diversity in the way a given action is performed. For example, the 'Throw' action is performed with left or right hand. We also see different timings, e.g., different moments when the hands are raised in the 'Jumping Jack' action. We notice that the model keeps the essence of the action semantics while changing nuances (angles, speed, phase) or action-irrelevant body parts. We refer to the supplemental video on our website for further qualitative analyses.

One limitation of our model is that the maximum duration it can generate depends on computational resources since we output all of the sequence at once. Moreover, the actions are from a predefined set. Future work will explore open-vocabulary actions, which might become possible with further progress in 3D motion estimation from unconstrained videos.

Video. We provide a supplemental video on our website to illustrate qualitatively the diversity in our generations and compare with Action2Motion [Guo et al. 2020]. Moreover, we show the effect of the choice of the loss function introduced in the paper. We further present results of changing the duration of the generations. We also inspect the latent space by interpolating the noise vector. Finally, we present the denoising capability of our model by encoding-decoding through our latent space. This takes jerky motions and produces smooth but natural looking motion.

3.4 Conclusions

In this chapter, we presented a new Transformer-based VAE model to synthesize action-conditioned human motions. We provided a detailed analysis to assess different components of our proposed approach. We obtained state-of-the-art performance on action-conditioned motion generation, significantly improving over prior work. Furthermore, we explored various use cases in motion denoising and action recognition. One especially attractive property of our method is that it operates on a sequence-level latent space. This Transformer-based VAE

model will then be used as the basis for many new methods, in particular in the next chapter, for the text-motion synthesis task.

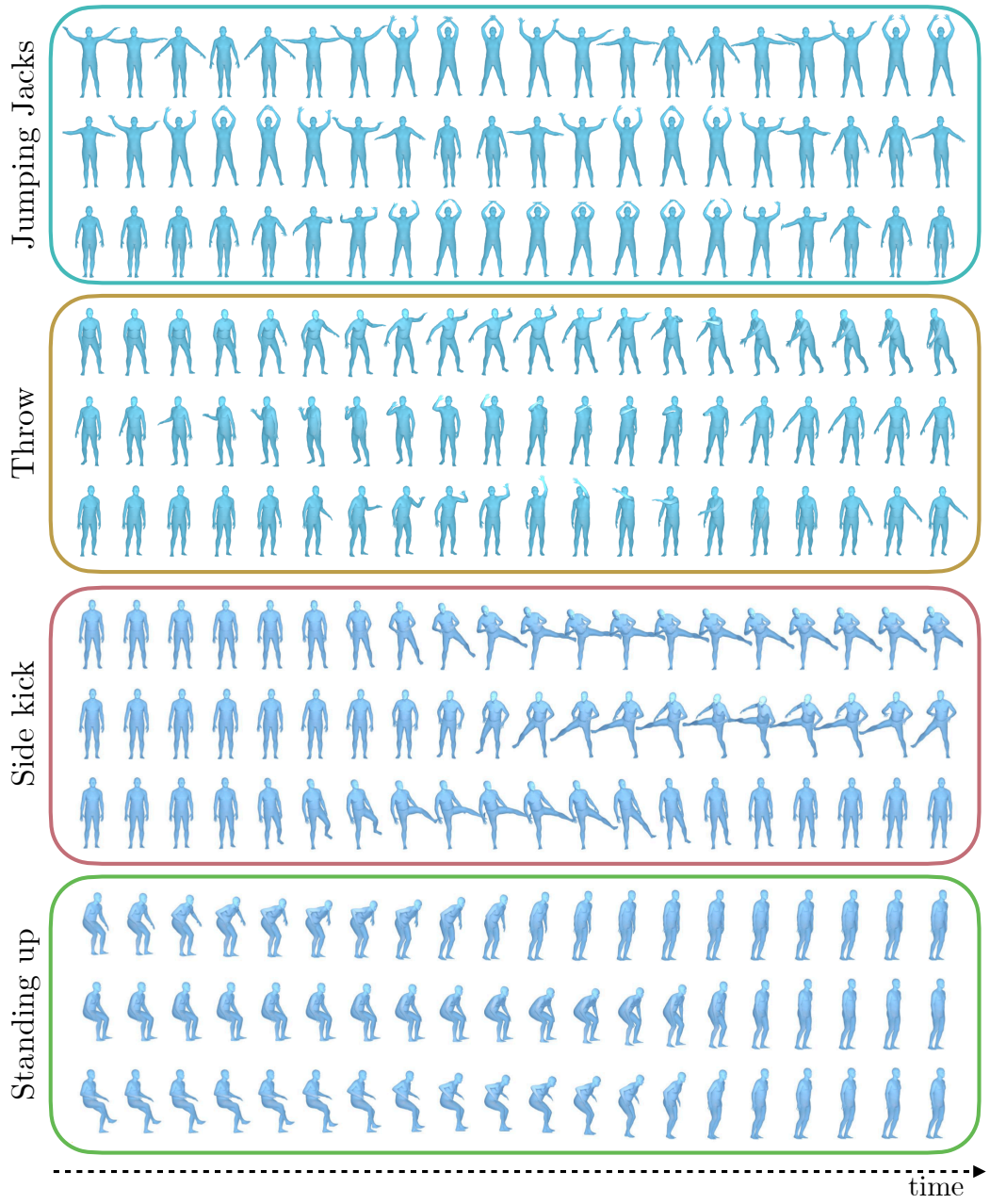


Figure 3.6: **Qualitative results:** We illustrate the diversity of our generations for several actions by showing 3 generations. The horizontal axis represents the time axis and 20 equally spaced frames are visualized out of 60-frame generations. We demonstrate that our model is capable of generating different ways of performing a given action. More results can be found in the supplemental video on our website.

Chapter 4

Generating Diverse Human Motions from Textual Descriptions

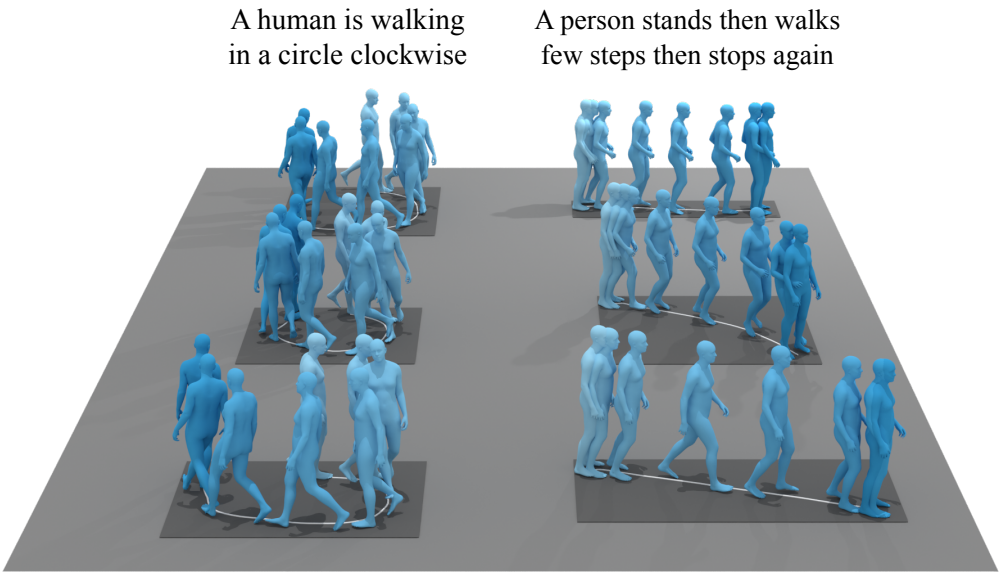


Figure 4.1: **Text-to-Motions** (`TEMOS`) learns to synthesize human motion sequences conditioned on a textual description and a duration. SMPL pose sequences are generated by sampling from a single latent vector, z. Here, we illustrate the diversity of our motions on two sample texts, providing three generations per text input. Each image corresponds to a motion sequence where we visualize the root trajectory projected on the ground plane and the human poses at multiple equidistant time frames. The flow of time is shown with a color code where lighter blue denotes the past.

This chapter presents our second contribution on the generation of 3D human motions. We focus this time on the generation of realistic and diverse human motion sequences from text descriptions.

This challenging task of text-to-motion synthesis requires joint modeling of both modalities: understanding and extracting useful human-centric information from the text, and then generating plausible and realistic sequences of human poses. In contrast to most previous work which focuses on generating a single, deterministic, motion from a textual description, we design a variational approach that can produce *multiple* diverse human motions. We propose `TEMOS`, a text-conditioned generative model leveraging variational autoencoder (VAE) training with human motion data, in combination with a text encoder that produces distribution parameters compatible with the VAE latent space. We show the `TEMOS` framework can produce both skeleton-based animations as in prior work, as well more expressive SMPL body motions. We evaluate our approach on the KIT Motion-Language benchmark and, despite being relatively straightforward, demonstrate significant improvements over the state of the art. Code, models and an illustrative video are available on our project page: https://mathis.petrovich.fr/temos.

4.1 Introduction

We explore the problem of generating 3D human motions, i.e., sequences of 3D poses, from natural language textual descriptions (in English). Generating text-conditioned human motions has numerous applications both for the virtual (e.g., game industry) and real worlds (e.g., controlling a robot with speech for personal physical assistance). For example, in the film and game industries, motion capture is often used to create special effects featuring humans. Motion capture is expensive, therefore technologies that automatically synthesize new motion data could save time and money.

Language represents a natural interface for people to interact with computers [Hill 1983], and our work provides a foundational step towards creating human animations using natural language input. The problem of generating

human motion from free-form text, however, is relatively new since it relies on advances in both language modeling and human motion synthesis. Regarding the former, we build on advances in language modeling using transformers. In terms of human motion synthesis, much of the previous work has focused on generating motions conditioned on a single action label, not a sentence, e.g., (Chapter 3, [Guo et al. 2020]). Here we go further by encoding both the language and the motion using transformers in a joint latent space. The approach is relatively straightforward, yet achieves results that significantly outperform the latest state of the art. We perform extensive experiments and ablation studies to understand which design choices are critical.

Despite recent efforts in this area, most current methods generate only *one* output motion per text input [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021]. That is, with the input "A man walks in a circle", these methods synthesize one motion. However, one description often can map to *multiple* ways of performing the actions, often due to ambiguities and lack of details, e.g., in our example, the size and the orientation of the circle are not specified. An ideal generative model should therefore be able to synthesize multiple sequences that respect the textual description while exploring the degrees of freedom to generate natural variations. While, in theory, the more precise the description becomes, the less space there is for diversity; it is a desirable property for natural language interfaces to manage intrinsic ambiguities of linguistic expressions [Gao et al. 2015]. In this chapter, we propose a method that allows sampling from a distribution of human motions conditioned on natural language descriptions. Figure 4.1 illustrates multiple sampled motions generated from two input texts; check our project webpage for video examples.

A key challenge is building models that are effective for temporal modeling. Most prior work employs autoregressive models that iteratively decode the next time frame given the past. These approaches may suffer from drift over time and often, eventually, produce static poses [Martinez et al. 2017]. In contrast, sequence-level generative models encode an entire sequence and can exploit long-range context. In this work, we incorporate the powerful Transformer models [Vaswani et al. 2017], which have proven effective for various sequence modeling tasks [Devlin et al. 2019; Bain et al. 2021]. We design a simple yet effective architecture, where both the motion and text are input to Transformer

encoders before projecting them to a cross-modal joint space. Similarly, the motion decoder uses a Transformer architecture taking positional encodings and a latent vector as input, and generating a 3D human motion (see Figure 4.2). Notably, a single sequence-level latent vector is used to decode the motion in one shot, without any autoregressive iterations. Through detailed ablation studies, we show that the main improvement over prior work stems from this design.

Most similar to our work is Ghosh et. al. [Ghosh et al. 2021], which builds on Language2Pose [Ahuja et al. 2019]. Our key difference is the integration of a variational approach for sampling a diverse set of motions from a single text. Our further improvements include the use of Transformers to encode motion sequences into a single embedding instead of the autoregressive approach in [Ghosh et al. 2021]. This allows us to encode distribution parameters of the VAE as in Chapter 3, proving effective in our state-of-the-art results. [Ghosh et al. 2021] also encode the upper body and lower body separately, whereas our approach does not need such hand crafting.

A well-known challenge common to generative models is the difficulty of evaluation. While many metrics are used in evaluating generated motions, each of them is limited. Consequently, in this work, we rely on both quantitative measures that compare against the ground truth motion data associated with each test description, and human perceptual studies to evaluate the perceived quality of the motions. The former is problematic particularly for this work, because it assumes one true motion per text, but our method produces multiple motions due to its probabilistic nature. We find that human judgment of motion quality is necessary for a full picture.

Moreover, the state of the art reports results on the task of future motion prediction. Specifically, [Ghosh et al. 2021] assume the first pose in the generated sequence is available from the ground truth. In contrast, we evaluate our method by synthesizing the full motion from scratch; i.e. without conditioning on the first, ground truth, frame. We provide results for various settings, e.g., comparing a random generation against the ground truth, or picking the best out of several generations. We outperform previous work even when sampling a single random generation, but the performance improves as we increase the

number of generations and pick the best.

A further addition we make over existing text-to-motion approaches is to generate sequences of SMPL body models [Loper et al. 2015]. Unlike classical skeleton representations, the parametric SMPL model provides the body surface, which can support future research on motions that involve interaction with objects or the scene. Such skinned generations were considered in other work on unconstrained or action-conditioned generation [Petrovich et al. 2021; Zhang et al. 2021]. Here, we demonstrate promising results for the text-conditioning scenario as well. The fact that the framework supports multiple body representations, illustrates its generality.

In summary, our contributions are the following: (i) We present Text-to-Motions (`TEMOS`), a novel cross-modal variational model that can produce diverse 3D human movements given textual descriptions in natural language. (ii) In our experiments, we provide an extensive ablation study of the model components and outperform the state of the art by a large margin both on standard metrics and through perceptual studies. (iii) We go beyond stick figure generations, and exploit the SMPL model for text-conditioned body surface synthesis, demonstrating qualitatively appealing results.

4.2 Method

In this section, we start by formulating the problem (Section 4.2.1). We then provide details on our model design (Section 4.2.2), as well as our training strategy (Section 4.2.3).

4.2.1 Task definition

Given a sentence describing a motion, the goal is to generate various sequences of 3D human poses and trajectories that match the textual input. Next, we describe the representation for the text and motion data.

Textual description represents a written natural language sentence (e.g., in English) that describes what and how a human motion is performed. The sentence can include various levels of detail: a precise sequence of actions such as *"A human walks two steps and then stops"* or a more ambiguous description such as *"A man walks in a circle"*. The data structure is a sequence of words $W_{1:N} = W_1, \ldots, W_N$ from the English vocabulary.

3D human motion is defined as a sequence of human poses $H_{1:F} = H_1, \ldots, H_F$, where F denotes the number of time frames. Each pose H_f corresponds to a representation of the articulated human body. In this work, we employ two types of body motion representations: one based on skeletons, one based on SMPL [Loper et al. 2015]. First, to enable a comparison with the state of the art, we follow the rotation-invariant skeleton representation from Holden et. al. [Holden et al. 2016], which is used in the previous work we compare with [Ahuja et al. 2019; Ghosh et al. 2021]. Second, we incorporate the parametric SMPL representation by encoding the global root trajectory of the body and parent-relative joint rotations in 6D representation [Zhou et al. 2019]. We refer the reader to Chapter 2 for more details about motion representations.

More generally, a human motion can be represented by a sequence of F poses each with p dimensions, so that at frame f, we have $H_f \in \mathbb{R}^p$. Our goal is, given a textual description $W_{1:N}$, to sample from a distribution of plausible motions $H_{1:F}$ and to generate multiple hypotheses.

4.2.2 `TEMOS` model architecture

Following [Ahuja et al. 2019], we learn a joint latent space between the two modalities: motion and text (see Figure 4.2). To incorporate generative modeling in such an approach, we employ a VAE [Kingma et al. 2014] formulation that requires architectural changes. We further employ Transformers [Vaswani et al. 2017] to obtain sequence-level embeddings both for the text and motion data. Next, we describe the two encoders for motion and text, followed by the motion decoder.

Motion and text encoders. We have two encoders for representing motion $\mathscr{M}_{enc}$ and text $\mathscr{T}_{enc}$ in a joint space. The encoders are designed to be as symmetric as possible across the two modalities. To this end, we adapt the ACTOR Transformer-based VAE motion encoder from Chapter 3 by making it class-agnostic (i.e., removing action conditioning). This encoder takes as input a sequence of vectors of arbitrary length, as well as learnable *distribution tokens*. The outputs corresponding to the distribution tokens are treated as Gaussian distribution parameters μ and Σ of the sequence-level latent space. Using the reparameterization trick [Kingma et al. 2014], we sample a latent vector $z \in \mathbb{R}^d$ from this distribution (see Figure 4.2). The latent space dimensionality d is set to 256 in our experiments.

For the motion encoder $\mathscr{M}_{enc}$, the input sequence of vectors is $H_{1:F}$, representing the poses. For the text encoder $\mathscr{T}_{enc}$, the inputs are word embeddings for $W_{1:N}$ obtained from a pretrained language model DistilBERT [Sanh et al. 2019]. We freeze the weights of DistilBERT unless stated otherwise.

Motion decoder. The motion decoder $\mathscr{M}_{dec}$ is a Transformer decoder (as in ACTOR in Chapter 3, but without the bias token to make it class agnostic), so that given a latent vector z and a duration F, we generate a 3D human motion sequence $\widehat{H}_{1:F}$ non-autoregressively from a single latent vector. Note that such approach does not require masks in self-attention, and tends to provide a globally consistent motion. The latent vector is obtained from one of the two encoders during training (described next, in Section 4.2.3), and the duration is represented as a sequence of positional encodings in the form of sinusoidal functions. We note that our model can produce variable durations, which is another source of diversity (see supplementary video on our website).

4.2.3 Training strategy

For our cross-modal neural network training, we sample a batch of text-motion pairs at each training iteration. In summary, both input modalities go through their respective encoders, and both encoded vectors go through the motion decoder to reconstruct the 3D poses. This means we have one branch that is text-to-motion and another branch that is an autoencoder for motion-to-motion

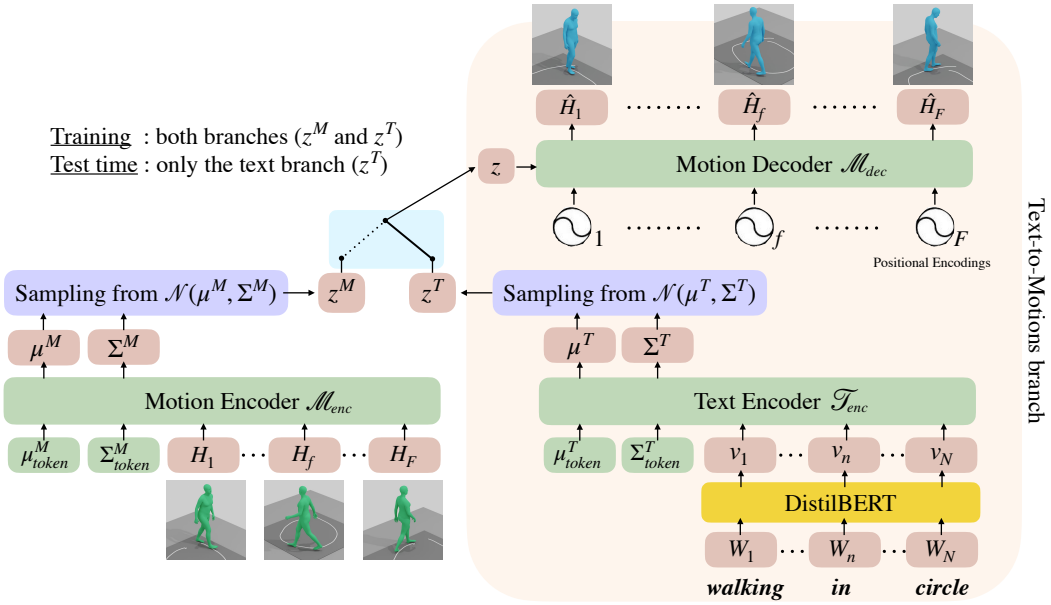


Figure 4.2: **Method overview:** During training, we encode both the motion and text through their respective Transformer encoders, together with modal-specific learnable distribution tokens. The encoder outputs corresponding to these tokens provide Gaussian distribution parameters on which the KL losses are applied and a latent vector z is sampled. Reconstruction losses on the motion decoder outputs further provide supervision for both motion and text branches. In practice, our word embedding consists of a variational encoder that takes input from a pre-trained and frozen DistilBERT [Sanh et al. 2019] model. Trainable layers are denoted in green, the inputs/outputs in brown. At test time, we only use the right branch, which goes from an input text to a diverse set of motions through the random sampling of the latent vector z^T on the cross-modal space. The output motion duration is determined by the number of positional encodings F.

(see Figure 4.2). At test time, we only use the text-to-motion branch. This approach proved effective in previous work [Ahuja et al. 2019]. Here, we first briefly describe the loss terms to train this model *probabilistically*. Then, we provide implementation details.

Given a ground-truth pair consisting of human motion $H_{1:F}$ and textual description $W_{1:N}$, we use (i) two reconstruction losses – one per modality, (ii) KL divergence losses comparing each modality against Gaussion priors, (iii) KL divergence losses, as well as a cross-modal embedding similarity loss to compare the two modalities to each other.

Reconstruction losses ($\mathcal{L}_\mathbf{R}$). We obtain $\widehat{H}^M_{1:F}$ and $\widehat{H}^T_{1:F}$ by inputting the motion embedding and text embedding to the decoder, respectively. We

compare these motion reconstructions to the ground-truth human motion $H_{1:F}$ via:

$$\mathcal{L}_{\text{R}} = \mathscr{L}_1(H_{1:F}, \widehat{H}^M_{1:F}) + \mathscr{L}_1(H_{1:F}, \widehat{H}^T_{1:F}) \tag{4.1}$$

where $\mathscr{L}_1$ denotes the smooth L1 loss.

KL losses ($\mathcal{L}_{\text{KL}}$). To enforce the two modalities to be close to each other in the latent space, we minimize the Kullback-Leibler (KL) divergences between the distributions of the text embedding $\phi^T = \mathcal{N}(\mu^T, \Sigma^T)$ and the motion embedding $\phi^M = \mathcal{N}(\mu^M, \Sigma^M)$. To regularize the shared latent space, we encourage each distribution to be similar to a normal distribution $\psi = \mathcal{N}(0, I)$ (as in standard VAE formulations). Thus we obtain four terms:

$$\begin{aligned}\mathcal{L}_{\text{KL}} = \text{KL}(\phi^T, \phi^M) + \text{KL}(\phi^M, \phi^T) \\ + \text{KL}(\phi^T, \psi) + \text{KL}(\phi^M, \psi).\end{aligned} \tag{4.2}$$

Cross-modal embedding similarity loss ($\mathcal{L}_{\text{E}}$). After sampling the text embedding $z^T \sim \mathcal{N}(\mu^T, \Sigma^T)$ and the motion embedding $z^M \sim \mathcal{N}(\mu^M, \Sigma^M)$ from the two encoders, we also constrain them to be as close as possible to each other, with the following loss term (i.e., loss between the cross-modal embeddings):

$$\mathcal{L}_{\text{E}} = \mathscr{L}_1(z^T, z^M). \tag{4.3}$$

The resulting total loss is defined as a weighted sum of the three terms: $\mathcal{L} = \mathcal{L}_{\text{R}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} + \lambda_{\text{E}}\mathcal{L}_{\text{E}}$. We empirically set λ_{KL} and λ_{E} to 10^{-5}, and provide ablations. While some of the loss terms may appear redundant, we experimentally validate each term.

Implementation details and hyperparameters. We train our models for 1000 epochs with the AdamW optimizer [Kingma et al. 2015; Loshchilov et al. 2019] using a fixed learning rate of 10^{-4}. Our minibatch size is set to 32. For all the encoders and the decoder of `TEMOS`, we set the embedding dimensionality to 256, the number of layers to 6, the number of heads in

multi-head attention to 4, the dropout rate to 0.1, and the dimension of the intermediate feedforward network to 1024 in the Transformers. Ablations of some of these hyperparameters are presented in the Section 4.3.4.

Runtime. Training our `TEMOS` model takes about 4.5 hours for 1K epochs, with a batch size of 32, on a single Tesla V100 GPU (16GB) using about 15GB GPU memory for training (i.e., *16 seconds* per epoch). In comparison, according to their paper, Ghosh et al. [Ghosh et al. 2021] trained their model for 350 epochs on a single Tesla V100 GPU in about 15 hours (i.e., *154 seconds* per epoch). While the rest of the hardware specifications or implementation efficiency may vary, given the same type of GPU for both methods, we can expect that our model trains an order of magnitude faster. This may be because our model generates the full motion with only one decoder pass. Previous work produces one frame at a time iteratively (i.e., the next frame has to wait for the previous one to be generated).

At training time, we input the full motion sequence, i.e., a variable number of frames for each training sample. At inference time, we can specify the desired duration F (see supplementary video on our webpage); however, for a fair evaluation with prior work we provide quantitative metrics with known ground-truth motion duration.

4.3 Experiments

We first present the data and performance measures used in our experiments (Section 4.3.1). Next, we compare to previous work (Section 4.3.2) and present an ablation study (Section 4.3.3). Then, we demonstrate our results with the SMPL model (Section 4.3.5). Finally, we discuss limitations (Section 4.3.6).

4.3.1 Data and evaluation metrics

KIT Motion-Language [Plappert et al. 2016] dataset (KIT) provides raw motion capture (MoCap) data, as well as processed data using the Master Motor Map (MMM) framework [Terlemez et al. 2014]. The motions comprise a

collection of subsets of the KIT Whole-Body Human Motion Database [Mandery et al. 2015] and of the [*CMU Graphics Lab Motion Capture Database* 2003]. The dataset consists of 3911 motion sequences with 6353 sequence-level description annotations, with 9.5 words per description on average. We use the same splits as in Language2Pose [Ahuja et al. 2019] by extracting 1784 training, 566 validation and 587 test motions (some motions do not have corresponding descriptions). As the model from [Ghosh et al. 2021] produce only 520 sequences in the test set (instead of 587), for a fair comparison we evaluate all methods with this subset, which we will refer to as the test set. If the same motion sequence corresponds to multiple descriptions, we randomly choose one of these descriptions at each training iteration, while we evaluate the method on the first description. Recent state-of-the-art methods on text-conditioned motion synthesis employ this dataset, by first converting the MMM axis-angle data into 21 xyz coordinates and downsampling the sequences from 100 Hz to 12.5 Hz. We do the same procedure, and follow the training and test splits explained above to compare methods. Additionally, we find correspondences from the KIT sequences to the AMASS MoCap collection [Mahmood et al. 2019] to obtain the motions in SMPL body format. We note that this procedure resulted in a subset of 2888 annotated motion sequences, as some sequences have not been processed in AMASS. We refer to this data as KIT_{SMPL}.

Statistics In the original KIT Motion-Language dataset [Plappert et al. 2016], there are 3911 motions and a total of 6352 text sequences (in which 900 motions are not annotated). Using a natural language processing parser, we extract "action phrases" from each sentence, based on verbs. For example, given the sentence "A human walks slowly", we automatically detect and lemmatize the verb, and attach complements to it, such that it becomes "walk slowly". With this procedure, we group sequences that correspond to the same action phrase and detect 4153 such action clusters out of 6352 sequences. The distribution of these clusters is very unbalanced: "walk forward" appears 596 times while there are 4030 actions that appear less than 10 times (3226 of them appear only once). On average, an action phrase appears 2.25 times.

Evaluation metrics. We follow the performance measures employed in Language2Pose [Ahuja et al. 2019] and Ghosh et al. [Ghosh et al. 2021] for quantitative evaluations. In particular, we report Average Positional Error

(APE) and Average Variance Error (AVE) metrics. However, we note that the results in [Ghosh et al. 2021] do not match the ones in [Ahuja et al. 2019] due to lack of evaluation code from [Ahuja et al. 2019]. We identified minor issues with the evaluation code of [Ghosh et al. 2021] (more details in the appendix of our paper [Petrovich et al. 2022]) therefore, we reimplement our own evaluation. Moreover, we introduce several modifications (which we believe make the metrics more interpretable): in contrast to [Ahuja et al. 2019; Ghosh et al. 2021], we compute the root joint metric by using the joint coordinates only (and not on velocities for x and y axes) and all the metrics are computed without standardizing the data (i.e., mean subtraction and division by standard deviation). Our motivation for this is to remain in the coordinate space since the metrics are *positional*. Note that the KIT data in the MMM format is canonicalized to the same body shape. We perform most of our experiments with this data format to remain comparable to the state of the art. We report results with the SMPL body format separately since the skeletons are not perfectly compatible. Finally, we convert our low-fps generations (at 12 Hz) to the original frame-rate of KIT (100 Hz) via linear interpolation on coordinates and report the error comparing to this original ground truth. We display the error in meters.

As discussed in Section 4.1, the evaluation is suboptimal because it assumes one ground truth motion per text; however, our focus is to generate multiple different motions. The KIT test set is insufficient to design distribution-based metrics such as FID, since there are not enough motions for the same text (see paragraph on statistics above). We therefore report the performance of generating a single sample, as well as generating multiple and evaluating the closest sample to the ground truth. We rely on additional perceptual studies to assess the correctness of multiple generations, which is described in the next section.

4.3.2 Comparison to the state of the art

Quantitative. We compare with the state-of-the-art text-conditioned motion generation methods [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021]

on the test set of the KIT dataset (as defined in 4.3.1). To obtain motions for these three methods, we use their publicly available codes (note that all three give the ground truth initial frame as input to their generations). We summarize the main results in Table 4.1. Our TEMOS approach substantially outperforms on all metrics, except APE on local joints. As pointed by [Ahuja et al. 2019; Ghosh et al. 2021], the most difficult metric that better differentiates improvements on this dataset is the APE on the root joint, and we obtain significant improvements on this metric. Moreover, we sample a random latent vector for reporting the results for TEMOS; however, as we will show next in Section 4.3.3, if we sample more, we are more likely to find the motion closer to the ground truth.

Qualitative. We further provide qualitative comparisons in Figure 4.4 with the state of the art. We show sample generations for Lin et al. [Lin et al. 2018a], JL2P [Ahuja et al. 2019], and Ghosh et al. [Ghosh et al. 2021]. The motions from our TEMOS model reflect the semantic content of the input text better than the others across a variety of samples. Furthermore, we observe that while [Lin et al. 2018a] generates overly smooth motions, JL2P has lots of foot sliding. [Ghosh et al. 2021], on the other hand, synthesizes unrealistic motions due to exaggerated foot contacts (and even extremely elongated limbs such as in 3rd column, 3rd row of Figure 4.4). Our generations are the most realistic among all. Further visualizations are provided in the supplementary video on our webpage.

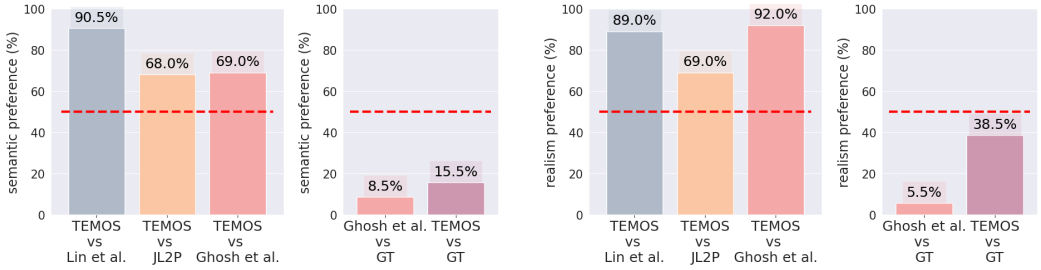
Perceptual study. These conclusions are further justified by two human perceptual studies that evaluate which methods are preferred in terms of semantics (correspondence to the text) or in terms of realism. We randomly sample 100 test descriptions and generate motion visualizations (as in the supplemental video) from all the three previous methods [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021], our method, and the ground truth (500 videos). From these visualizations, we create pairs of videos for each comparison, randomly swapping the left-right order of the video in each question (also 500 videos). For the semantic study, we display the text as well as the pair of videos simultaneously to Amazon Mechanical Turker (AMT) workers. The workers are asked to answer the question: "Which motion corresponds better to the textual description?". For the realism study, we use the same set of motion

Methods	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
Lin et. al. [Lin et al. 2018a]	1.966	1.956	0.105	1.969	0.790	0.789	0.007	0.791
JL2P [Ahuja et al. 2019]	1.622	1.616	**0.097**	1.630	0.669	0.669	0.006	0.672
[Ghosh et al. 2021]	1.291	1.242	0.206	1.294	0.564	0.548	0.024	0.563
TEMOS **(ours)**	**0.963**	**0.955**	0.104	**0.976**	**0.445**	**0.445**	**0.005**	**0.448**

Table 4.1: **State-of-the-art comparison:** We compare our method with recent works [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021], on the KIT Motion-Language dataset [Plappert et al. 2016] and obtain significant improvements on most metrics (values in meters) even if we are sampling a random motion per text conditioning for our model.

pairs and display them to other AMT workers but without showing the text description. They are asked to answer the question: "Which motion is more realistic?". For both studies, each worker answers a batch of questions, where the first 3 questions are discarded and used as a 'warmup' for the task. We further added 2 'catch trials' to detect unqualified workers, whose batch we discarded in our evaluation. We detected exactly 20 such workers out of 100 in both studies. Each pair of videos is shown to multiple workers between 2 and 5 (4 in average), from which we compute a majority vote to determine which generation is better than the other. If there is a tie, we assign a 0.5 equal score to both methods. The resulting percentage is computed over the 100 test descriptions.

The resulting ranking between our method and each of the state-of-the-art methods [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021] is reported in Figure 4.3. We see that humans perceive our motions as better matching the descriptions compared to all three state-of-the-art methods, especially significantly outperforming Lin et al. [Lin et al. 2018a] (users preferred TEMOS over [Lin et al. 2018a] 90.5% of the time). For the more competitive and more recent Ghosh et al. [Ghosh et al. 2021] method, we ask users to compare their generations against the ground truth. We do the same for our generations and see that users preferred our motions over the ground truth 15.5% of the time where the ones from Ghosh et al. [Ghosh et al. 2021] are preferred only 8.5% of the time. Our generations are also clearly preferred in terms of realism over the three methods. Our motions are realistic enough that they are preferred to real motion capture data 38.5% of the time, as compared to 5.5% of the time for Ghosh et al. [Ghosh et al. 2021].



(a) Which motion corresponds better to the textual description?

(b) Which motion is more realistic?

Figure 4.3: **Perceptual study:** (a) We ask users which motion corresponds better to the input text between two displayed samples generated from model A vs model B. (b) We ask other users which motion is more realistic without showing the textual description. We report the percentage for which the users show a preference for A. The red dashed line denotes the 50% level (equal preference). On the left of both studies, our generations from `TEMOS` were rated better than the previous work of Lin et al. [Lin et al. 2018a], JL2P [Ahuja et al. 2019], and Ghosh et al. [Ghosh et al. 2021]. On the right of both studies, we compare against the ground truth (GT) and see that our motions are rated as better than the GT 15.5% and 38.5% of the time, whereas Ghosh et al. [Ghosh et al. 2021] are at 8.5% and 5.5%.

4.3.3 Ablation study

In this section, we evaluate the influence of several components of our framework in a controlled setting.

Variational design. First, we 'turn off' the variational property of our generative model and synthesize a single motion per text. Instead of two learnable distribution tokens as in Figure 4.2, we use one learnable *embedding* token from which we directly obtain the latent vector using the corresponding encoder output (hence removing sampling). We removed all the KL losses such that the model becomes deterministic, and keep the embedding similarity loss to learn the joint latent space. In Table 4.2, we report performance metrics with this approach and see that we already obtain competitive performance with the deterministic version of our model, demonstrating the improvements from our temporal sequence modeling approach compared to previous works.

As noted earlier, our variational model can produce multiple generations for the same text, and a single random sample may not necessarily match the ground truth. In Table 4.2, we report results for one generation from a random z noise vector, or generating from the zero-vector that represents the mean of

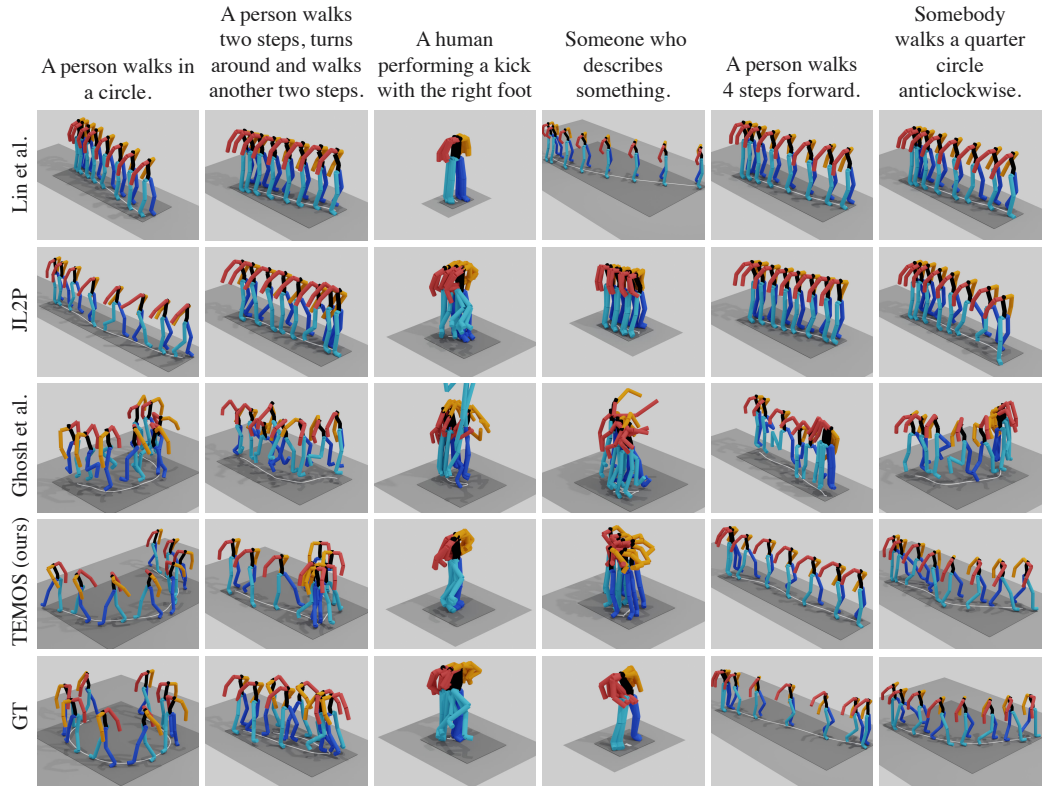


Figure 4.4: **Qualitative comparison to the state of the art:** We qualitatively compare the generations from our TEMOS model with the recent state-of-the-art methods and the ground truth (GT). We present different textual queries in columns, and different methods in rows. Overall, our generations better match semantically to the textual descriptions. We further overcome several limitations with the prior work, such as over-smooth motions in Lin et al. [Lin et al. 2018a], foot sliding in J2LP [Ahuja et al. 2019], and exaggerated foot contacts in Ghosh et al. [Ghosh et al. 2021], which can better be viewed in our supplementary video on our webpage.

the latent space ($z = \vec{0}$); both perform similarly. To assess the performance with multiple generations, we randomly sample 10 latent vectors per text, and provide two evaluations. First, we compare each of the 10 generations to the single ground truth, and average over all generations (10 random avg). Second, we record the performance of the motion that best matches to the ground truth out of the 10 generations (10 random best). As expected, Table 4.2 shows improvements with the latter. In Figure 4.5, we plot the reduction in APE root error as we increase the number of generations per text from 1 to 10, and observe a monotonic decrease as expected.

Furthermore, we measure the *worst case scenario*, where we generate 10 motions per text and record the error between the ground truth motion and

Model	Sampling	Average Positional Error ↓				Average Variance Error ↓			
		root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
Deterministic	n/a	1.175	1.165	0.106	1.188	0.514	0.513	0.005	0.516
Variational	1 sample, $z = \vec{0}$	1.005	0.997	0.104	1.020	0.443	0.442	0.005	0.446
Variational	1 random sample	0.963	0.955	0.104	0.976	0.445	0.445	0.005	0.448
Variational	10 random avg	1.001	0.993	0.104	1.015	0.451	0.451	0.005	0.454
Variational	10 random best	**0.784**	**0.774**	0.104	**0.802**	**0.392**	**0.391**	0.005	**0.395**

Table 4.2: **Variational vs deterministic models:** We first provide the performance of the deterministic version of our model. We then report results with several settings using our variational model: (i) generating a single motion per text to compare against the ground truth (either randomly or using a zero-vector representing the mean of the Gaussian latent space), and (ii) generating 10 motions per text, each compared against the ground truth separately (either averaging the metrics or taking the motion with the best metric). As expected, TEMOS is able to produce multiple hypotheses where the best candidates improve the metrics.

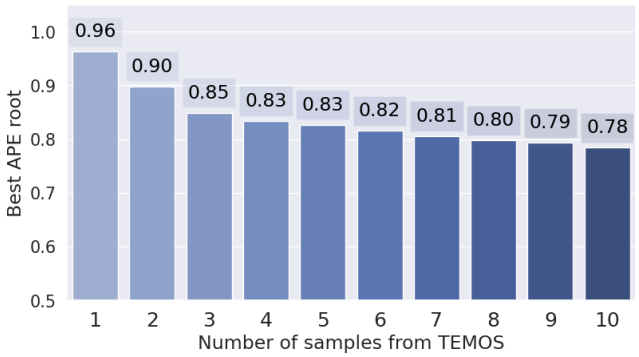


Figure 4.5: **Best APE root when sampling multiple generations:** Given a textual description, we generate multiple different motions, and select the motion that matches best to the ground truth sequence. We show that by sampling more generated sequences per text, we can reduce the APE root metric error.

the most different generated motion out of the 10. We obtain an APE of 1.24 (instead of 0.78 in the best case scenario, and 0.96 in the random scenario). Note that calling this worst case may not be accurate since the single ground truth motion does not represent the only possible motion, i.e., our generations may correspond well to the text without being close to the ground truth joints.

Architectural and loss components. Next, we investigate which component is most responsible for the performance improvement over the state of the art, since even the deterministic variant of our model outperforms previous works. Table 4.3 reports the performance by removing one component at each

Arch.	$\mathcal{L}_{KL}$	$\mathcal{L}_E$	Average Positional Error ↓ root joint	glob. traj.	mean loc.	mean glob.	Average Variance Error ↓ root joint	glob. traj.	mean loc.	mean glob.
GRU	$KL(\phi^T,\phi^M)+KL(\phi^M,\phi^T)+KL(\phi^T,\psi)+KL(\phi^M,\psi)$	✓	1.443	1.433	0.105	1.451	0.600	0.599	0.007	0.601
Transf.	$KL(\phi^T,\psi)$ **w/out** $\mathscr{M}_{enc}$	✗	1.178	1.168	0.106	1.189	0.506	0.505	0.006	0.508
Transf.	$KL(\phi^T,\phi^M)+KL(\phi^M,\phi^T)+KL(\phi^T,\psi)+KL(\phi^M,\psi)$	✗	1.091	1.083	0.107	1.104	0.449	0.448	0.005	0.451
Transf.	$KL(\phi^T,\psi)+KL(\phi^M,\psi)$ **w/out cross-modal KL losses**	✗	1.080	1.071	0.107	1.095	0.453	0.452	0.005	0.456
Transf.	$KL(\phi^T,\psi)+KL(\phi^M,\psi)$ **w/out cross-modal KL losses**	✓	0.993	0.983	0.105	1.006	0.461	0.460	0.005	0.463
Transf.	$KL(\phi^T,\phi^M)+KL(\phi^M,\phi^T)$ **w/out Gaussian priors**	✓	1.049	1.039	0.108	1.065	0.472	0.471	0.005	0.475
Transf.	$KL(\phi^T,\phi^M)+KL(\phi^M,\phi^T)+KL(\phi^T,\psi)+KL(\phi^M,\psi)$	✓	**0.963**	**0.955**	**0.104**	**0.976**	**0.445**	**0.445**	0.005	**0.448**

Table 4.3: **Architectural and loss study:** We conclude that the most critical component is the Transformer architecture, as opposed to a recurrent one (i.e., GRU). While the additional losses are helpful, they bring relatively minor improvements.

row. The APE root joint performance drops from 0.96 to i) 1.44 using GRUs instead of Transformers; ii) 1.18 without the motion encoder (using only one KL loss); iii) 1.09 without the cross-modal embedding loss; iv) 1.05 without the Gaussian priors; v) 0.99 without the cross-modal KL losses. Note that the cross-modal framework originates from JL2P [Ahuja et al. 2019]. While we observe slight improvement with each of the cross-modal terms, we notice that the model performance is already satisfactory even without the motion encoder. We therefore conclude that the main improvement stems from the improved non-autoregressive Transformer architecture, and removing each of the other components (4 KL loss terms, motion encoder, embedding similarity) also slightly degrades performance.

Language model finetuning. As explained in Section 4.2.2, we do not update the language model parameters during training, which are from the pretrained DistilBERT [Sanh et al. 2019]. We measure the performance with and without finetuning in Table 4.4 and conclude that freezing performs better while being more efficient. We note that we already introduce additional layers through our text encoder (see Figure 4.2), which may be sufficient to adapt the embeddings to our specific motion description domain.

4.3.4 Additional experiments

We conduct several experiments to explore the sensitivity of our model to certain hyperparameters. Our final set of hyperparameters was chosen using

LM params	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
Finetuned	1.402	1.393	0.113	1.414	0.559	0.558	0.006	0.562
Frozen	**0.963**	**0.955**	**0.104**	**0.976**	**0.445**	**0.445**	**0.005**	**0.448**

Table 4.4: **Language model finetuning:** We experiment with finetuning the language model (LM) parameters (i.e., DistilBERT [Sanh et al. 2019]) end-to-end with our motion-language cross-modal framework, and do not observe improvements. Here 'Frozen' refers to not updating the LM parameters.

the validation set (starting from a set of hyperparameters similar to ACTOR, in Chapter 3). Note that these hyperparameters did not always appear optimal on the test set. The following ablations provide a sense of robustness to different parameters. In particular, we show the effects of the batch size, $\{\lambda_{\text{KL}}, \lambda_{\text{E}}\}$ loss weighting parameters and the architecture parameters of Transformers. All other parameters, such as the learning rate (10^{-4}), the optimizer (AdamW) and the number of epochs (1000) are fixed. For all these experiments we use the skeleton-based MMM representation and the evaluation is based on a single random sample. We also experiment with various pretrained language models.

Batch size. In Table 4.5, we present results with batch sizes of 8, 16, 24 and 32. To handle variable-length training, we use padding and masking in the encoders and the decoder. To maintain reasonable memory consumption, we discard training samples that have more than 500 frames (after sub-sampling to 12.5 Hz): this corresponds to about 2.3% of the training data. We show that we obtain the best results by setting the batch size to 8 or 32. We set it to 32 in all other experiments as it takes less time to train.

Batch size	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
bs = 8	**0.950**	**0.941**	0.105	**0.965**	0.449	0.448	0.005	0.451
bs = 16	1.115	1.106	0.105	1.128	0.513	0.512	0.005	0.515
bs = 24	1.260	1.250	0.106	1.273	0.542	0.542	0.005	0.545
bs = 32	0.963	0.955	**0.104**	0.976	**0.445**	**0.445**	0.005	**0.448**

Table 4.5: **Batch size:** We see that the performance is the best for either a small batch size (=8) or a bigger batch size (=32). We were unable to use a higher batch size due to the GPU memory limit.

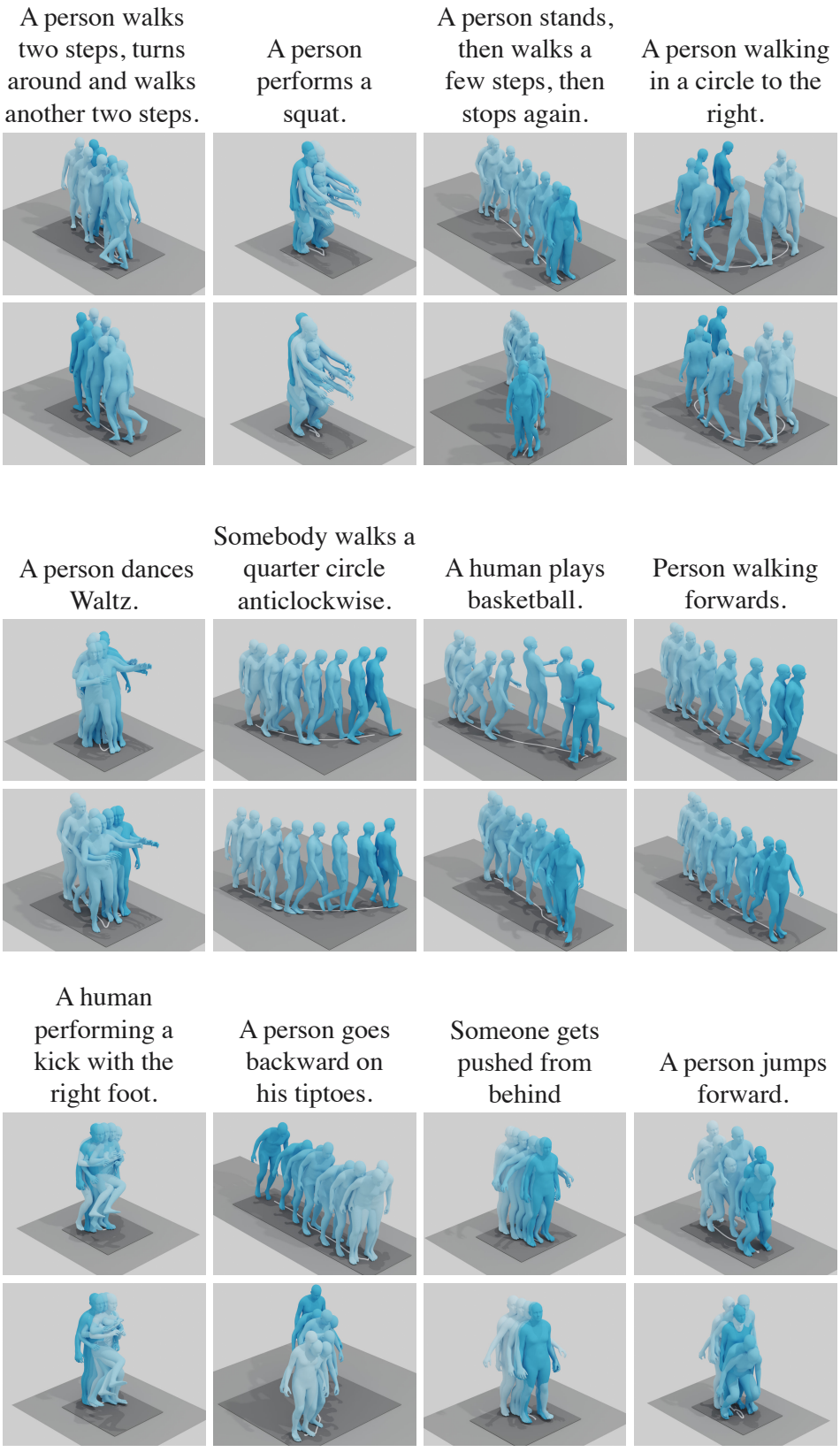


Figure 4.6: **Qualitative evaluation of the diversity:** We display two motion generations for each description. Our model shows certain diversity among different generations while respecting the textual description.

Losses weight	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
$\lambda_{KL} = \lambda_E = 10^{-3}$	1.219	1.210	0.111	1.230	0.555	0.554	0.006	0.556
$\lambda_{KL} = \lambda_E = 10^{-4}$	1.110	1.101	0.106	1.122	0.476	0.475	0.005	0.479
$\lambda_{KL} = \lambda_E = 10^{-5}$	**0.963**	**0.955**	**0.104**	**0.976**	**0.445**	**0.445**	0.005	**0.448**
$\lambda_{KL} = \lambda_E = 10^{-6}$	1.242	1.233	0.105	1.254	0.586	0.585	0.005	0.589
$\lambda_{KL} = \lambda_E = 10^{-7}$	1.034	1.025	0.108	1.049	0.488	0.487	0.005	0.491
$\lambda_{KL} = \lambda_E = 10^{-8}$	1.085	1.075	0.107	1.099	0.490	0.489	0.005	0.493
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-3}$	1.293	1.284	0.107	1.305	0.631	0.631	0.005	0.635
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-4}$	1.039	1.029	0.104	1.052	0.449	0.448	0.005	0.452
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-5}$	**0.963**	**0.955**	**0.104**	**0.976**	**0.445**	**0.445**	0.005	**0.448**
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-6}$	1.082	1.072	0.107	1.095	0.456	0.455	0.005	0.459
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-7}$	1.018	1.008	0.106	1.031	0.464	0.463	0.005	0.467
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-8}$	1.076	1.067	0.105	1.089	0.477	0.476	0.005	0.480
$\lambda_{KL} = 10^{-3}, \lambda_E = 10^{-5}$	1.145	1.135	0.111	1.157	0.507	0.506	0.006	0.509
$\lambda_{KL} = 10^{-4}, \lambda_E = 10^{-5}$	1.070	1.061	0.106	1.083	0.471	0.470	0.005	0.474
$\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-5}$	**0.963**	**0.955**	**0.104**	**0.976**	**0.445**	**0.445**	0.005	**0.448**
$\lambda_{KL} = 10^{-6}, \lambda_E = 10^{-5}$	0.971	0.962	0.105	0.986	0.455	0.454	0.005	0.457
$\lambda_{KL} = 10^{-7}, \lambda_E = 10^{-5}$	1.140	1.132	0.106	1.154	0.513	0.512	0.005	0.517
$\lambda_{KL} = 10^{-8}, \lambda_E = 10^{-5}$	1.025	1.015	0.107	1.039	0.461	0.460	0.005	0.464

Table 4.6: **Weight of the KL losses (λ_{KL}) and the embedding loss (λ_E):** The results are influenced more by changes in λ_E than in λ_{KL}, but otherwise if the values are not too low, the performances are similar. Note that the control row $\lambda_{KL} = 10^{-5}, \lambda_E = 10^{-5}$ is repeated in each block.

Weight of the KL losses and the embedding loss. In Table 4.6, we report results by varying both λ_{KL} and λ_E parameters (described in Section 4.2.3) from 10^{-3} to 10^{-8}. We show that overall the results are similar when λ_E is fixed. A too high value of 10^{-3} deteriorates the performance. We fix both of them to 10^{-5}.

Number of Transformer layers Here, we present two different experiments. The first experiment is to change globally the number of layers of all our Transformers. In Table 4.7, we see that the results are optimal when the number of layers is fixed to 6, which is used in all other experiments. Next, in Table 4.8, we experiment with a lighter model on top of the DistilBERT text encoder, by adding fewer layers and heads than 6. We see that 1 or 2 layers are not sufficient, but beginning with 4, the results are satisfactory. We use 6 layers in our model.

Pretrained language model In Table 4.9, we experiment with replacing DistilBERT [Sanh et al. 2019] with a larger pretrained language model. We compare with the original BERT [Devlin et al. 2019] model as well as the more

Hyperparameters	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
nlayers = 2	1.194	1.185	0.107	1.205	0.546	0.545	0.006	0.547
nlayers = 4	1.189	1.181	0.104	1.201	0.500	0.499	0.005	0.502
nlayers = 6	**0.963**	**0.955**	0.104	**0.976**	**0.445**	**0.445**	0.005	**0.448**

Table 4.7: **Number of layers in all Transformers:** While our results are slightly better for larger models, we observe that the performance is not very sensitive to changes in the number of layers in Transformers.

Hyperparameters	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
nlayers = 1	1.163	1.151	0.110	1.175	0.529	0.528	0.006	0.531
nlayers = 2	1.170	1.161	0.107	1.182	0.452	0.451	0.005	0.454
nlayers = 3	1.094	1.085	0.105	1.106	0.474	0.473	0.005	0.476
nlayers = 4	**0.916**	**0.908**	**0.104**	**0.930**	**0.440**	**0.440**	0.005	**0.444**
nlayers = 6	0.963	0.955	**0.104**	0.976	0.445	0.445	0.005	0.448

Table 4.8: **Number of layers in the Transformer of the text encoder only:** We fix the Transformer layers of the motion encoder and motion decoder to 6 (as in the other experiments), but we only change the number of layers of the *text encoder* (the one on top of DistilBert). The results suggest that training a light model on top of the language model still gives descent results, but adding more layers helps.

recent RoBERTa [Liu et al. 2019b] model. The results are similar and suggest that DistilBERT is sufficient for this task, while having fewer parameters.

Language model	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
DistilBERT [Sanh et al. 2019]	**0.963**	**0.955**	**0.104**	**0.976**	0.445	0.445	0.005	0.448
BERT [Devlin et al. 2019]	0.986	0.977	0.105	1.000	**0.441**	**0.441**	0.005	**0.444**
RoBERTa [Liu et al. 2019b]	1.066	1.056	0.107	1.079	0.492	0.491	0.005	0.494

Table 4.9: **Language model:** We experiment with language models larger than DistilBERT and do not observe significant changes in the performance.

4.3.5 Generating skinned motions

We evaluate the variant of our model which uses the parametric SMPL representation to generate full body meshes. To evaluate quantitatively our SMPL-based model, and obtaining results comparable with the MMM framework, we extract the most similar skeleton subset from the SMPL-H joints (provided by AMASS). The correspondence can be found in Table 4.10. Both in SMPL-H and MMM,

Type	Joints										
MMM	root	BP	BT	BLN	BUN	LS	LE	LW	RS	RE	RW
SMPL-H	pelvis	spine1	spine3	neck	head	left_shoulder	left_elbow	left_wrist	right_shoulder	right_elbow	right_wrist
MMM	LH	LK	LA	LMrot	LF	RH	RK	RA	RMrot	RF	
SMPL-H	left_hip	left_knee	left_ankle	left_heel	left_foot	right_hip	right_knee	right_ankle	right_heel	right_foot	

Table 4.10: **Correspondence** between the SMPL-H joints and the MMM framework joints.

the bodies are canonicalized with a standard body shape (robot-style for MMM, and average neutral body for SMPL-H). We further rescale the SMPL joints with a factor of 0.64, to match them with MMM joints. We evaluate the model with and without this rescaling. The performance metrics on KIT$_{SMPL}$ test set can be found in Table 4.11. Performance is comparable to that of MMM-based training when evaluating with rescaled joints. We expect future work to compare against the non-rescaled version when employing the SMPL model to interpret the metrics with respect to real human sizes. We provide qualitative examples in Figure 4.6 to illustrate the diversity of our generations for a given text (more examples can be found in the supplementary video on our webpage). For each text, we present 2 random samples. Each column shows a different text input. For all the visualization renderings in this chapter, the camera is fixed and the bodies are sampled evenly across time. Moreover, the forward direction of the first frame is always facing the same canonical direction. Our observation is that the model can generate multiple plausible motions corresponding to the same text, exploring the degrees of freedom remaining from ambiguities in the language description. On the other hand, if the text describes a precise action, such as "A person performs a squat" the diversity is reduced. The results are better seen as movies; see supplementary video where we also display other effects such as generating variable durations, and interpolating in the latent space.

4.3.6 Limitations

Our model has several limitations. Firstly, the vocabulary of the KIT data is relatively small with 1263 unique words compared to the full open-vocabulary setting of natural language, and are dominated by locomotive motions. We therefore expect our model to suffer from out-of-distribution descriptions.

		Average Positional Error ↓				Average Variance Error ↓			
		root	glob.	mean	mean	root	glob.	mean	mean
Dataset	rescaled	joint	traj.	loc.	glob.	joint	traj.	loc.	glob.
KIT_{SMPL}	✓	0.698	0.689	0.091	0.712	0.157	0.157	0.004	0.161
KIT_{SMPL}	✗	1.097	1.077	0.169	1.118	0.384	0.383	0.009	0.393

Table 4.11: **Our results with SMPL model:** We evaluate our model trained on the SMPL data against the ground truth of the test set of KIT_{SMPL} (joints extracted from the AMASS dataset). The first row shows results after a rescaling (to get skeletons closer to MMM processed joints). The second row shows results with the same set of joints but without the final rescaling. Both results are in meters.

Moreover, we do not have a principled way of measuring the diversity of our models since the training does not include multiple motions for the exact same text. Secondly, we notice that if the input text contains typos (e.g., 'wals' instead of 'walks'), `TEMOS` might drastically fail, suggesting that a preprocessing step to correct them beforehand might be needed. Finally, our method cannot scale up to very long motions (such as walking for several minutes) due to the quadratic memory cost.

Video. We provide a supplemental video, available on our website, which we encourage the reader to watch since motion is critical in our results, and this is hard to convey in a static document. In the video, we illustrate: (i) comparison with previous work, (ii) training with skeleton versus SMPL data, (iii) diversity of our model, (iv) generation of variable size sequences, (v) interpolation between two texts in our latent space, and (vi) failure cases.

4.4 Conclusion

In this chapter, we introduced a variational approach to generate diverse 3D human motions given textual descriptions in the form of natural language. In contrast to previous methods, our approach considers the intrinsically ambiguous nature of language and generates multiple plausible motions respecting the textual description, rather than deterministically producing only one. We obtain

state-of-the-art results on the widely used KIT Motion-Language benchmark, outperforming prior work by a large margin both in quantitative experiments and perceptual studies. Our improvements are mainly from the architecture design of incorporating sequence modeling via Transformers. Furthermore, we employ full body meshes instead of only skeletons.

This chapter introduces a common latent space between text and motion: we can now compare text and motion in the same space. This property will be tested in the next chapter, for the task of text-to-motion retrieval.

Chapter 5

Text-to-Motion Retrieval Using Contrastive 3D Human Motion Synthesis

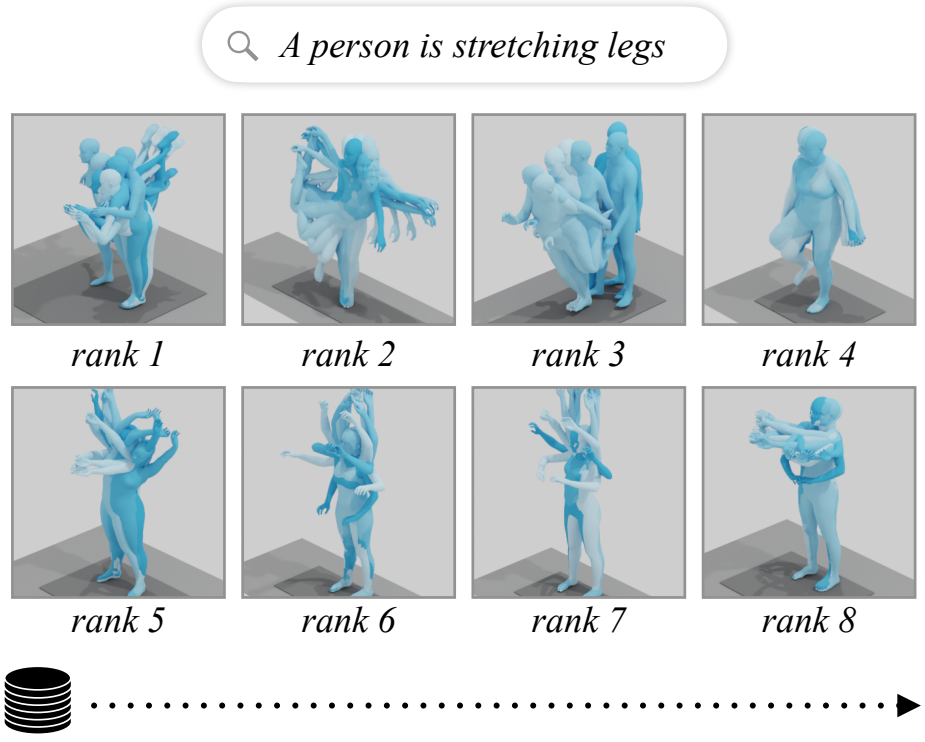


Figure 5.1: **Text-to-Motion Retrieval (TMR):** We illustrate the task of text-based motion retrieval where the goal is to rank a gallery of motions according to their similarity to the given query in the form of a natural language description.

This chapter presents our contribution on the task of 3D human motion retrieval from text.

We present TMR, a simple yet effective approach for text to 3D human motion retrieval. While previous work has only treated retrieval as a proxy evaluation metric, we tackle it as a standalone task. Our method extends the state-of-the-art text-to-motion synthesis model TEMOS, and incorporates a contrastive loss to better structure the cross-modal latent space. We show that maintaining the motion generation loss, along with the contrastive training, is crucial to obtain good performance. We introduce a benchmark for evaluation and provide an in-depth analysis by reporting results on several protocols. Our extensive experiments on the KIT-ML and HumanML3D datasets show that TMR outperforms the prior work by a significant margin, for example reducing the median rank from 54 to 19. Finally, we showcase the potential of our approach on moment retrieval. Our code, models and demo are publicly available on our website https://mathis.petrovich.fr/tmr.

5.1 Introduction

The language of movement cannot be translated into words.

Barbara Mettler

We ask the question whether a cross-modal space exists between 3D human motions and language. Our goal is to retrieve the most relevant 3D human motion from a gallery, given a natural language query that describes the desired motion (as illustrated in Figure 5.1). While text-to-image retrieval is a well-established problem within the broader vision & language field [Radford et al. 2021], there has been less focus on the related task of *text-to-motion* retrieval. Searching an existing motion capture dataset based on text input can often serve as a viable alternative to text-to-human-motion synthesis in many applications, while also providing the added benefit that the retrieved motion is guaranteed to be realistic. Additionally, once a cross-modal embedding is built to map text and motions into a joint representation space, both of the symmetrical

text-to-motion and *motion-to-text* tasks can be performed. Such a retrieval-based solution has a range of applications, including automatically indexing large motion capture collections, and helping to initialize the cumbersome text labeling process, by assigning the nearest text to each motion.

Let us first differentiate text-to-motion *retrieval* from text-to-motion *synthesis*. Motion synthesis (Chapter 4, [Chen et al. 2023; Tevet et al. 2023]) involves generating *new* data samples that go beyond the existing training set, while motion retrieval searches through existing motion capture collections. For certain applications, reusing motions from a collection may be sufficient, provided the collection is large enough to contain what the user is searching for. Unlike generative models for motion synthesis, which struggle to produce physically plausible, realistic sequences (previous Chapters 3, 4), a retrieval model has the advantage that it always returns a realistic motion. With this motivation, we pose the problem as a nearest neighbor search through a cross-modal text-motion space.

Early work performs search through motion databases to build motion graphs [Kovar et al. 2002; Arikan et al. 2003] by finding paths between existing motions and synthesizing new motions by stitching motions together with generated transitions. If the motion database is labeled with actions, the user can specify a series of actions to combine [Arikan et al. 2003]. In contrast, our search database is *not* labeled with text. *Motion matching* [Büttner et al. 2015], on the other hand, seeks to find the animation that best fits the current motion by searching a database of animations, doing motion-to-motion retrieval [Sidenbladh et al. 2002]. Our framework fundamentally differs from these lines of work in that our task is multi-modal, i.e., user query is text, which is compared against motions. The most similar work to ours is the very recent model from Guo et al. [Guo et al. 2022a], which trains for a joint embedding space between the two modalities. This model is only used to provide a performance measure for motion synthesis tasks, by querying a generated motion within a gallery of 32 descriptions (i.e., motion-to-text retrieval), and counting how many times the correct text is retrieved[1]. While this can be considered as the first text-motion retrieval model in the literature, its main limitation is the low performance, in

[1]While the paper [Guo et al. 2022a] describes a motion-to-text retrieval metric, we notice that the provided code performs text-to-motion retrieval.

particular when the gallery contains fine-grained descriptions. We substantially improve over [Guo et al. 2022a], by incorporating a joint synthesis & retrieval framework, as well as a more powerful contrastive training [Oord et al. 2018].

We get inspiration from image-text models such as BLIP [Li et al. 2022a] and CoCa [Yu et al. 2022], which formulate a multi-task objective. Besides the standard dual-encoder matching (such as CLIP [Radford et al. 2021] with two unimodal encoders for image and text), [Li et al. 2022a; Yu et al. 2022] also employ a text synthesis branch, performing image captioning. Such a generative capability potentially helps the model go beyond 'bag-of-words' understanding of vision-language concepts, observed for the naive contrastive models [Yuksekgonul et al. 2022; Doveh et al. 2023]. In our case, we depart from `TEMOS` (Chapter 4) which already has a synthesis branch to generate motions from text. We incorporate a cross-modal contrastive loss (i.e., InfoNCE [Oord et al. 2018]) in this framework to jointly train text-to-motion synthesis and text-to-motion retrieval tasks. We empirically demonstrate significant improvements with this approach when ablating the importance of each task. We also compare to the method of [Guo et al. 2022a], whose motion-to-text retrieval model is adopted for measuring text-to-motion generation performance automatically by other works [Zhang et al. 2022a; Dabral et al. 2023; Tevet et al. 2023].

Text-motion data differs from its text-image counterparts particularly due to the nature of motion descriptions. In fact, for an off-the-shelf large language model, sentences describing different motions tend to be similar, since they fall within the same topic of human motions. For example, the text-text cosine similarities [Song et al. 2020] after encoding motion descriptions from the KIT training set [Plappert et al. 2016] are on average 0.71 on a scale between [0, 1], while this value is 0.56 (almost orthogonal) on a random subset of LAION [Schuhmann et al. 2021] image descriptions with the same size. This poses several challenges. Typical motion datasets [Plappert et al. 2016; Punnakkal et al. 2021; Guo et al. 2022a] contain similar motions with different accompanying texts, e.g., 'person walks', 'human walks', as well as similar texts with different motions, e.g., 'walk backwards', 'walk forwards'. With naive contrastive training [Oord et al. 2018], one would make all samples within a batch as negatives, except the corresponding label for a given anchor. In this work, we take into account the fact that there are potentially significant

similarities between pairs within a batch. Specifically, we discard pairs that have a text-text similarity in their labels that is above a certain threshold. Such careful negative sampling leads to performance improvements.

In this chapter, we illustrate an additional use case for our retrieval model – zero-shot temporal localization – and highlight this task as a potential future avenue for research. Similar to temporal localization in videos with natural language queries [Regneri et al. 2013; Gao et al. 2017; Hendricks et al. 2017; Escorcia et al. 2019; Lei et al. 2020], also referred to as "moment retrieval", we showcase the grounding capability of our model by directly applying it on long motion sequences to retrieve corresponding moments. We illustrate results on the BABEL dataset [Punnakkal et al. 2021], which typically contains a series of text annotations for each motion sequence. Note that the task is zero-shot, because the model has not been trained for localization, and at the same time has not seen BABEL labels, which come from a different domain (e.g., typically action-like descriptions instead of full sentences).

Our contributions are the following: (i) We address the little-studied problem of text-to-motion retrieval, and introduce a series of evaluation benchmarks with varying difficulty. (ii) We propose a joint synthesis and retrieval framework, as well as negative filtering, and obtain state-of-the-art performance on text-motion retrieval. (iii) We provide extensive experiments to analyze the effects of each component in controlled settings.

5.2 Method

In this section, we introduce the task and the terminology associated with text-to-motion retrieval (Section 5.2.1). Next, we present our model, named `TMR`, and its training protocol (Section 5.2.2). We then explain our simple approach for identifying and filtering incorrect negatives (Section 5.2.3), followed by a discussion of the implementation details (Section 5.2.4).

5.2.1 Definitions

Given a natural language query T, such as '*A person walks and then makes a right turn.*', the goal is to rank the motions from a database (i.e., the gallery) according to their semantic correspondence with the text query, and to retrieve the motion that matches best to the textual description. In other words, the task involves sorting the database so that the top ranked motions are the most relevant matches, i.e., creating a search engine to index motions. Additionally, we define the symmetric (and complementary) task, namely motion-to-text retrieval, where the aim is to retrieve the most suitable caption that matches a given motion from a database of texts.

3D human motion refers to a sequence of human poses. The task does not impose any limitations on the type of representation used, such as joint positions, rotations, or parametric models such as SMPL [Loper et al. 2015]. As detailed in Section 5.3.1, we choose to use the representation employed by [Guo et al. 2022a] to facilitate comparisons with previous work. While our experiments are based on the SMPL skeleton, our method is applicable to any skeleton topology. One could alternatively use retargetting [Villegas et al. 2018; Aberman et al. 2020; Wang et al. 2020] to adapt to other body representations.

Text description refers to a sequence of words describing the action performed by a human in natural language. We do not restrict the format of the motion description. The text can be simply an action name (e.g., 'walk') or a full sentence (e.g., 'a human is walking'). The sentences can be fine-grained (e.g., 'a human is walking in a circle slowly'), and may contain one or several actions, simultaneously (e.g., 'walking while waving') or sequentially (e.g., 'walking then sitting').

5.2.2 Joint training of retrieval and synthesis

We introduce `TMR`, which extends the Transformer-based text-to-motion synthesis model `TEMOS` (Chapter 4) by incorporating additional losses to make it suitable for the retrieval task. The architecture consists of two independent encoders for inputting motion and text, as well as a decoder that outputs

motion (see Figure 5.2 for an overview). In the following, we review TEMOS components and our added contrastive training.

Dual encoders. One approach to solving cross-modal retrieval tasks involves defining a similarity function between the two modalities. In our case, the two modalities are text and motion. The similarity function can be applied to compare a given query with each element in the database, and the maximum value would indicate the best match. In this chapter, we follow the approach taken by previous work on metric learning, such as CLIP [Radford et al. 2021], by defining one encoder for each modality and then computing the cosine similarity between their respective embeddings. Such dual embedding has the advantage of fast inference time since the gallery embeddings can be computed and stored beforehand [Miech et al. 2021].

Our model is built upon the components of TEMOS (Chapter 4), which already provides a motion encoder and a text encoder, mapping them to a joint space (building on the idea from Language2Pose [Ahuja et al. 2019]); this serves as a strong baseline for our work. Both motion and text encoders are Transformer encoders [Vaswani et al. 2017] with additional learnable distribution parameters, as in the VAE-based ACTOR (Chapter 3). They are probabilistic in nature, outputting parameters of a Gaussian distribution (μ and Σ) from which a latent vector $z \in \mathbb{R}^d$ can be sampled. While the text encoder takes text features from a pre-trained and frozen DistilBERT [Sanh et al. 2019] network as input, the motion sequence is fed directly in the motion encoder. Note that when performing retrieval, we directly use the output embedding that corresponds to the mean token (μ^M for motion, μ^T for text).

Motion decoder. TEMOS is trained for the task of motion synthesis and comes equipped with a motion decoder branch. This decoder is identical to the one used in ACTOR (Chapter 3), which supports variable-duration generation. More specifically, it takes a latent vector $z \in \mathbb{R}^d$ and a sinusoidal positional encoding as input, and generates a motion non-autoregressively through a single forward pass. We show in Section 5.3.2 that keeping this branch helps improve the results.

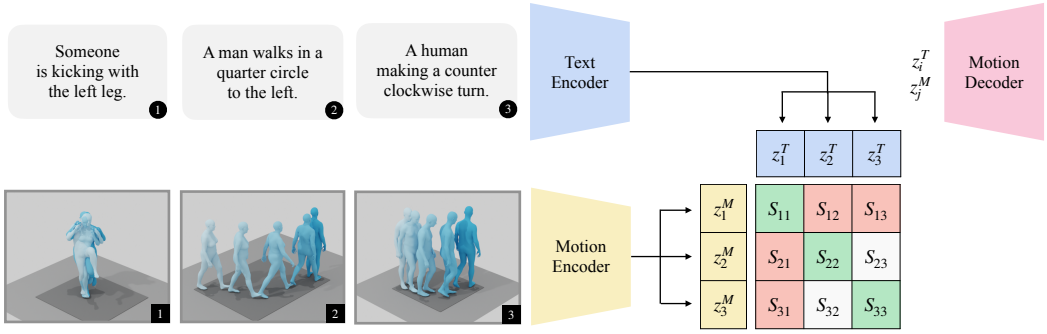


Figure 5.2: **Joint motion retrieval and synthesis:** A simplified view of our TMR framework is presented, where we focus on the similarity matrix defined between text-motion pairs within a batch. Here, we show a batch of 3 samples for illustration purposes. The goal of the contrastive objective is to maximize the diagonal denoting positive pairs (green), and to minimize the off-diagonal negative pair items that have text similarity below a threshold (red). In this example, remaining similarities S_{23} and S_{32} are discarded from the loss computation because there is high text similarity between T_2 and T_3. The rest of the model remains similar to TEMOS (Chapter 4, which decodes a motion from both text z^{T_i} and motion z^{M_j} latent vectors. See text for further details.

TEMOS losses. We keep the same base set of losses from Chapter 4, defined as the weighted sum $\mathcal{L}_{\texttt{TEMOS}} = \mathcal{L}_{\text{R}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} + \lambda_{\text{E}}\mathcal{L}_{\text{E}}$. In summary, a reconstruction loss term $\mathcal{L}_{\text{R}}$ measures the motion reconstruction given text or motion input (via a smooth L1 loss). A Kullback-Leibler (KL) divergence loss term $\mathcal{L}_{\text{KL}}$ is composed of four losses: two of them to regularize each encoded distributions — $\mathcal{N}(\mu^M, \Sigma^M)$ for motion and $\mathcal{N}(\mu^T, \Sigma^T)$ for text — to come from a normal distribution $\mathcal{N}(0, I)$. The other two enforce distribution similarity between the two modalities. A cross-modal embedding similarity loss $\mathcal{L}_{\text{E}}$ enforces both text z^T and motion z^M latent codes to be similar to each other (with a smooth L1 loss). We set λ_{KL} and λ_{E} to 10^{-5} in our experiments as in Chapter 4.

Contrastive training. While TEMOS has a cross-modal embedding space, its major drawback to be usable as an effective retrieval model is that it is never trained with negatives, but only positive motion-text pairs. To overcome this limitation, we incorporate a contrastive training with the usage of negative samples to better structure the latent space. Given a batch of N (positive) pairs of latent codes $(z_1^T, z_1^M), \cdots, (z_N^T, z_N^M)$, we define any pair (z_i^T, z_j^M) with $i \neq j$ as negative. The similarity matrix S computes the pairwise cosine similarities for all pairs in the batch $S_{ij} = \cos(z_i^T, z_j^M)$. In contrast to [Guo et al. 2022a], who consider one random negative per batch (and use a margin loss), we adopt

the more recent formulation of InfoNCE [Oord et al. 2018], which was proven effective in other work [Bain et al. 2021; Radford et al. 2021; Li et al. 2022a]. This loss term can be defined as follows:

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{2N}\sum_{i}\left(\log\frac{\exp S_{ii}/\tau}{\sum_j \exp S_{ij}/\tau} + \log\frac{\exp S_{ii}/\tau}{\sum_j \exp S_{ji}/\tau}\right), \tag{5.1}$$

where τ is the temperature hyperparameter.

Training loss. The total loss we use to train `TMR` is the weighted sum $\mathcal{L}_{\texttt{TEMOS}} + \lambda_{\text{NCE}}\mathcal{L}_{\text{NCE}}$ where λ_{NCE} controls the importance of the contrastive loss.

5.2.3 Filtering negatives

As mentioned in Section 5.1, the descriptions accompanying motion capture collections can be repetitive or similar across the training motions. We wish to prevent defining negatives between text-motion pairs that contain similar descriptions.

Consider for example the two text descriptions, "*A human making a counterclockwise turn*" and "*A person walks quarter a circle to the left*". In the KIT-ML benchmark [Plappert et al. 2016], these two descriptions appear as two different annotations for the same motion. Due to the flexibility and ambiguity of natural language, different words may describe the same concepts (e.g., 'counterclockwise', 'circle to the left').

During training, the random selection of batches can adversely affect the results because the model may have to push away two latent vectors that correspond to similar meanings. This can force the network to focus on unimportant details (e.g., 'someone' vs 'human'), ultimately resulting in decreased performance due to unstable behavior and reduced robustness to text variations.

To alleviate this issue, we leverage an external large language model to provide sentence similarity scores. In particular, we use MPNet [Song et al. 2020] to encode sentences and compute similarities between two text descriptions. We then determine whether to filter a pair of text descriptions (t_1, t_2) if their similarity is higher than a certain threshold, referring them as 'wrong negatives'.

During training, we filter wrong negative pairs from the loss computation. Note that we refrain from defining them as positives, as the language model may also incorrectly say two descriptions are similar when they are not.

5.2.4 Implementation details

We use the AdamW optimizer [Loshchilov et al. 2019] with a learning rate of 10^{-4} and a batch size of 32. Since the batch size can be an important hyperparameter for the InfoNCE loss, due to determining the number of negatives, we report experimental results with different values. The latent dimensionality of the embeddings is $d = 256$. We set the temperature τ to 0.1, and the weight of the contrastive loss term λ_{NCE} to 0.1. The threshold to filter negatives is set to 0.8. We provide experimental analyses to measure the sensitivity to these added hyperparameters.

5.3 Experiments

We start by describing the datasets and evaluation protocol used in the experiments (Section 5.3.1). We then report the performance of our model on our new retrieval benchmark along with comparison to prior work (Section 5.3.2). Next, we present our ablation study measuring the effects of the additional contrastive loss, the negative filtering, and the hyperparameters (Section 5.3.3). Then, we provide qualitative results for retrieval (Section 5.3.4), and show its potential for motion generation (Section 5.3.5). Finally, we present the use case of moment retrieval (Section 5.3.6).

5.3.1 Datasets and evaluation

HumanML3D dataset (H3D) [**Guo et al. 2022a**]. provides natural language labels to describe the motions in AMASS [Mahmood et al. 2019] and HumanAct12 [Guo et al. 2020] motion capture collections. We follow the motion pre-processing procedure of [Guo et al. 2022a], and apply the SMPL

Protocol	Methods	Text-motion retrieval						Motion-text retrieval					
		R@1 ↑	R@2 ↑	R@3 ↑	R@5 ↑	R@10 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	R@5 ↑	R@10 ↑	MedR ↓
(a) All	TEMOS	2.12	4.09	5.87	8.26	13.52	173.0	3.86	4.54	6.94	9.38	14.00	183.25
	[Guo et al. 2022a]	1.80	3.42	4.79	7.12	12.47	81.00	2.92	3.74	6.00	8.36	12.95	81.50
	TMR	**5.68**	**10.59**	**14.04**	**20.34**	**30.94**	**28.00**	**9.95**	**12.44**	**17.95**	**23.56**	**32.69**	**28.50**
(b) All with threshold	TEMOS	5.21	8.22	11.14	15.09	22.12	79.00	5.48	6.19	9.00	12.01	17.10	129.0
	[Guo et al. 2022a]	5.30	7.83	10.75	14.59	22.51	54.00	4.95	5.68	8.93	11.64	16.94	69.50
	TMR	**11.60**	**15.39**	**20.50**	**27.72**	**38.52**	**19.00**	**13.20**	**15.73**	**22.03**	**27.65**	**37.63**	**21.50**
(c) Dissimilar subset	TEMOS	33.00	42.00	49.00	57.00	66.00	4.00	35.00	44.00	50.00	56.00	70.00	3.50
	[Guo et al. 2022a]	34.00	48.00	57.00	72.00	84.00	3.00	34.00	47.00	59.00	72.00	83.00	3.00
	TMR	**47.00**	**61.00**	**71.00**	**80.00**	**86.00**	**2.00**	**48.00**	**63.00**	**69.00**	**80.00**	**84.00**	**2.00**
(d) Small batches	TEMOS)	40.49	53.52	61.14	70.96	84.15	2.33	39.96	53.49	61.79	72.40	85.89	2.33
	[Guo et al. 2022a]	52.48	71.05	80.65	89.66	**96.58**	1.39	52.00	71.21	81.11	89.87	**96.78**	1.38
	TMR	**67.16**	**81.32**	**86.81**	**91.43**	95.36	**1.04**	**67.97**	**81.20**	**86.35**	**91.70**	95.27	**1.03**

Table 5.1: **Text-to-motion retrieval benchmark on HumanML3D:** We establish four evaluation protocols as described in Section 5.3.1, with decreasing difficulty from (a) to (d). Our model TMR substantially outperforms the prior work of [Guo et al. 2022a] and TEMOS (Chapter 4), on the challenging H3D dataset.

Protocol	Methods	Text-motion retrieval						Motion-text retrieval					
		R@1 ↑	R@2 ↑	R@3 ↑	R@5 ↑	R@10 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	R@5 ↑	R@10 ↑	MedR ↓
(a) All	TEMOS	7.11	13.25	17.59	24.10	35.66	24.00	**11.69**	**15.30**	**20.12**	26.63	36.39	26.50
	[Guo et al. 2022a]	3.37	6.99	10.84	16.87	27.71	28.00	4.94	6.51	10.72	16.14	25.30	28.50
	TMR	**7.23**	**13.98**	**20.36**	**28.31**	**40.12**	**17.00**	11.20	13.86	**20.12**	**28.07**	**38.55**	**18.00**
(b) All with threshold	TEMOS	18.55	24.34	30.84	42.29	56.39	7.00	17.71	22.41	28.80	35.42	47.11	13.25
	[Guo et al. 2022a]	13.25	22.65	29.76	39.04	49.52	11.00	10.48	13.98	20.48	27.95	38.55	17.25
	TMR	**24.58**	**30.24**	**41.93**	**50.48**	**60.36**	**5.00**	**19.64**	**23.73**	**32.53**	**41.20**	**53.01**	**9.50**
(c) Dissimilar subset	TEMOS	24.00	40.00	46.00	54.00	70.00	5.00	33.00	39.00	45.00	49.00	64.00	6.50
	[Guo et al. 2022a]	16.00	29.00	36.00	48.00	66.00	6.00	24.00	29.00	36.00	46.00	66.00	7.00
	TMR	**26.00**	**46.00**	**60.00**	**70.00**	**83.00**	**3.00**	**34.00**	**45.00**	**60.00**	**69.00**	**82.00**	**3.50**
(d) Small batches	TEMOS	43.88	58.25	67.00	74.00	84.75	2.06	41.88	55.88	65.62	75.25	85.75	2.25
	[Guo et al. 2022a]	42.25	62.62	75.12	87.50	**96.12**	1.88	39.75	62.75	73.62	86.88	**95.88**	1.95
	TMR	**49.25**	**69.75**	**78.25**	**87.88**	95.00	**1.50**	**50.12**	**67.12**	**76.88**	**88.88**	94.75	**1.53**

Table 5.2: **Text-to-motion retrieval benchmark on KIT-ML:** As in Table 5.1, we report the four evaluation protocols, this time on the KIT dataset. Again, TMR significantly improves over [Guo et al. 2022a] and TEMOS (Chapter 4) across all protocols and metrics.

layer [Loper et al. 2015] to extract joint positions, canonicalize the skeletons to share the same topology (i.e., same bone lengths), then compute motion features (extracting local positions, velocities and foot contacts similar to Holden et al. [Holden et al. 2017]). The data is then augmented by mirroring left and right (both in motions and their corresponding texts). After this procedure, and following the official split, we obtain 23384, 1460, 4380 motions for the training, validation, and test sets, respectively. On average, each motion is annotated 3.0 times with different text. During training we randomly select one as the matching text, for testing we use the first text.

KIT Motion-Language dataset (KIT) [**Plappert et al. 2016**] also come from motion capture data, with an emphasis on locomotion motions. It originally consists of 3911 motion sequences and 6278 text sentences. We pre-process the motions with the same procedure as in H3D. The data is split into 4888, 300, 830 motions for training, validation, and test sets, respectively. In this dataset, each motion is annotated 2.1 times on average.

Evaluation protocol. We report standard retrieval performance measures: recall at several ranks, R@1, R@2, etc. for both text-to-motion and motion-to-text tasks (identical to the R Precision metrics adopted by [Guo et al. 2022a]). Recall at rank k measures the percentage of times the correct label is among the top k results; therefore higher is better. We additionally report median rank (MedR), where lower is better. Note that the *retrieval* performance is evaluated on a gallery of unseen real motions (i.e., test set). A study of the *synthesis* performance of `TMR` can be found in Section 5.3.5.

We define several evaluation protocols, mainly changing the gallery set.
(a) **All** the test set is used as a first protocol, without any modification. This set is partially problematic because there are repetitive texts across motions, or just minor differences (e.g., person vs human, walk vs walking).
(b) **All with threshold** means we search over all the test set, but, in this case, we accept a retrieved motion as correct if its text label is similar to the query text above a threshold. For example, retrieving the motion corresponding to *"A human walks forward"* should be correct when the input query is *"Someone is walking forward"*. We set a high threshold of 0.95 (scaled between [0, 1]) to remove very similar texts without removing too many fine-grained details (see

below statistics on how often similar text descriptions appear in the datasets).
(c) **Dissimilar subset** refers to sampling 100 motion-text pairs whose texts are maximally far from each other (using an approximation of the quadratic knapsack problem [Aïder et al. 2022]). This evaluation can be considered as an easy, but clean, subset of the previous ones.
(d) **Small batches** is included to mimic the protocol described by Guo et al. [Guo et al. 2022a], who randomly pick batches of 32 motion-text pairs and report the average performance. An ideal evaluation metric should not have randomness, and a gallery size of 32 is relatively easy compared to the previous protocols.

Number of similar text descriptions in the test set. As explain above, the evaluation protocol (b) marks retrieved motions as correct if their corresponding text is similar to the queried text above a threshold of 0.95 (note that this threshold is different from the one used in training). Here, we report the total number of pairs that are above this threshold for each dataset. For KIT, on the 830 sequences of the test set, there are $344,035$ unique pairs of texts ($830 * 829/2$) from which $2,467$ of them are similar (about 0.7% of the data). For H3D, on the $4,380$ sequences of the test set, there are $9,590,010$ unique pairs of texts ($4380 * 4380/2$) from which $6,017$ of them are similar (about 0.06% of the data).

5.3.2 A new benchmark & comparison to prior work

We present the performance of our model on this new retrieval benchmark, on H3D (Table 5.1) and KIT (Table 5.2) datasets, across all evaluation protocols. We also compare against prior work `TEMOS` (Chapter 4) and Guo et al. [Guo et al. 2022a]. For `TEMOS`, we retrain their model on both datasets to have a comparable benchmark since the original model differs in motion representation and lacks left/right data augmentation (and they only provide a KIT-pretrained model, not H3D). For [Guo et al. 2022a], we take their publicly available models trained on these two datasets.

`TEMOS` in particular is not designed to perform well on retrieval, since its cross-modal embedding space is only trained with positive pairs. However, Guo et al. train contrastively with negatives as well, using a margin loss [Hadsell

Motion **Recons.**	**InfoNCE**	**Margin**	Text-motion retrieval				Motion-text retrieval			
			R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
✗	✗	✓	15.06	22.17	25.78	12.00	8.19	11.57	16.39	19.50
✗	✓	✗	19.76	25.30	36.87	6.00	17.47	19.76	30.60	**9.50**
✓	✗	✗	18.55	24.34	30.84	7.00	17.71	22.41	28.80	13.25
✓	✗	✓	19.88	24.46	34.46	7.00	14.70	19.76	28.19	12.50
✓	✓	✗	**24.58**	**30.24**	**41.93**	**5.00**	**19.64**	**23.73**	**32.53**	**9.50**

Table 5.3: **Losses:** We experiment with various loss definitions (i) with/without the motion reconstruction, and (ii) the choice of the contrastive loss between InfoNCE and margin-based. We see that InfoNCE [Oord et al. 2018] is a better alternative to the contrastive loss with Euclidean margin [Hadsell et al. 2006] (employed by Guo et al. [Guo et al. 2022a]). The reconstruction loss through the motion decoder branch further boosts the results.

et al. 2006]. For all 4 evaluation sets with varying difficulties, TMR outperforms the prior work, suggesting our model better captures the finegrained nature of motion descriptions. The model of [Guo et al. 2022a] is adopted as part of motion synthesis evaluation in several works. TMR may therefore provide a better alternative. Our significant improvements over the state of the art can be credited to (i) jointly training for synthesis and retrieval, (ii) adopting the more recent contrastive objective, InfoNCE [Oord et al. 2018], while (iii) carefully eliminating wrong negatives. In the following, we ablate these components in controlled experiments.

5.3.3 Ablation study

The rest of the quantitative evaluation uses the '(b) All with threshold' evaluation protocol, on the KIT dataset.

Which losses matter? Table 5.3 compares several variants of TMR where we check (a) whether the jointly trained motion synthesis branch helps retrieval, and (b) how important the form of the contrastive loss is. When removing the synthesis branch and only using the the contrastive loss, we perform a deterministic encoding (i.e., with a single token instead of two tokens μ, σ). First, we see that the motion synthesis branch certainly improves results compared with only training using a contrastive loss (e.g., 41.93 vs 36.87 R@3).

Threshold	Text-motion retrieval				Motion-text retrieval			
	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
0.55	19.40	23.25	30.48	9.00	17.83	21.69	29.52	14.00
0.60	17.95	26.87	36.87	6.00	20.60	24.70	31.81	11.25
0.65	23.01	28.67	36.39	7.00	19.04	21.69	29.76	11.50
0.70	22.29	29.64	38.80	6.00	18.19	22.77	32.05	**9.00**
0.75	20.00	27.11	37.83	6.00	20.24	24.46	**34.22**	9.50
0.80	**24.58**	**30.24**	**41.93**	**5.00**	19.64	23.73	32.53	9.50
0.85	21.45	25.78	38.43	6.50	**20.84**	24.10	33.37	9.50
0.90	23.25	30.12	40.48	6.00	20.00	**25.18**	33.13	9.50
0.95	20.48	26.99	38.43	6.00	19.28	23.37	31.93	10.25
✗	22.17	27.83	36.02	7.00	16.75	21.33	32.17	11.50

Table 5.4: **Filtering negatives:** We compare several threshold values for filtering negatives from the loss comparison due to having similar texts. We observe that removing negatives based on text similarity above 0.8 (from a scale between [0,1]) performs well overall.

This possibly forces the latent vector to capture the full content of the input text (i.e., instead of picking up on a subset of words, or bag-of-words [Yuksekgonul et al. 2022; Doveh et al. 2023] upon finding a shortcut that satisfies the contrastive loss). Second, in the presence of a contrastive loss, the InfoNCE formulation is significantly better than the margin loss employed by previous work [Guo et al. 2022a] (41.93 vs 34.46 R@1). Note that for this experiment, we keep the same negative filtering for both the margin loss and InfoNCE (we provide additional experiments without the negative filtering in our original paper [Petrovich et al. 2023]).

The effect of filtering negatives. As explained in Section 5.2.3, during training we filter out pairs whose texts are closer than a threshold in an embedding space, and do not count them in the contrastive loss computation. Note that we still keep each item in the batch for the motion synthesis objective. In Table 5.4, we perform experiments with a range of different values for this threshold selection. On a scale between [0, 1], a threshold of 0.8 has the best results, balancing keeping a sufficient number of negatives and removing the wrong ones. Without filtering at all, the performance remains at 36.02 R@3 (compared to 41.93). In Table 5.5, we compute the amount of negatives that

Threshold	0.55	0.6	0.65	0.7	0.75	**0.8**	0.85	0.9	0.95
% filtered negatives	98.04	88.04	68.56	48.27	31.54	**17.29**	7.41	2.78	0.71

Batch size	16	**32**	64	128
% filtered negatives	17.02	**17.29**	16.96	17.28

Table 5.5: **Percentage of filtered negatives per batch in KIT:** We compute the average percentage of negative pairs per batch that are discarded from the loss computation due to text similarity. The percentage decreases with higher thresholds as expected (top), but the batch size does not have a significant impact (bottom).

are filtered on average per batch, depending on the threshold and the batch size. In our current setting, 17.29% of the negatives are discarded. We see that this rate remains similar across batch sizes.

Hyperparameters of the contrastive training. We show the sensitivity of our model to several hyperparameters added when extending `TEMOS`: (i) temperature τ of the cross entropy of InfoNCE [Oord et al. 2018] in Eq.5.1, (ii) the λ_{NCE} weighting parameter, (iii) the batch size, which determines the number of negatives and (iv) the latent dimensionality. We see in Table 5.6 that the model is indeed sensitive to the temperature, which is a common observation in other settings. The weight parameter and the batch size are relatively less important while also influencing the results to a certain extent. The latent space dimensionality observe The dimensionality of the latent space is set to $d = 256$ as in `TEMOS` (Chapter 4) but we can observe that $d = 128$ brings overall better performance.

5.3.4 Qualitative results

In Figure 5.3, we provide sample qualitative results for text-to-motion retrieval on the full test set of H3D. For each query text displayed on the left, top-5 retrieved motions are shown on the right along with their similarity scores. Note that the ground-truth text labels (at the bottom of each motion) for the retrieved motions are not used, and the gallery motions are unseen at training. For the first two examples with 'playing violin' and 'handstand', we retrieve

Temp.	Text-motion retrieval				Motion-text retrieval			
τ	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
0.001	9.52	21.81	27.23	12.00	7.47	9.76	16.51	15.50
0.01	21.45	29.04	38.80	6.00	**21.08**	**27.11**	**33.61**	**9.50**
0.1	**24.58**	**30.24**	**41.93**	**5.00**	19.64	23.73	32.53	**9.50**
1.0	1.08	1.93	3.61	306.5	1.81	1.93	2.41	372.0

(a)

Weight	Text-motion retrieval				Motion-text retrieval			
$\lambda_{\mathbf{NCE}}$	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
0.001	18.55	23.25	36.75	7.00	18.19	**24.34**	31.45	11.50
0.01	20.84	26.99	37.23	7.00	18.92	23.13	32.17	10.25
0.1	**24.58**	**30.24**	**41.93**	**5.00**	**19.64**	23.73	32.53	**9.50**
1.0	19.52	24.46	34.46	7.00	19.04	**24.34**	**35.06**	**9.50**

(b)

Batch	Text-motion retrieval				Motion-text retrieval			
size	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
16	**25.42**	**31.57**	40.12	6.00	**20.36**	24.10	**33.73**	**8.00**
32	24.58	30.24	**41.93**	**5.00**	19.64	23.73	32.53	9.50
64	20.24	26.51	38.19	6.00	19.52	**24.22**	32.05	9.50
128	18.55	28.80	36.75	7.00	14.94	18.43	26.14	11.50

(c)

Latent	Text-motion retrieval				Motion-text retrieval			
dim.	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓	R@1 ↑	R@2 ↑	R@3 ↑	MedR ↓
64	18.80	28.67	38.43	6.00	18.07	21.81	31.45	9.50
128	**25.90**	**31.20**	40.72	6.00	**23.73**	**27.35**	**36.39**	**9.25**
256	24.58	30.24	**41.93**	**5.00**	19.64	23.73	32.53	9.50
512	23.13	28.43	35.42	7.00	20.36	26.39	33.61	10.50

(d)

Table 5.6: **Hyperparameters of the contrastive training:** We measure the sensitivity to the parameters τ (temperature), λ_c the weight of the contrastive loss, the batch size and the latent dimensionality. Note that the learning rate is proportionally altered when changing the batch size. We display a wide range of values to show the full trends.

the ground-truth motion at rank 1. We observe that the next ranked motions depict visually similar motions as well (e.g., 'cartwheel' involves standing on the hands). For the free-from prompt example 'Someone is swimming' (i.e., the exact text does not appear in the gallery), the three first motions resemble or involve the swimming action, whereas motions at ranks 4 and 5 are incorrect. We notice that the incorrect motions have a low similarity (< 0.6), and the human bodies are rotated similarly as in swimming.

We show additional qualitative results on the challenging H3D dataset for text-to-motion retrieval on the 4 proposed protocols. Protocols (a)(b) are used in Figures 5.4 and 5.5; (c) in Figure 5.6; and (d) in Figure 5.7. All examples are randomly chosen, (i.e., not cherry picked); therefore, are representative of the corresponding protocols. Overall, we observe that our model is capable of retrieving motions that are semantically similar to the text descriptions. The performance naturally improves as we move from harder to easier protocols. Our detailed observations can be found in the respective figure captions.

Video. We provide a supplementary video on our project page to allow viewing motions dynamically. In the video, we demonstrate qualitative results for text-to-motion retrieval on the two datasets KIT [Plappert et al. 2016] and H3D [Guo et al. 2022a]. Moreover, we illustrate the use case of moment retrieval on BABEL [Punnakkal et al. 2021].

5.3.5 Motion synthesis

Since `TMR` has a generative branch, similarly to `TEMOS`, it can be used as a motion generator, even though this is not its primary purpose. In Table 5.7, we examine the capabilities of `TMR` as a motion generator, alongside `TEMOS` and the generative approach by [Guo et al. 2022a], across different datasets and evaluators. Notably, `TMR` demonstrates competitive or superior text-to-motion retrieval performance compared to [Guo et al. 2022a] synthesis model across all evaluation metrics. This indicates that `TMR`, while not primarily designed for motion generation, does not compromise on synthesis quality. Moreover, when evaluation models — specifically those capable of both retrieval and synthesis like `TEMOS` and `TMR` — are used, there is a discernible preference

Motions	Eval on KIT-ML			Eval on H3D		
	Guo Ret.	TEMOS	TMR	Guo Ret.	TEMOS	TMR
Real motions	42.25	44.88	49.25	52.41	42.33	67.16
Guo Syn	36.88	47.00	48.38	45.80	37.73	55.38
TEMOS	43.88	90.50	76.88	40.76	79.71	72.38
TMR	43.50	71.88	89.25	44.67	57.35	92.44

Table 5.7: **Motion synthesis results:** We report R@1 text-to-motion retrieval performance of *generated* motions by the synthesis method of [Guo et al. 2022a] (Guo Syn.), TEMOS (Chapter 4), and our TMR synthesis branch, as well as the 'Real motions', on both KIT-ML (left) and H3D (right) benchmarks. Rows are different motion generation methods, columns are different retrieval evaluation models: retrieval method of [Guo et al. 2022a] (Guo Ret.), TEMOS, and our TMR retrieval branch. We use the protocol (d), i.e., 32 gallery size protocol from [Guo et al. 2022a] . We make several observations: (i) TMR, when used for motion synthesis, performs better than or similar to Guo Syn. [Guo et al. 2022a] across all 3 retrieval evaluation models, showing we do not sacrifice synthesis performance. (ii) Evaluation with retrieval models that can also perform synthesis (TEMOS and TMR) favors motions generated by their own model. (iii) Certain numbers are better than Real motions, potentially because generations are sometimes more faithful to the input text, which may incompletely describe the real motion, or due to the bias mentioned in (ii).

for motions synthesized by the same model, highlighting an inherent model bias. Interestingly, in some instances, generated motions outperform real motions, suggesting that synthetic motions can be more aligned with the input text descriptions, possibly due to the text's inability to fully encapsulate the real motion or the aforementioned bias. This highlights the versatility and effectiveness of TMR in synthesizing motions that matched to textual descriptions.

5.3.6 Use case: moment retrieval

While our focus is retrieval, once our model is trained, it can be used for a different use case. Here, we test the limits of our approach, by qualitatively evaluating the capability of TMR on the task of temporally localizing a natural language query in a long 3D motion sequence. This is similar in spirit to moment retrieval in videos [Regneri et al. 2013; Gao et al. 2017; Hendricks et al. 2017; Escorcia et al. 2019; Lei et al. 2020]. It is also related to categorical action localization in 3D motions [Sun et al. 2022b]; however, our input is

free-form text instead of symbolic action classes.

In Figure 5.8, we show four examples, where we apply a model pre-trained on H3D on BABEL sequences. In each example, the queried text is displayed on the top. The x-axis denotes the frame number, the green rectangle represents the ground-truth location for the given action, and the dashed red line marks the localization with the maximum similarity. We simply compute the motion features in a sliding window manner. The similarity between the text label and a 20-frame window centered at each frame is shown in the y-axis as a 1D plot over time. Despite our model not being trained for temporal localization, we observe its grounding potential. Moreover, there exists a domain gap between BABEL label used at test time and H3D used for training. In Figure 5.9, we provide complementary qualitative results to Figure 5.8. At the right of Figure 5.9 (b), we also show the localization potential on four very long sequences. As the search space gets larger, the similarity plot gets noisier; however, the maximum similarity still occurs at the ground-truth location (marked in green).

For quantitative evaluation, we first obtain the predicted localization by selecting the window size and location that gives the best similarity with the text query. Then, we compute the temporal IoU (intersection over union) between the ground-truth segment and the predicted one. In Figure 5.10, we report the localization accuracy, where a segment is counted as positive when it has an IoU more than a given threshold. We see that this simple approach can achieve reasonable results (20% of accuracy, with a threshold of 0.4). With a fixed window size of [20, 40, 60] frames, we obtain [17%, 19%, 14%], respectively. A dedicated localization method may consider moment proposal generation as in prior video localization work [Soldan et al. 2021; Soldan et al. 2022], or a proposal-free approach that trains directly to regress temporal boundaries.

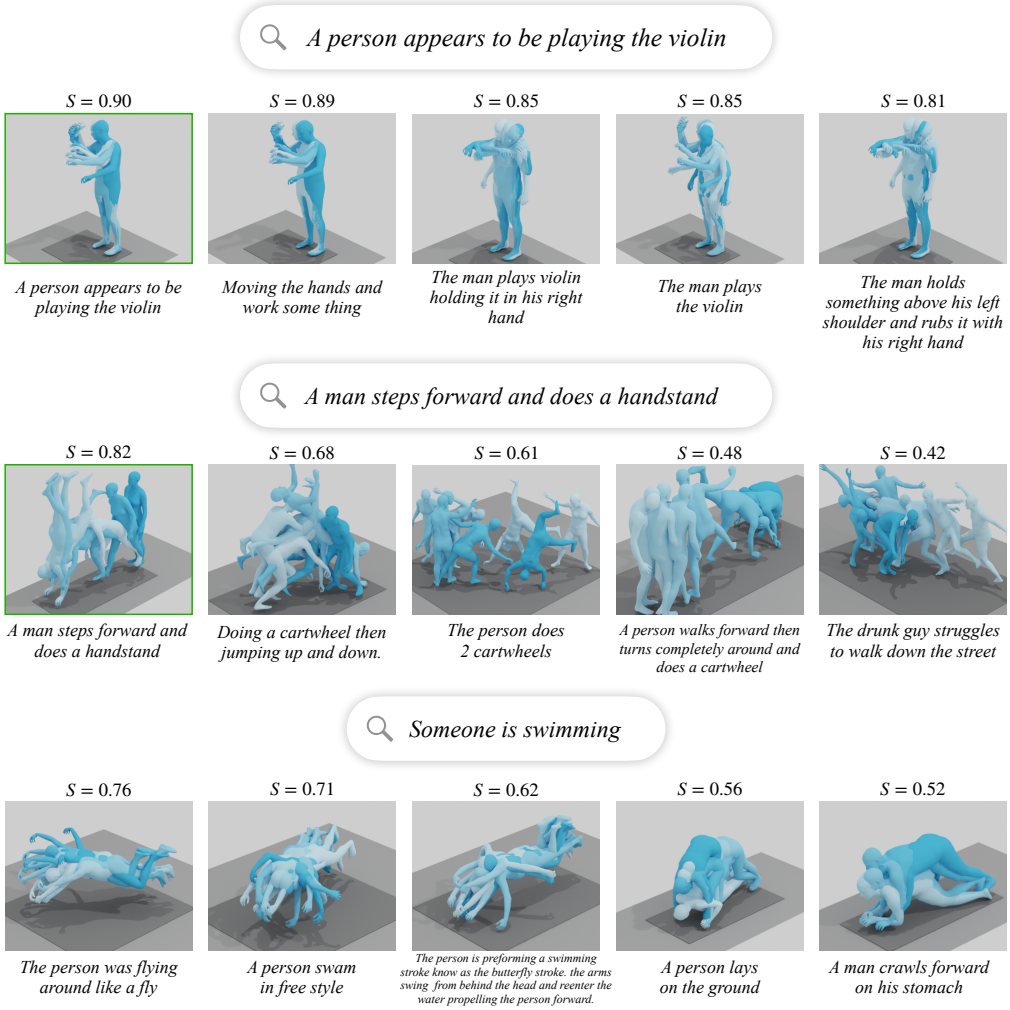


Figure 5.3: **Qualitative retrieval results:** We demonstrate example queries on the left, and corresponding retrieved motions on the right, ranked by text-motion similarity. The similarity values are displayed on the top. For each retrieved motion, we also show their accompanying ground-truth text label; note that we do not use these descriptions, but only provide them for analysis purposes. The motions from the gallery are all from the test set (unseen during training). In the first row, all top-5 retrieved motions correspond visually to 'playing violin' and the similarity scores are high > 0.80. In the second row, we correctly retrieve the 'handstand' motion at top-1, but the other motions mainly perform 'cartwheel' (which involves shortly standing on hands), but with a lower similarity score < 0.70. For the last example, we query a free-form text 'Someone is swimming', which does not exist in the gallery (but the word 'swim' does). The model successfully finds swimming motions among the top-3, and the other two motions involve the body parallel to the ground.

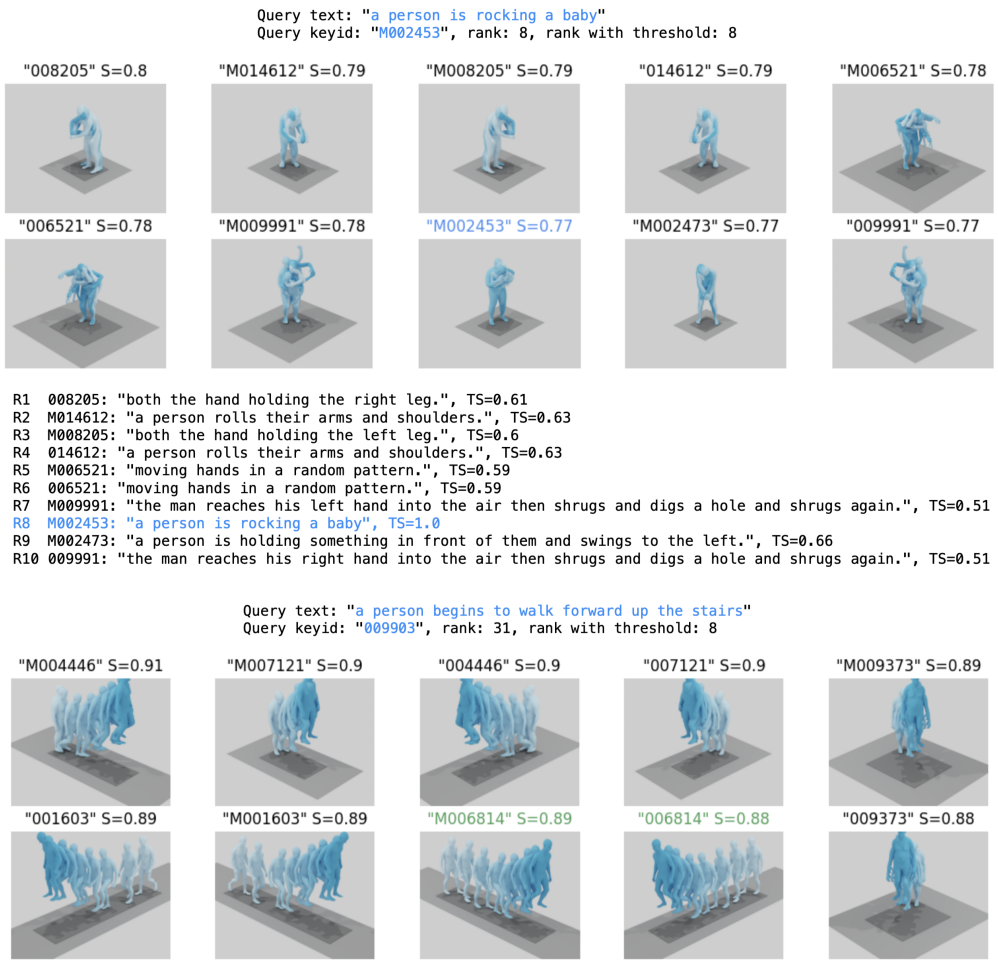


Figure 5.4: **Protocols (a) and (b) using all 4,380 motions in H3D:** For each text query, we show the top 10 ranks for the text-to-motion retrieval. Our model generalizes to the concept of "rocking a baby" in the first example, even though this exact same text was not seen in the training set. In the second example, our model retrieves motions that are all coherent with the input query. However, according to evaluation protocol (a), the correct motion is ranked at 31. With the permissive protocol (b), we mark the rank 8 as correct, because their text similarity (TS) is higher than the threshold 0.95.

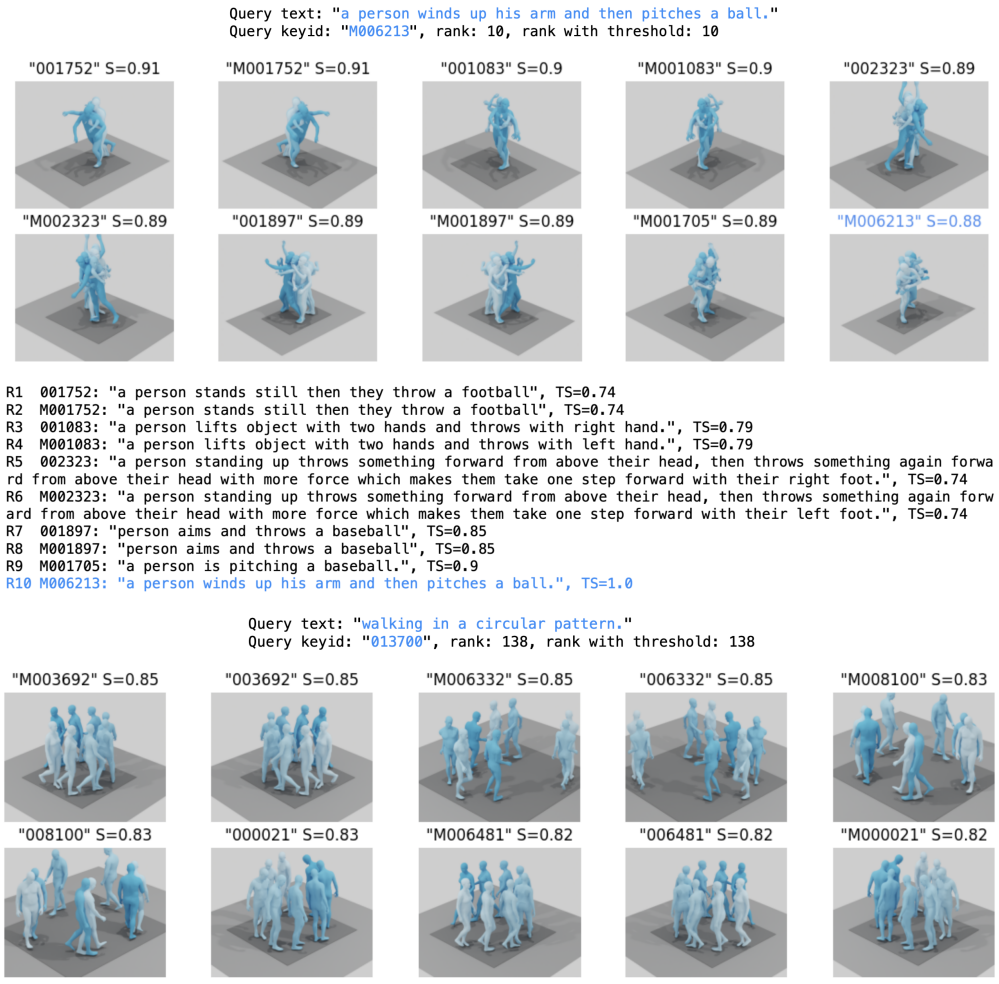


Figure 5.5: **Protocols (a) and (b) using all 4,380 motions in H3D (continued):** On both examples, we see that our model retrieves reasonable motions, although the correct motions are ranked at 10 and 138.

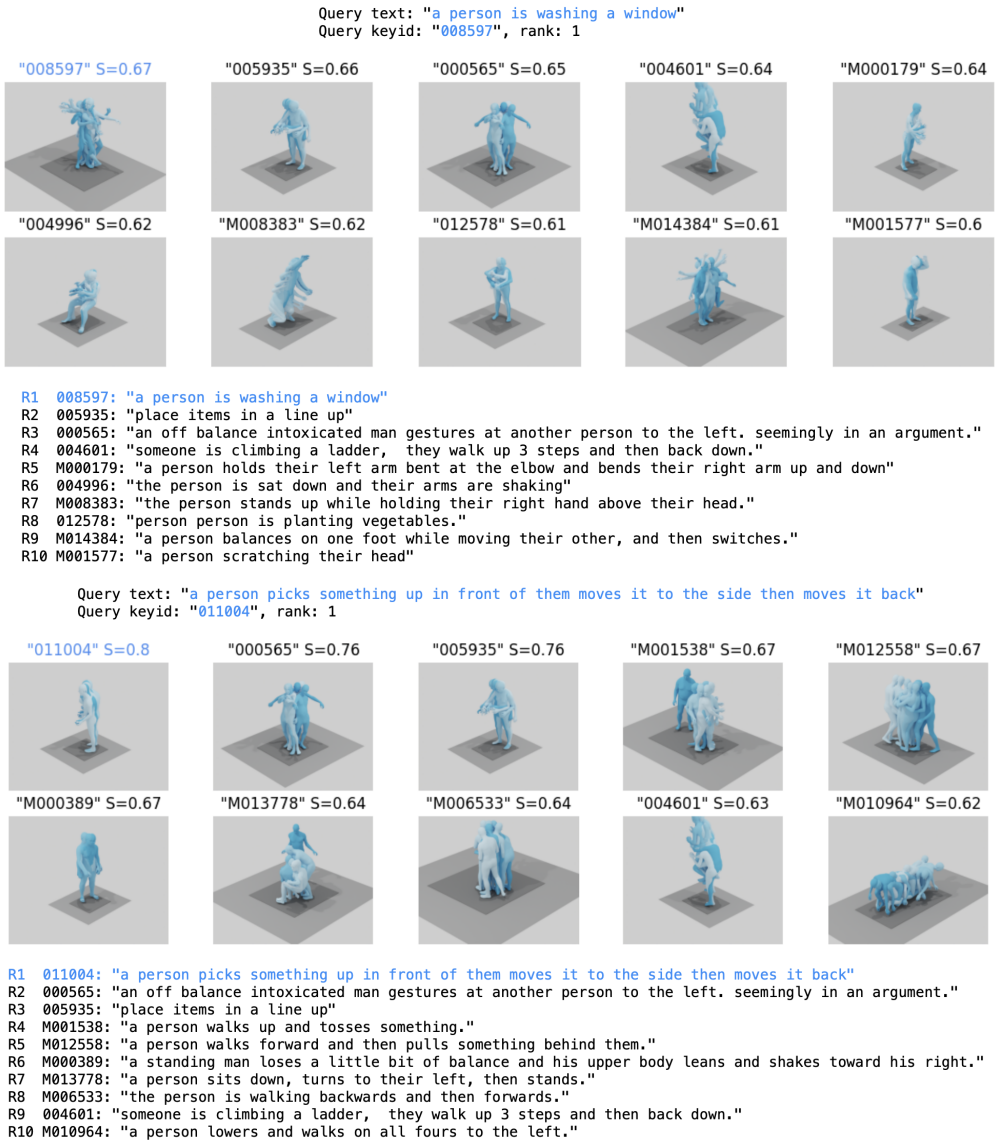


Figure 5.6: **Protocol (c) using the most dissimilar 100 texts on H3D:** As there are fewer motions than in protocols (a)(b), and they are more likely to be different, we naturally observe a better performance.

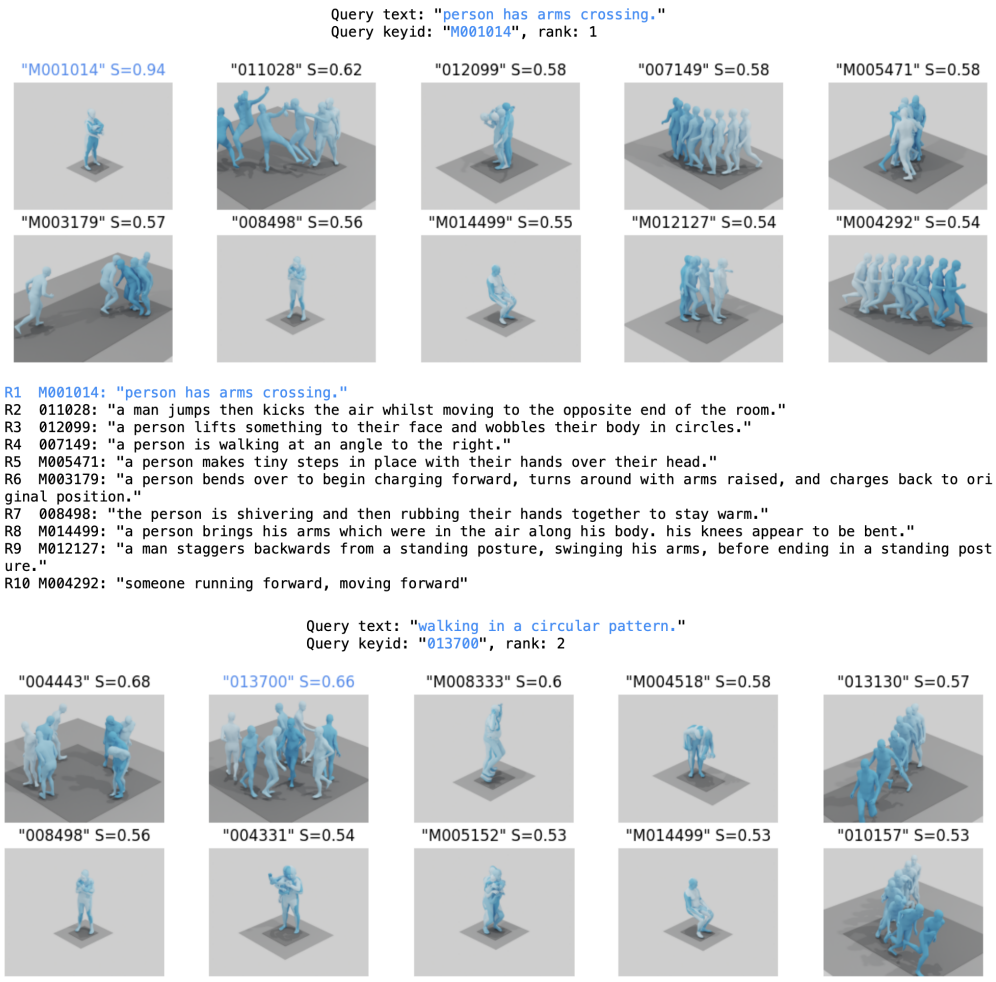


Figure 5.7: **Protocol (d) using random batches of size 32 on H3D:** As the gallery is very small, the correct motion tends to be at top ranks.

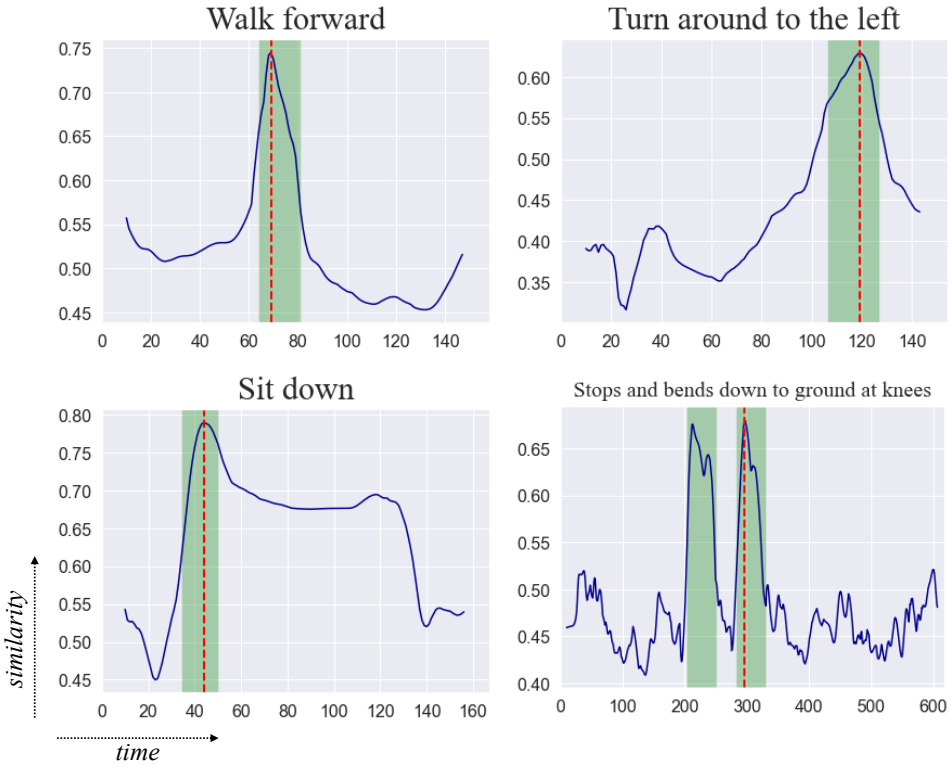


Figure 5.8: **Moment retrieval:** We plot the similarity between the temporally annotated BABEL text labels and the motions in a sliding window manner, and obtain a 1D signal over time (blue). We observe that a localization ability emerges from our model, even though it was not trained for temporal localization, and was not with the domain of BABEL labels. The ground-truth temporal span is denoted in green and the maximum similarity is marked with a dashed red line. More examples are provided in Figure 5.9.

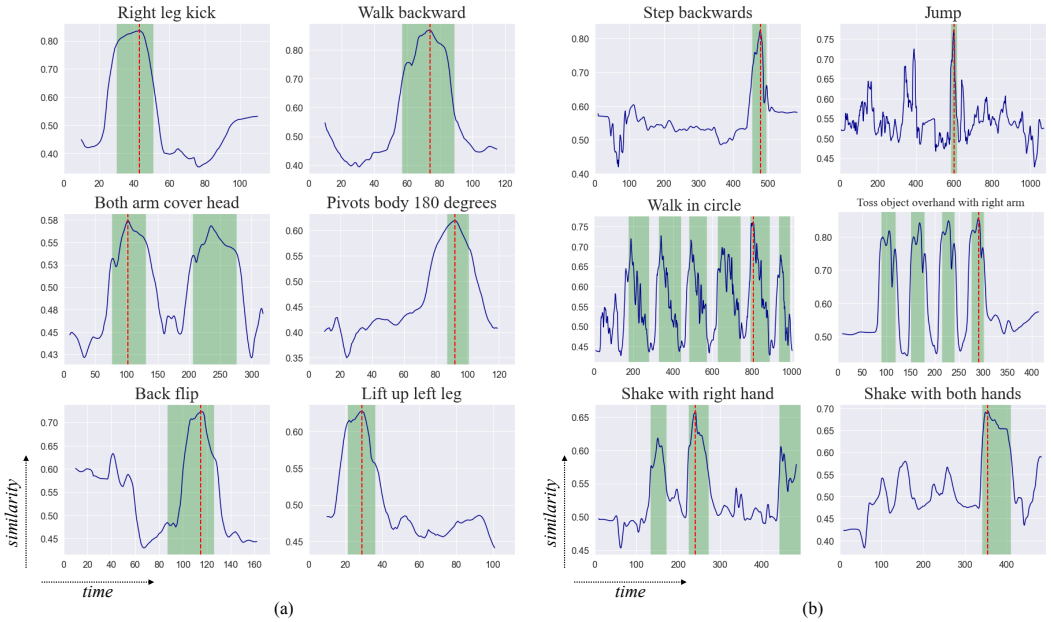


Figure 5.9: **Moment retrieval (qualitative):** To complement Figure 5.8 (a) we provide six additional temporal localization results for various text queries on the BABEL dataset. (b) We further visualize six challenging examples when querying on very long motion sequences, i.e., more than 500 frames (25 seconds).

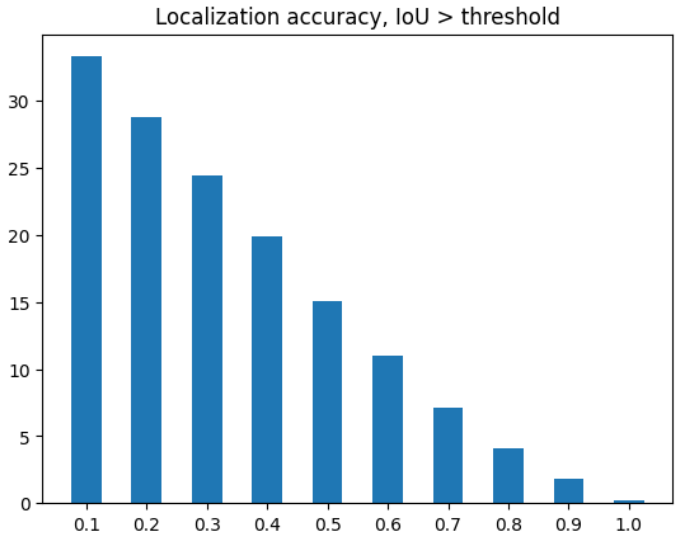


Figure 5.10: **Moment retrieval (quantitative):** We plot the localization accuracy (y-axis) with various IoU thresholds (x-axis).

5.3.7 Limitations

Our model comes with some limitations. Compared to the vast amount of data (e.g., 400M images [Schuhmann et al. 2021]) in image-text collections used to train competitive foundation models, our motion-text training data can be considered extremely small (e.g., 23K motions in H3D). The generalization performance of motion retrieval models to in-the-wild motions is therefore limited. Data augmentations such as altering text can potentially help to a certain extent (cf. [Athanasiou et al. 2023]); however, more motion capture is still needed. Another limitation concerns the case where one wishes to replace motion synthesis by retrieving a training motion. In this use case, the model requires the full encoded database (i.e., encoded training set) to be stored in memory, which can be memory-inefficient. Note that the encoded representation is considerably smaller than the original motion data. This could be dealt with using a hierarchical structuring of the motions and language descriptions (e.g. into upper-body motions, lower-body motions, sitting motions, standing motions, etc.).

5.4 Conclusion

In this chapter, we tackle the relatively little-studied problem of motion retrieval with natural language queries. We introduce `TMR`, a framework to jointly train text-to-motion retrieval and text-to-motion synthesis, with a special attention to the definition of negatives, taking into account the fine-grained nature of motion-language databases. We significantly improve over prior work, and provide a series of experiments highlighting the importance of each component. Future work may consider incorporating a language synthesis branch, along with the motion synthesis branch, to build a symmetrical framework, which could bring further benefits.

This text-motion retrieval model can be used for evaluation purposes, providing a valuable tool for assessing the quality of 3D human motion generation. In the next chapter, we will focus on the synthesis of 3D human motion from a multi-track timeline, and employ the `TMR` model to evaluate the generations.

Chapter 6

Multi-Track Timeline Control for Text-Driven 3D Human Motion Generation

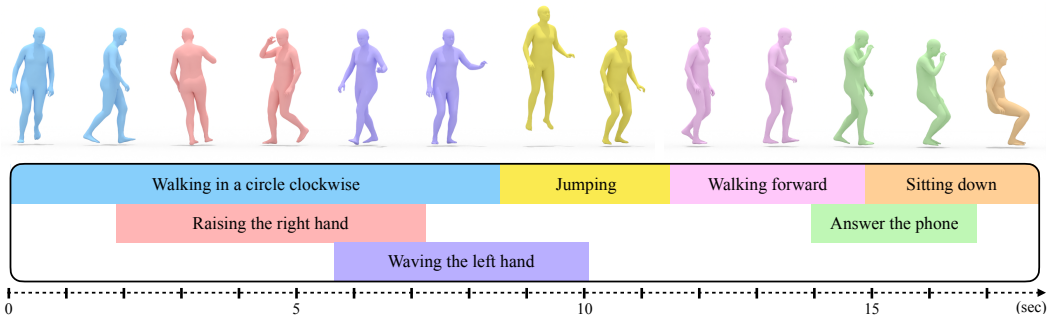


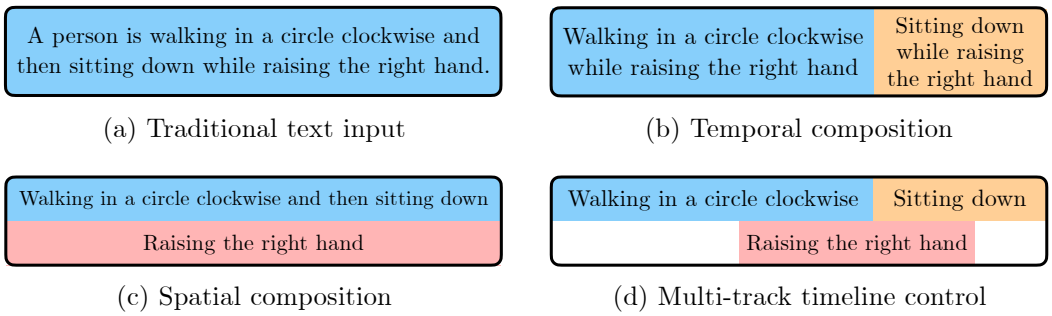
Figure 6.1: **Multi-track timeline control:** We introduce a new problem setting for text-driven motion synthesis, where the input consists of parallel tracks allowing simultaneous actions, as well as continuous temporal intervals enabling sequential actions. A long and complex motion can be generated (top) given the structured input of multiple simple textual descriptions, each corresponding to a temporal interval (bottom).

This chapter presents our final contribution to 3D human motion generation. Here, we create a new input interface to give users better control over the generation process.

Recent advances in generative modeling have led to promising progress on synthesizing 3D human motion from text, with methods that can generate character animations from short prompts and specified durations. However, using a single text prompt as input lacks the fine-grained control needed by animators, such as composing multiple actions and defining precise durations for parts of the motion. To address this, we introduce the new problem of timeline control for text-driven motion synthesis, which provides an intuitive, yet fine-grained, input interface for users. Instead of a single prompt, users can specify a multi-track timeline of multiple prompts organized in temporal intervals that may overlap. This enables specifying the exact timings of each action and composing multiple actions in sequence or at overlapping intervals. To generate composite animations from a multi-track timeline, we propose `STMC`, a new test-time denoising method. This method can be integrated with any pre-trained motion diffusion model to synthesize realistic motions that accurately reflect the timeline. At every step of denoising, our method processes each timeline interval (text prompt) individually, subsequently aggregating the predictions with consideration for the specific body parts engaged in each action. Experimental comparisons and ablations validate that our method produces realistic motions that respect the semantics and timing of given text prompts. Our code and models are publicly available at `https://mathis.petrovich.fr/stmc`.

6.1 Introduction

Motivated by applications in video games, entertainment, and virtual avatar creation, recent work has demonstrated substantial progress in learning to generate 3D human motion [Holden et al. 2016; Petrovich et al. 2021; Rempe et al. 2021; Xie et al. 2024]. Generating motions from text descriptions is of particular interest; it has the potential to democratize animation with a natural language interface that is intuitive for beginner and expert users alike. To this end, several methods have been proposed that synthesize reasonable character



(a) Traditional text input (b) Temporal composition

(c) Spatial composition (d) Multi-track timeline control

Figure 6.2: **Text-driven motion synthesis tasks:** Our framework generalizes (a) traditional *text-to-motion synthesis* given one text and one duration, (b) *temporal composition* given a sequence of texts for non-overlapping intervals, and (c) *spatial composition* given a set of texts for a single interval. (d) *Multi-track timeline control* uses a set of texts for arbitrary intervals, allowing fine-grained control over the timings of several complex actions.

animations given a single text prompt and fixed duration as input [Petrovich et al. 2022; Zhang et al. 2022a; Tevet et al. 2023].

While these methods are a promising first step towards faster and more accessible animation interfaces, they lack the precise control that is crucial for many animators. Consider the input prompt (see Figure 6.2d): "A human walks in a circle clockwise, then sits, simultaneously raising their right hand towards the end of the walk, the hand raising halts midway through the sitting action." Due to a lack of representative training data, prior work struggles with such complex text prompts [Petrovich et al. 2022; Tevet et al. 2023]. Namely, the prompt includes *temporal* composition [Athanasiou et al. 2022] where multiple actions are performed in sequence (e.g., walking *then* sitting), along with *spatial* composition [Athanasiou et al. 2023] where several actions are performed simultaneously with differing body parts (e.g., walking *while* raising hand). Furthermore, such lengthy prompts quickly become unwieldy for the user and, despite their detailed descriptions, are still ambiguous with respect to the timing and duration of the constituent actions.

To improve controllability, we propose the new problem of *multi-track timeline control for text-driven 3D human motion synthesis*. In this task, the user provides a structured and intuitive timeline as input (Figure 6.1), which contains several (potentially overlapping) temporal intervals. Each interval corresponds to a precise textual description of a motion. As shown

in Figure 6.2d, the complex example prompt discussed earlier becomes simple to specify within the timeline, and allows animators to control the timing of each action. Such a timeline interface is already common in animation and video editing software, and is analogous to control interfaces that have recently emerged from the text-to-image community [Zhang et al. 2023a], e.g., image generation from a segmentation mask.

Multi-track timeline control for text-driven motion synthesis is a generalization of several motion synthesis tasks, and therefore brings many challenges. In particular, the multi-track timeline input can achieve (see Figure 6.2):

- *Text-to-motion synthesis* [Guo et al. 2022a; Petrovich et al. 2022] – specifying a single interval (i.e., duration) with one textual description,
- *Temporal composition* [Athanasiou et al. 2022; Zhang et al. 2023b] – a sequence of textual descriptions corresponding to non-overlapping intervals,
- *Spatial (body-part) composition* [Athanasiou et al. 2023] – a set of text prompts performed simultaneously with differing body parts.

Solving this task is difficult due to the lack of training data containing complex compositions and long durations. For example, a timeline-controlled model must handle the multi-track input containing several prompts, rather than a single text description. Moreover, the model must account for both spatial and temporal compositions to ensure seamless transitions, unlike prior work that has addressed each of these individually. The timeline also relaxes the assumption of a limited duration (<10 sec) made by many recent text-to-motion approaches [Zhang et al. 2022a; Chen et al. 2023; Tevet et al. 2023].

To address these challenges, we introduce a method for **S**patio-**T**emporal **M**otion **C**ollage (`STMC`). Our method copes with the lack of appropriate training data by operating at test time, leveraging a pre-trained motion diffusion model such as off-the-shelf MDM [Tevet et al. 2023] or MotionDiffuse [Zhang et al. 2022a]. At each denoising step, `STMC` first applies the diffusion model on each text prompt in the timeline independently to predict a denoised motion for the corresponding intervals. Our key insight is to stitch together such independent generations in both space and time before continuing to denoise. For spatial

compositions, automatic body part associations [Athanasiou et al. 2023] allow coherently concatenating predictions together. Score arithmetic [Zhang et al. 2023b] is used to ensure smooth transitions for temporal compositions. To further improve the performance of `STMC`, we introduce MDM-SMPL, which makes several improvements to prior motion diffusion models [Tevet et al. 2023], including directly using the SMPL [Loper et al. 2015] body representation.

The performance of `STMC` on timeline control for text-driven motion synthesis is verified through comprehensive comparisons and a perceptual user study. In summary, the central contribution of this work consists of: (i) the new problem of *multi-track timeline control for text-driven 3D human motion synthesis*, and (ii) a novel test-time technique, `STMC`, that effectively structures the denoising process to ensure faithful execution of all prompts in a timeline. As a side contribution, (iii) we upgrade MDM to directly support the SMPL body representation instead of skeletons, and reduce runtime through fewer denoising steps. Our code and models are publicly available on our website.

6.2 Method

We first formulate the new problem setup of multi-track timeline control (Section 6.2.1), then propose a motion denoising strategy to handle timeline inputs (Section 6.2.2 and Section 6.2.3), and finally summarize our improved diffusion model (Section 6.2.4).

6.2.1 Timeline control problem formulation

Inputs. As illustrated in Figure 6.1, the multi-track timeline enables users to define multiple intervals, each linked to a natural language prompt describing the desired human motion. For the jth prompt in the timeline, we represent its temporal interval as $[a_j, b_j]$ and the corresponding prompt as C_j. The intervals are arranged in a multi-track layout on the timeline, allowing for overlaps. Both the duration of each interval and of the overall timeline are variable, and users can add an arbitrary number of tracks (rows) to the timeline

(although, in practice, a character can most often perform a handful of actions simultaneously).

Outputs. The goal is to generate a 3D human motion that follows all the text instructions at the specified intervals. A human motion $\boldsymbol{x}$ lasting N timesteps is represented as a sequence of pose vectors $\boldsymbol{x} = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N)$ with each pose $\boldsymbol{x}^i \in \mathbb{R}^d$. Several recent works [Zhang et al. 2022a; Tevet et al. 2023] use the pose representation from [Guo et al. 2022a] with d=263, which contains root velocities along with local joint positions, rotations, and velocities. Other pose representations like SMPL [Loper et al. 2015] can also be used (see Section 6.2.4).

6.2.2 Background: motion diffusion models

Our generation method (Section 6.2.3) leverages a pre-trained motion diffusion model such as MDM [Tevet et al. 2023] or MotionDiffuse [Zhang et al. 2022a] trained on single text prompts, which we briefly review here. These methods follow a denoising diffusion scheme and synthesize animations through iterative denoising of a noisy pose sequence. Given a clean motion $\boldsymbol{x}_0$, a Gaussian diffusion process is employed to corrupt the data to be approximately $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Each step of this process is given by:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\mathbf{I}) \tag{6.1}$$

with β_t defined by the noise schedule. Note the denoising step t is not to be confused with the temporal timestep i, which indexes the sequence of poses in the motion. In practice, one can make sampling $\boldsymbol{x}_t$ easier by using the reparameterization trick $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$.

Sampling from a diffusion model requires reversing this process to recover a clean motion from random noise. While $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is hard to compute, the probability conditioned on $\boldsymbol{x}_0$ is tractable [Ho et al. 2020]:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \boldsymbol{\Sigma}_t) \ , \tag{6.2}$$

where

$$\mu_t(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\boldsymbol{x}_0 \tag{6.3}$$

$$\boldsymbol{\Sigma}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\mathbf{I}\ . \tag{6.4}$$

Since $\boldsymbol{x}_t$ is known at sampling time, we approximate the reverse distribution by training a denoising model $\hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t, C)$ to estimate $\boldsymbol{x}_0$, where C is the text conditioning. This model is trained with the simplified loss function as in [Ho et al. 2020] (i.e., without the t-dependent factor):

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\epsilon},t,\boldsymbol{x}_0,C}\left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, C) - \boldsymbol{x}_0\right\|_2^2 \tag{6.5}$$

with $\boldsymbol{x}_0$ and C sampled from a dataset of motion-text pairs, step t sampled uniformly, and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ used to corrupt the ground truth motion. To enable classifier-free guidance [Ho et al. 2021] at sampling time, the text conditioning C is dropped with some probability at each training iteration.

At test time, the sampling (reverse) process starts from random noise and denoises iteratively for T steps to obtain a clean 3D human motion. At each denoising step, the model is conditioned on the single input text prompt (e.g., Figure 6.2a).

For more details on diffusion models, we recommend that the reader look at the unified perspective of [Luo 2022].

6.2.3 `STMC`: Spatio-Temporal Motion Collage

`STMC` operates only at test time, enabling an off-the-shelf, pre-trained denoising model to generate motion conditioned on a multi-track timeline. At *every* denoising step, our method takes as input the current noisy motion $\boldsymbol{x}_t$ encapsulating the entire timeline and outputs a corresponding clean motion $\hat{\boldsymbol{x}}_0$. As shown in Figure 6.3c, `STMC` uses the denoising model to independently predict a clean motion crop corresponding to each of the input text prompts. These predictions are stitched together spatially using body part annotations for each text prompt (Figure 6.3a), and stitched in time to ensure the clean motion

smoothly spans the entire timeline (Figure 6.3b). This final composite motion becomes the output of the current step $\hat{\boldsymbol{x}}_0$, which is used to sample $\boldsymbol{x}_{t-1}$ with eq. (6.2) and continue the denoising process. To enable body part stitching, `STMC` assumes the denoiser operates on explicit poses [Zhang et al. 2022a; Tevet et al. 2023], rather than in a latent space [Chen et al. 2023].

Motion cropping and denoising. The input $\boldsymbol{x}_t$ at denoising step t extends over the duration of the entire timeline. As shown in Figure 6.3c, we first temporally split the input into motion "crops" to separately denoise each text prompt. For each interval $[a_j, b_j]$, the motion is cropped in time to $\boldsymbol{x}_t^{a_j:b_j} = \boldsymbol{x}_t[a_j : b_j]$. The crop, along with the text prompt C_j, is given to the denoising model to predict a corresponding clean motion crop $\hat{\boldsymbol{x}}_0^{a_j:b_j}$. Denoising each text prompt independently gives high-quality motion from pre-trained models since each prompt typically contains a single action and the interval duration is reasonably short (<10 sec).

Two or more text prompts in the timeline may overlap in time, meaning the predicted clean crops will also overlap. As a concrete example, suppose the crops for "walking in a circle" and "raising right hand" are overlapping, as in Figure 6.3. In this case, it is not clear which of the two generated motions should be assigned to the overlapping region. To construct a motion that matches both prompts, we need the leg motion from "walking in a circle" and the right arm motion from "raising right hand". We therefore stitch together outputs from overlapping prompts based on automatically labeled body parts, as detailed next.

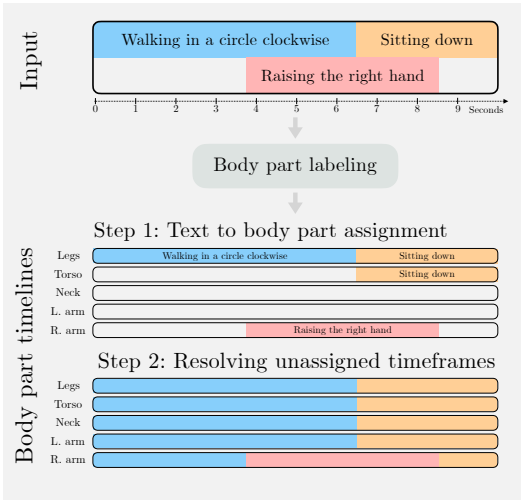
Spatial (body-part) stitching. Spatial stitching follows `SINC` [Athanasiou et al. 2023] (Annex B), which proposed to combine compatible body-part motions from mocap sequences through simple concatenation. While `SINC` applies stitching only once, `STMC` does so at *every* step of denoising, encouraging a more coherent composition of movements by allowing the denoiser to correct any artifacts. This is possible because the denoiser outputs explicit human poses (i.e., we know which indices correspond to arms, legs, etc. within the pose vector), so we can extract body-part motions from separate crops and spatially combine them to obtain a composite motion. To achieve this, we first pre-process the input timeline to assign a text prompt to each body part at

every timestep, thereby creating a separate motion timeline for every body part (see Figure 6.3a): *left arm*, *right arm*, *torso*, *legs* and *head*.

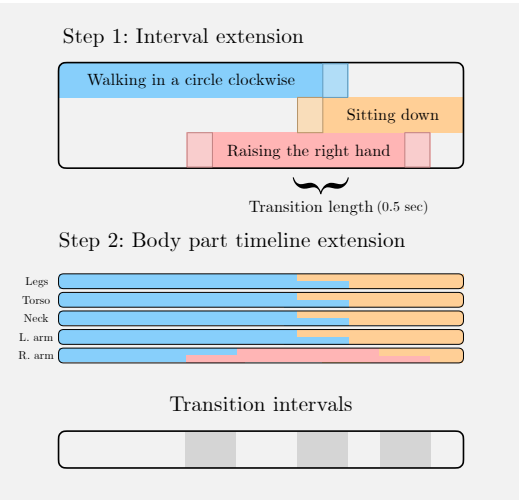
As shown in Figure 6.3a, each text prompt in the multi-track timeline is first annotated with a set of body parts involved in the motion. This can be done automatically by querying GPT-3 [Brown et al. 2020] as in `SINC`, or directly given by the user for additional creative control. Then, each text prompt is assigned to its annotated body parts within the corresponding time interval, which assumes that body parts at overlapping intervals are compatible (e.g., if a prompt is annotated with "legs", then no other prompt should involve legs throughout its entire interval).

Resolving unassigned timeframes. To fill in the remainder of the body-part timelines where body parts have not been annotated to a text prompt, heuristics similar to `SINC` are used. As shown in Figure 6.4, we first cut the body part timelines so that there are no new texts appearing or disappearing within each cut (left). Then, we apply the `SINC` [Athanasiou et al. 2023] heuristic (right) for each cut. The heuristic consists of (1) choosing a "base" text prompt, (2) assigning all the body parts to the base text, and (3) assigning (overriding) the body parts corresponding to the other texts. Note that the other texts are sorted based on the number of body parts involved in decreasing order.

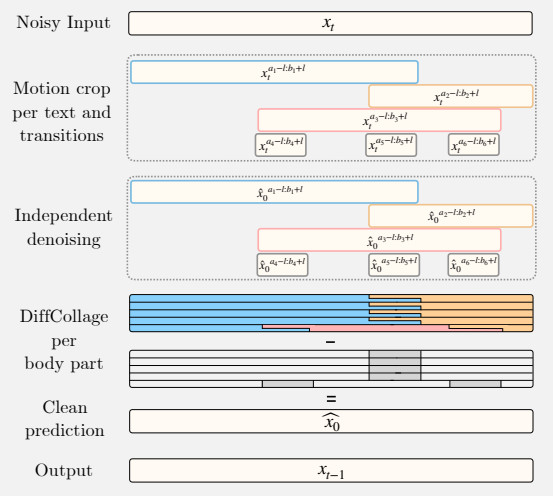
Finally, during the denoising step (Figure 6.3c), each crop $\boldsymbol{x}_t^{a_j:b_j}$ is split into separated body-part motions and concatenated together as specified by the body-part timelines to obtain the output $\hat{\boldsymbol{x}}_0$.



(a) Body part timeline partitioning

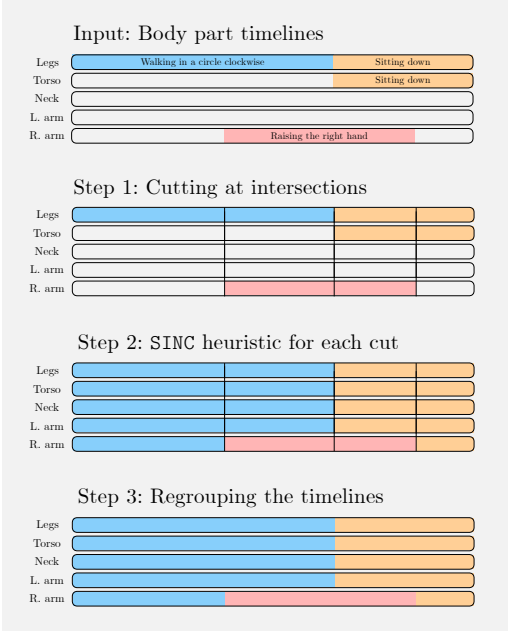


(b) Temporal stitching

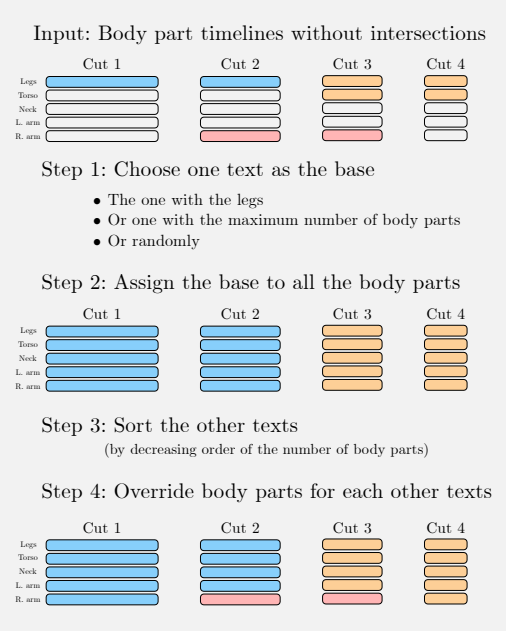


(c) STMC denoising step t

Figure 6.3: **Overview of STMC:** Before denoising, the multi-track timeline is first **(a)** partitioned into relevant body parts per text (using LLM-based labeling [Athanasiou et al. 2023]) to create body part timelines, which are then **(b)** extended to overlap, leading to the transition intervals used for temporal stitching *per body part* with DiffCollage [Zhang et al. 2023b]. **(c)** At each denoising step, motions for each prompt are denoised independently before being combined based on the body-part timelines. The composite motion is re-noised by sampling $\boldsymbol{x}_{t-1}$ from $\mathcal{N}(\mu_t(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_{\mathbf{0}}), \boldsymbol{\Sigma}_t)$ (as in eq. (6.2)) before being passed to the next step. A more detailed explanation of Step 2 of (a) can be found in Figure 6.4.



(a) Resolving unassigned timeframes



(b) SINC heuristic

Figure 6.4: **Additional details of STMC:** To create the final body parts timeline (step 2 of Figure 6.3) , we need to "fill the holes" by assigning a text to all locations of the body parts timeline (left). This is done by first splitting the timelines such that there is no intersection with other intervals, and then applying the SINC heuristic for each cut (right). Finally, we regroup the intervals by removing the cuts to obtain full body part timelines.

Temporal stitching. Because the motion crops are denoised independently, simple temporal concatenation of body-part motions from different text prompts will cause abrupt transitions. To mitigate these potential artifacts, we apply DiffCollage [Zhang et al. 2023b] to *each body-part* motion. As shown in Figure 6.3b, instead of directly denoising $\boldsymbol{x}_t^{a_j:b_j}$ for each text prompt, we denoise an expanded time interval $[a_j - l, b_j + l]$, where l is the desired overlap length between adjacent motion crops (e.g., fixed to 0.25 sec). Concretely, for the temporal transition between prompts j and k, we have $\hat{\boldsymbol{x}}_0^{a_j-l:b_j+l}$ and $\hat{\boldsymbol{x}}_0^{a_k-l:b_k+l}$ after denoising. We then *unconditionally* denoise a small (0.5 sec) crop of motion centered on the overlap between j and k to obtain $\hat{\boldsymbol{x}}_0^{\text{uncond}}$. The final predicted motion spanning intervals j and k is computed as $\hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0^{a_j-l:b_j+l} + \hat{\boldsymbol{x}}_0^{a_k-l:b_k+l} - \hat{\boldsymbol{x}}_0^{\text{uncond}}$, as depicted in Figure 6.3c. This equation derives from a factor graph representation of the problem, as detailed in DiffCollage [Zhang et al. 2023b].

6.2.4 SMPL support for motion diffusion model

While `STMC` works well with off-the-shelf models [Zhang et al. 2022a; Tevet et al. 2023] (see Section 6.3), we propose several practical improvements to MDM [Tevet et al. 2023] to further enhance results.

Pose representation. We propose a motion representation for diffusion that includes SMPL pose parameters. We represent a pose $\boldsymbol{x} \in \mathbb{R}^d$ by $\boldsymbol{x} = [r_z,\ \dot{r_x},\ \dot{r_y},\ \dot{\alpha},\ \boldsymbol{\theta},\ \mathbf{j}]$ where r_z is the Z (up) coordinate of the pelvis, $\dot{r_x}$ and $\dot{r_y}$ are the linear velocities of the pelvis, $\dot{\alpha}$ is the angular velocity of the Z angle of the body, $\boldsymbol{\theta}$ are the SMPL [Loper et al. 2015] pose parameters (encoded with the $6D$ representation [Zhou et al. 2019]), and $\mathbf{j}$ are the joints positions (computed with the SMPL layer). Inspired by Holden et al. [Holden et al. 2016] and Guo et al. [Guo et al. 2022a], which use a rotation invariant representation, we represent the joints $\mathbf{j}$ in a coordinate system local to the body. To make $\boldsymbol{\theta}$ local to the body, we remove the Z rotation from the SMPL global orientation. This representation enables us to directly extract the SMPL pose parameters, eliminating the need for expensive test-time optimization-based methods to fit the generated motion on a SMPL body, as in previous work [Bogo et al. 2016; Zuo et al. 2021]. Moreover, the local joint rotations in SMPL, which are relative to parents in the kinematic tree, are more amenable to body-part stitching than root-relative joint positions. This is because any change to a joint rotation is propagated to all children in the kinematic tree, unlike root-relative joint positions which may not be coherent when simply concatenated together.

Architecture and training. We use a similar architecture as MDM [Tevet et al. 2023], but make the following changes (in addition to using the SMPL body):

- We use a cosine schedule as introduced by [Chen 2023] with 100 steps instead of a linear schedule with 1000 steps.

- After padding to the maximum duration in a batch, we mask the padded area in the Transformer encoder so that the padded area is not used for the computation.

Other minor changes include using two separate tokens for the diffusion step t and the text embedding (instead of one), using two register tokens (introduced in [Darcet et al. 2024]), and pre-computing CLIP embeddings for faster training. We train the model for 10000 epochs with a batch size of 128.

6.3 Experiments

We first present the data (Section 6.3.1) and the evaluation protocols (Section 6.3.2) used in the experiments. We then show comparisons with baselines quantitatively (Section 6.3.3) and with a perceptual study (Section 6.3.4), followed by qualitative results (Section 6.3.5). We conclude with a discussion of the limitations (Section 6.3.6).

6.3.1 Datasets

HumanML3D [**Guo et al. 2022a**] is a text-motion dataset that provides textual descriptions for a subset of the AMASS [Mahmood et al. 2019] and HumanAct12 [Guo et al. 2020] motion capture datasets. It consists of 44970 text annotations for 14616 motions. This dataset is used to train all diffusion models used in our experiments. For MDM [Tevet et al. 2023] and MotionDiffuse [Zhang et al. 2022a], we use publicly available models pre-trained on the released version of HumanML3D with the original motion representation from [Guo et al. 2022a]. Consequently, these methods require test-time optimization to obtain SMPL pose parameter outputs. For training our MDM-SMPL diffusion model, which is designed to directly generate SMPL pose parameters, we re-process the dataset and exclude the HumanAct12 subset as SMPL poses are not available for this dataset.

Multi-track timeline (MTT) dataset. To properly evaluate our new task, we introduce a new challenging dataset of 500 multi-track timelines. Each timeline in the dataset is automatically constructed and contains three prompts on a two-track timeline (e.g., Figure 6.2d). To construct these timelines, we first manually collect a set of 60 texts covering a diverse set of "atomic" actions

(e.g., "punch with the right hand", "jump forward", "run backwards", see the appendix of [Petrovich et al. 2024] for the full list), and annotate the involved body parts for each text. To serve as ground truth for computing evaluation metrics (Section 6.3.2), we also select motion samples from AMASS that correspond to each text. Based on the atomic texts, we automatically generate timelines containing three prompts and two tracks (rows). For each timeline, the first track is filled with two consecutive prompts sampled from the set of texts and given randomized durations. We randomly sample durations with a mean of 6.0 seconds and a standard deviation of 1.0 seconds. A third random text with complementary body-part annotations is then placed in the second track at a random location in time.

The main reasons for restricting the evaluation to three prompts are (i) to keep the cognitive load for users low in the perceptual study, subsequently increasing the reliability of the results, and (ii) to construct a minimal setup where we can fairly compare against baselines in a controlled setting, eliminating confounding factors such as the number of prompts. Though these timelines contain only three prompts, they already pose a significant challenge (see Section 6.3.3). Examples of timelines in the dataset are provided in 6.5 and qualitative results beyond three prompts can be found in the project page.

6.3.2 Evaluation metrics

Given the novelty of the task, identifying relevant metrics to evaluate different methods is crucial. Instead of relying on a single metric, we disentangle the evaluation of semantic correctness (how faithful individual motion crops are to the textual descriptions) from that of realism (e.g., temporal smoothness).

Semantic metrics. Firstly, we evaluate the alignment between the generated motion and the text description within the specified intervals on the timeline, which we term "per-crop semantic correctness". To assess this, we utilize our text-to-motion retrieval model `TMR` [Petrovich et al. 2023] (Chapter 5). Similar to how CLIP [Radford et al. 2021] functions for images and texts, `TMR` provides a joint embedding space that can be used to determine the similarity between a text and motion. Using `TMR`, we encode each atomic text

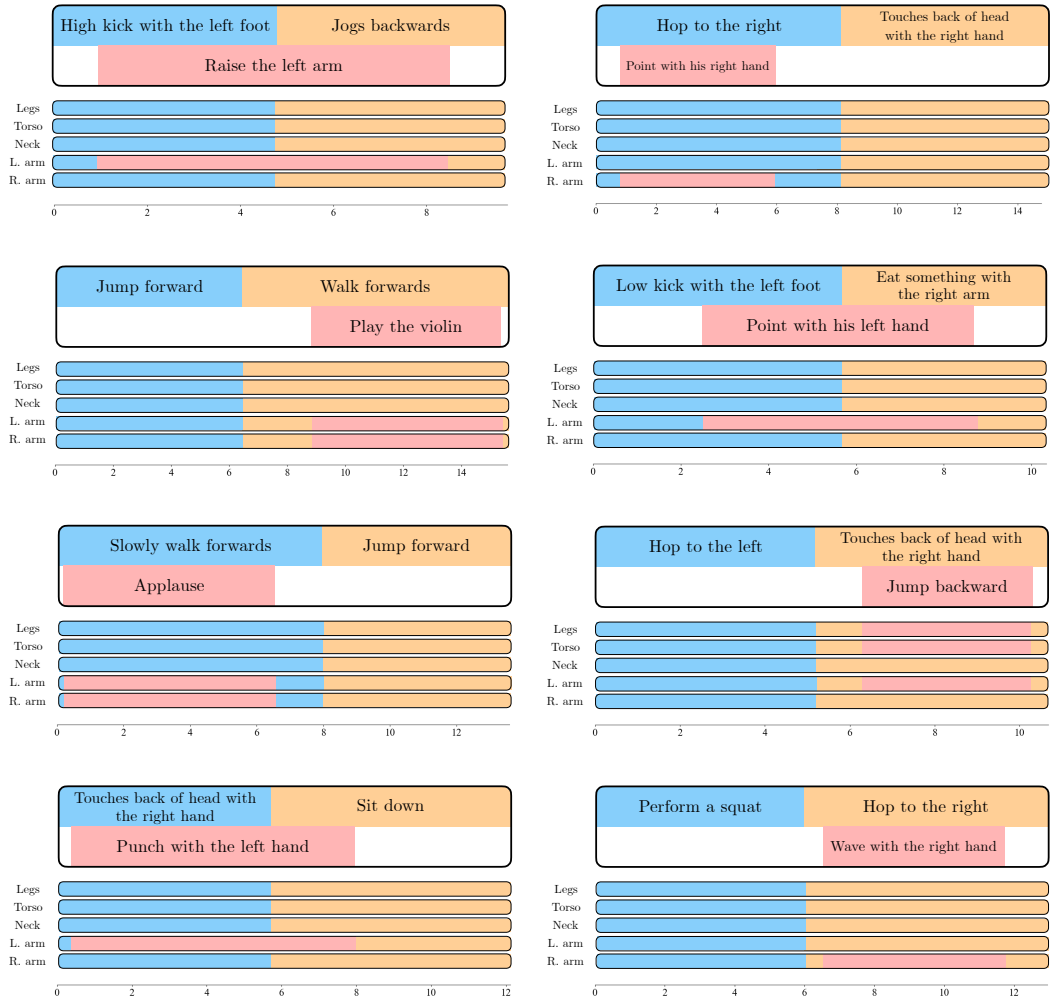


Figure 6.5: **Example timelines from MTT dataset:** We display several generated timelines, along with the automatically generated body part timelines. Although each timeline contains only three prompts, the generated timelines are diverse and specify complicated motions.

prompt and corresponding motion from our MTT dataset to obtain ground truth text and motion embeddings, respectively. Each generated motion crop is also embedded and the *TMR-Score*, a measure of cosine similarity ranging from 0 to 1, is calculated between the generated motion embedding and the ground truth. We report both motion-to-text similarity by comparing against the ground truth text embedding (*TMR-Score M2T*) and motion-to-motion similarity against the ground truth motion embedding (*TMR-Score M2M*). Such embedding similarity measures are akin to BERT-Score [Zhang et al. 2020a] for text-text, CLIP-Score [Hessel et al. 2021] for image-text, and more recently TEMOS-Score [Athanasiou et al. 2022] for motion-motion similarity. Since TMR is trained contrastively, its retrieval performance is better than TEMOS [Petrovich

et al. 2022] which only trains with positive pairs, leading to our decision to instead use TMR-Score. Moreover, its embedding space is optimized with cosine similarity, making the values potentially more calibrated across samples.

Ideally, the *TMR-Score M2T* between a generated motion crop and the corresponding input text prompt should surpass those of other texts. Hence, we also measure motion-to-text retrieval metrics (as in [Guo et al. 2022a]) including the frequency of the correct text prompt being in the top-1 (*R@1*) and top-3 (*R@3*) retrieved texts from the entire set of atomic texts.

Realism metrics. Secondly, we evaluate the realism of the generated motions, which includes transitioning smoothly between actions. While the Frechet Inception Distance (*FID*) between generated and ground truth motion in a learned feature space (e.g., TMR) is a common metric for quality, the embedding space of TMR is not trained on motions that are longer than 10 sec, and may therefore be unreliable for longer motions. Hence, we follow DiffCollage [Zhang et al. 2023b] and compute the *FID+* to evaluate transitions. The *FID+* metric measures FID based on 5 random 5-second motion crops from each timeline-conditioned motion generation. Following TEACH [Athanasiou et al. 2022] (Chapter A), we also measure the *transition distance* as the Euclidean distance (in cm) between the poses in two consecutive frames around the transition time. We choose to compute this distance in the local coordinate system of the body to more effectively capture transitions for individual body parts, rather than being dominated by global motion. This metric is sensitive to abrupt pose changes, and a motion should not have high transition distance to remain realistic.

Perceptual study. Since no quantitative metric can fully capture the subtleties of human motion, we also conduct perceptual studies, where human raters on Amazon Mechanical Turk judge the quality of the generated motions [Turk 2023]. To compare two generation methods, raters are presented with two videos of generated motions side-by-side rendered on a skeleton. The multi-track timeline is also visible with an animated bar that progresses along the timeline as the videos play. Users are asked which motion is *more realistic* and which one is *better at following the text in the timeline*; they may choose one of the two motions or mark "no preference". The studies presented in

Section 6.3.3 are performed on a set of 100 motions with multiple raters judging each pair. The preference for each video is determined by a majority vote from all raters. Responses are filtered for quality by using three "warmup" questions at the start of each 15-question survey along with two "honeypot" examples with objectively correct answers. The honeypot examples test a rater's understanding of the task: one example shows a motion with obviously severe limb stretching (realism understanding test) and the other displays a motion generated from a different timeline than the one displayed (timeline understanding test). If a rater fails to answer either of these questions correctly, all of their responses are discarded.

Method	Input type		Per-crop semantic correctness				Realism	
	#tracks	#crops	R@1 ↑	R@3 ↑	TMR-Score ↑ M2T	TMR-Score ↑ M2M	FID ↓	Transition distance ↓
Ground truth	-	-	55.0	73.3	0.748	1.000	0.000	1.5
MotionDiffuse [Zhang et al. 2022a]	Single	Single	10.9	21.3	0.558	0.546	0.621	1.9
DiffCollage	Single	Multi	22.6	43.3	0.633	0.612	0.532	4.6
SINC w/o Lerp	Multi	Multi	23.8	45.9	0.656	0.630	0.554	3.8
SINC w/ Lerp	〃	〃	24.9	46.7	0.663	0.632	0.552	1.0
STMC (ours)	〃	〃	24.8	46.7	0.660	0.632	0.531	1.5
MDM [Tevet et al. 2023]	Single	Single	9.5	19.7	0.556	0.549	0.666	2.5
DiffCollage	Single	Multi	24.9	42.3	0.636	0.623	0.600	2.2
SINC w/o Lerp	Multi	Multi	21.5	41.8	0.629	0.626	0.638	10.2
SINC w/ Lerp	〃	〃	23.3	43.1	0.634	0.628	0.630	2.8
STMC (ours)	〃	〃	25.1	46.0	0.641	0.633	0.606	2.4
MDM-SMPL	Single	Single	12.1	23.5	0.573	0.578	0.484	1.8
DiffCollage	Single	Multi	29.1	49.7	0.675	0.656	0.446	1.2
SINC w/o Lerp	Multi	Multi	32.3	50.5	0.676	0.667	0.463	4.2
SINC w/ Lerp	〃	〃	31.8	51.0	0.679	0.668	0.457	1.2
STMC (ours)	〃	〃	30.5	50.9	0.675	0.665	0.459	0.9

Table 6.1: **Quantitative baseline comparison**: Our method STMC is compared to several strong baselines when using three different denoising models. The single-text and DiffCollage baselines struggle to handle complex compositional prompts that results from collapsing the timeline down to a single track. The SINC baselines produce reasonable semantic accuracy by denoising prompts independently as in STMC, but cause abrupt or unnatural transitions with higher transition distance (underlined) or FID.

6.3.3 Quantitative comparison with baselines

We apply our STMC test-time approach on the pretrained diffusion models of MotionDiffuse [Zhang et al. 2022a], MDM [Tevet et al. 2023], and MDM-SMPL

(ours). For each denoiser, we establish several strong baselines by repurposing existing methods to the timeline-conditioned generation task for comparison. Results are shown in Table 6.1. Next to each method, the table indicates how many tracks the input timelines have (*#tracks*) and how many text prompts can be contained in a track (*#crops*). Next, we introduce each baseline and analyze results.

Single-text input [Zhang et al. 2022a; Tevet et al. 2023] baseline. The simplest approach to condition motion diffusion on a timeline is to convert the timeline into a single text description, which aligns with the model's training input format (e.g., Figure 6.2a). Given that our timeline dataset is consistently comprised of three motions (`A`, `B`, and `C`), we formulate single-text prompts as follows: "`A` and then `B` while `C`". While timing information can be included in the prompt, e.g., "`A` for 4 seconds", this is out-of-distribution for models trained on HumanML3D, leading to worse results. This method parallels the baseline strategies of `SINC` [Athanasiou et al. 2023] for spatial composition and `TEACH` [Athanasiou et al. 2022] for temporal composition.

As shown for each denoiser in Table 6.1, this approach is ineffective for both semantic correctness metrics and realism. Since these models cannot generate motions longer than 10 sec and there is no timing information in the prompt, for this experiment, outputs are limited to a maximum duration of 10 sec and semantic correctness metrics are reported over the entire duration of the motion rather than per-crop. The poor performance is a result of the models not being trained on the types of complex compositional prompts that result from collapsing the timeline to a single text description.

DiffCollage baseline [Zhang et al. 2023b]. Instead of converting the multi-track timeline into a single prompt, one can collapse it into a single track timeline containing a series of consecutive text prompts, i.e., transform the problem to be one of temporal composition. DiffCollage can then be used to temporally compose the sequence of actions. For example, the timeline in Figure 6.2d would be split into ["walking in a circle," "walking in a circle while raising the right hand," "sitting down while raising the right hand," "sitting down"]. Note that, unlike the single-text baseline, this splitting preserves the timings (*#crops*) in the timeline.

While the DiffCollage baseline generally produces smooth transitions and reasonable FID scores, the semantic accuracy is consistently worse than STMC. This is due to the complex spatial compositions within the prompts after collapsing the timeline into a single track, which models trained on HumanML3D struggle with. In contrast, STMC uses body-part stitching throughout denoising to compose actions from simpler prompts.

SINC [Athanasiou et al. 2023] **baseline**. Rather than performing body-part stitching iteratively at every denoising step, an alternative approach is to stitch body motions together only once after all crops have finished the entire denoising process. This is most similar to SINC and forms the basis for two baselines that accept the full multi-track timeline as input, similar to STMC.

SINC w/o Lerp concatenates body part motions at the end of denoising without considering temporal transitions. As a result, transitions tend to be abrupt as evidenced by high transition distances in Table 6.1 and occasional "teleporting" limbs in qualitative results. To mitigate this, *SINC w/ Lerp* employs linear interpolation (lerp) at transitions for smoother results, similar to the approach in TEACH [Athanasiou et al. 2022]. Though this leads to smoothness at transitions, FID scores tend to be slightly higher than STMC. The cause is obvious qualitatively, where the generated motion often appears mechanical and unnatural, sometimes resulting in foot sliding. Despite issues with motion quality, these SINC baselines effectively capture the semantics of each motion crop since crops are denoised independently.

Analysis of the results. Our method STMC consistently performs effectively across *both* semantic and realism metrics, unlike baselines that tend to sacrifice performance in one category for the other. For example, DiffCollage achieves the best FID using MDM, but its inability to handle spatial compositions results in worse semantics than STMC across all models. Additionally, SINC baselines perform best in terms of semantics for MotionDiffuse and MDM-SMPL, but result in abrupt or unnatural transitions with FID or transition distance that is often higher than STMC. Such transitions are also readily apparent in qualitative results (see supplementary video on our webpage). It is also notable that using MDM-SMPL with STMC performs on par with MDM and MotionDiffuse, while enabling direct SMPL output and significantly reducing (by 10×) the number

of diffusion steps. Fewer steps, combined with pre-computing text embeddings, enable sampling MDM-SMPL in less than 5 seconds on average. This is a substantial improvement over MDM, which takes 4 minutes to generate motions followed by 8 min of optimization to obtain SMPL poses, on average.

While the performance of `STMC` is promising, the semantic metrics for ground truth motions indicate room for improvement. As discussed in Section 6.3.6, `STMC` is currently limited by the pre-trained diffusion model that it leverages for each motion crop; we expect improvements in these models to also boost `STMC`.

Varying the overlap size. We experiment with varying the size of the overlap for temporal stitching (corresponding to $2 * l$) and display the results in Table 6.2. We find that a smaller overlap size results in a higher transition distance. This means that the transitions may be more noticeable. However, it also leads to a more accurate match of each crop with its corresponding description, as indicated by higher per-crop semantic correctness metrics. With a larger overlap size, the transitions become smoother (i.e., lower transition distance), but this comes at the cost of reduced per-crop semantic correctness metrics.

Total overlap (s)	Per-crop semantic correctness				Realism	
	R@1 ↑	R@3 ↑	TMR-Score ↑ M2T	TMR-Score ↑ M2M	FID ↓	Transition distance ↓
0.25	30.1	51.7	0.675	0.666	0.459	1.0
0.4	29.9	51.1	0.675	0.666	0.459	1.0
0.5	30.5	50.9	0.675	0.665	0.459	0.9
0.6	30.3	50.8	0.674	0.665	0.459	0.9
0.75	28.9	50.4	0.672	0.664	0.460	0.9
1.0	28.5	49.1	0.670	0.662	0.459	0.9
1.25	28.9	48.6	0.668	0.660	0.458	0.9

Table 6.2: **Influence of the overlap size**: We report the performance of `STMC` (with MDM-SMPL) while varying the total overlap size ($2 * l$). We observe that a smaller overlap size leads to a higher transition distance but each crop matches the description better (higher per-crop semantic correctness metrics). We observe the opposite for a larger overlap size.

6.3.4 Perceptual study

We perform two separate user studies to compare `STMC` to ***`SINC`*** *with Lerp* and *DiffCollage* when using MDM. Figure 6.6 shows results of both studies, measuring human preference for motion realism and semantic accuracy. On the left, `STMC` is preferred or similar to `SINC` 66% of the time for realism and 62% of the time for semantic accuracy, with 4.2 raters judging each video on average after filtering bad responses. Compared to DiffCollage on the right, our method is preferred or similar 68% of the time for realism and 70% for semantic accuracy, with 2.8 raters judging each video after filtering. This demonstrates that `STMC` improves the motion in ways that are discernible by humans but may not be fully captured in quantitative metrics.

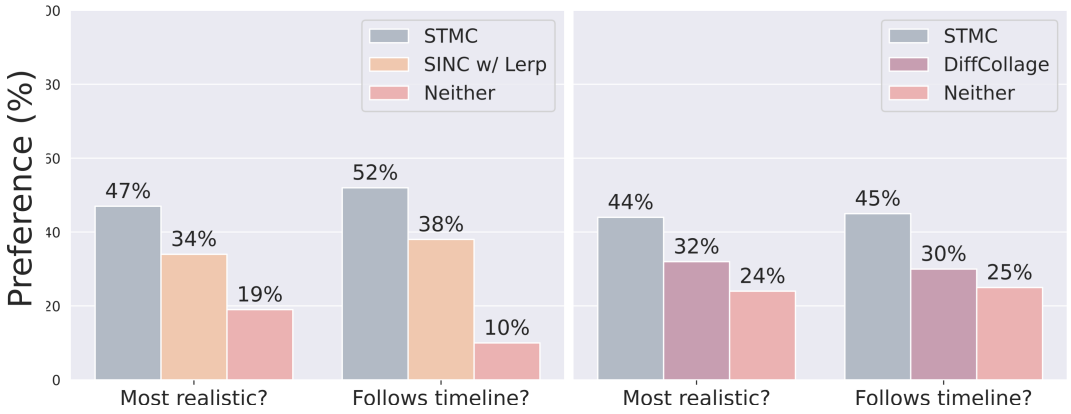


Figure 6.6: **Perception study results:** Our `STMC` method is preferred over baselines by human raters for both motion realism and semantic accuracy. (Left) Comparison against the strong `SINC` with Lerp baseline. (Right) Comparison against the DiffCollage baseline. MDM [Tevet et al. 2023] is used as the denoiser in these experiments.

6.3.5 Qualitative results

We visualize motions generated by `STMC` with MDM-SMPL in Figure 6.7, given multi-track timelines as input from our MTT dataset. The coloring follows the input text, prioritizing the newest prompt when there is an overlap across tracks. These results show that `STMC` is capable of generating realistic motions for complex multi-prompt timelines, which follow the timing and duration of the given intervals. Please see the caption for full analysis of these examples.

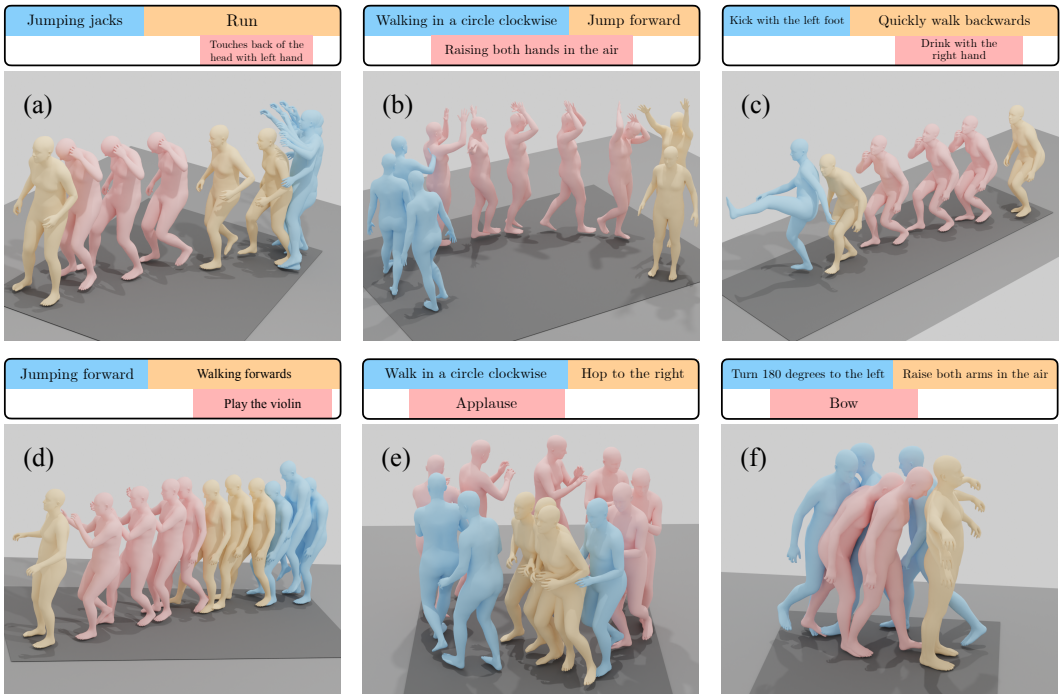


Figure 6.7: **Qualitative results:** We visualize the results of STMC with MDM-SMPL on several input timelines and color the bodies depending on their location in the timeline. We see that STMC is capable of generating realistic motions, which capture the semantics of the given text prompts with the desired timing and duration. In **(a)** and **(c)**, STMC generates motions that precisely follow the instructions, controlling a single arm while still performing another action. The accurate timing of intervals is demonstrated in **(b)** where the arms are still up in the air when transitioning from "walking" to "jumping", which is difficult to achieve with alternative methods. In **(c)** and **(d)**, we observe that STMC is capable of generating compositions that were not present in the ground truth data, such as "walking backwards while eating" or "walking while playing violin".

Video. We encourage the reader to view the supplementary video on our website to observe qualitative results in motion. We visually explain the method and show qualitative results from STMC, along with a comparison to the baselines. By looking at the generated motions in the video, we can see more clearly the differences between the baselines and STMC. In particular, although SINC w/ Lerp has good metrics overall, some motions do not look natural to the human eye.

6.3.6 Limitations

While STMC expands the capabilities of pre-trained motion diffusion models to take a multi-track timeline as input, it is also limited by the models that it relies on. For example, our proposed body-part stitching process produces spatially composed motions throughout denoising that the off-the-shelf models are not trained to robustly handle. One potential direction to ameliorate this is a more sophisticated stitching "schedule" where body parts are not combined until later in the denoising process instead of at every step. STMC also inherits the limitations of SINC, e.g., restricting overlapping motions to have compatible body part combinations.

6.4 Conclusion

In this chapter, we proposed the new problem of multi-track timeline control for text-driven 3D human motion generation. The timeline input gives users fine-grained control over the timing and duration of actions, while still maintaining the simplicity of natural language. We tackled this challenging problem using a new test-time denoising process called spatio-temporal motion collage (STMC), which enables pre-trained diffusion models to handle the spatial and temporal compositions present in timelines. Finally, extensive quantitative and qualitative evaluation demonstrated the advantage of STMC over strong baseline methods and its ability to generate realistic motions that are faithful to a multi-track timeline from the user.

Chapter 7

Discussion

This concluding chapter summarizes the contributions presented in this manuscript (Section 7.1) and outlines directions for future research (Section 7.2).

7.1 Summary of contributions

This thesis has addressed the problem of natural language control for 3D human motion generation. Our contributions are fourfold:

In Chapter 3, we have introduced our sequence-level Transformer-based VAE model, `ACTOR`, to synthesize action-conditioned human motions using the SMPL parametrization. We demonstrated that the generator can be trained with noisy human motion estimates from monocular videos. We explored different use cases in motion denoising and action recognition while providing an extensive analysis of architecture ablation and loss components.

In Chapter 4, we have proposed `TEMOS`, a model that can generate multiple and varied human motions from a given free-form text. We created a cross-modal latent space between texts and motions by leveraging a pretrained text encoder and using carefully designed training losses. We demonstrated the competitiveness of `TEMOS` over previous deterministic works and presented a comprehensive ablation study of the model's components.

In Chapter 5, we have tackled the task of motion retrieval with natural language queries. We built TMR on top of TEMOS, by incorporating a contrastive loss in the cross-modal latent space. We showed that maintaining the generation loss and adding a simple negative filtering strategy improved the model performance. We created a series of benchmarks of varying difficulty and showed a use case in zero-shot moment retrieval.

In Chapter 6, we have designed a new task of multi-track timeline control for text-driven 3D human motion generation, which gives users fine-grained control over the timing and duration of actions. We addressed this problem with STMC, a test-time method that enables pre-trained motion diffusion models to handle timeline inputs, and showed its performance against carefully crafted baselines. In addition, we trained a diffusion-based model to directly generate SMPL pose parameters.

7.2 Limitations and future work

Dataset balance. The datasets currently used are imbalanced, with a predominance of simple actions like "walking" over more complex movements. Future research could benefit from balancing the datasets to enhance model performance across a wider range of actions. One approach to achieve this would be to leverage the TMR latent space. By encoding the motions of the entire training set into latent vectors, clustering techniques can be applied to these vectors. This clustering would group the training data by similar "motion concepts". During training, random batches could then be created by sampling from each cluster, potentially improving the training of motion generation models.

Language generation branch. Both TEMOS and TMR incorporate text and motion encoders, but solely a motion decoder. Introducing a text decoder (i.e., motion captioning, as in [Guo et al. 2022b; Radouane et al. 2023; Radouane et al. 2024]) could offer interesting insights into improving TEMOS' generation capabilities and TMR's retrieval performance. One possible approach would be to add a text decoder on top of the joint motion-text embedding space, so that

TMR would become symmetric (i.e., text/motion encoders and text/motion decoders) and would operate on four reconstructions losses instead of two. This could potentially refine the latent space, improving both motion synthesis and retrieval, while adding the capability of motion captioning. Motion captioning on its own can be useful, for example for generating pseudo-ground truth data for unlabeled motion datasets.

Out-of-distribution vocabulary. The models explored in this thesis struggle to generalize to textual descriptions not present in the training dataset. For example, in the case of `TEMOS`, when presented with an action such as "playing the piano" that is absent from the KIT-ML dataset, the model defaults to generating the closest available action, such as "playing the violin." Enhancing the models' ability to generalize to out-of-distribution texts represents a significant area for future work. One potential solution would be to leverage large-scale captioned videos and extract motions using state-of-the-art 3D motion estimation methods. While these motions may be somewhat jittery, they can provide a broader range of semantic concepts compared to indoor motion capture datasets. A model can initially be trained on motion capture data to establish a foundation of realism and then be fine-tuned on the video-based dataset. This approach could improve the model's ability to handle diverse and previously unseen textual descriptions.

Physical realism. Our motion generations do not necessarily follow the physics laws, sometimes exhibiting issues such as foot-sliding or unrealistic jumping motions that fail to adhere to gravitational laws. Addressing these quality concerns through the incorporation of physics knowledge – either by integrating a foot-sliding loss, utilizing physics simulators, or embedding physics principles directly – could greatly improve motion realism.

Body shape variation. Our work uses a SMPL body model with mean body shape, neglecting individual differences in body morphology that influence motion dynamics. Addressing the challenge of body shape variation, while handling issues such as interpenetration, could lead to more personalized and realistic motion generation. A number of retargeting techniques could be used to transfer our motion generations to other individuals.

Detailed motions. Currently, the generated motions lack details, particularly in the hands, face, feet, or finegrained movements of body parts beyond the articulated joints (contracting certain muscles, expanding the stomach etc.). Actions requiring finger motions or facial expressions are not adequately represented, resulting in a loss of expressiveness and realism. Future efforts could focus on modeling these finer details to enrich the overall quality of generated motions potentially by gathering suitable data.

Diversity. The diversity of the generated motions is sometimes limited, which is mainly due to the lack of diversity in the training data itself. Investigating synthetic data augmentation methods could be a promising way of creating more diverse training datasets, in return improving the diversity of motion synthesis. One approach to data augmentation could involve identifying body parts that are irrelevant to the action performed and freely moving them. While this approach has potential, it is non-trivial how to design the framework.

Motion estimation enhancement. The latent spaces developed in this work have potential for use as priors in motion estimation and action recognition tasks. For example, using a motion prior for "riding a bike" could increase the accuracy of estimation of occluded body parts, such as the rider's leg.

Human-environment interaction. Throughout this thesis, we have focused on single-human motion generation in isolation, that is without the interaction with other humans or objects within the environment. For instance, with the "sitting down" action, the generated humans appear to sit on an invisible chair, while for the "drinking" action, there is nothing to drink from. Future work could explore incorporating environmental interactions to generate more contextually rich and realistic human motions.

Virtual characters. In the future, I see virtual characters or robots act and interact naturally like humans in dynamic environments. This setup requires generating motions in real-time. In this thesis, we developed models for generating human motions offline. I believe an online motion generation model could leverage the knowledge from an offline model to adapt to new situations, much like how humans continuously adjust to changing circumstances.

Conclusion. The future of natural language control for 3D human motion synthesis is bright and full of potential. While the field still faces challenges, growing interest and collaboration among researchers is paving the way for rapid progress. These developments will revolutionise applications in entertainment, virtual reality and robotics, making virtual interactions more lifelike and immersive. As the field continues to grow, the quality and variety of synthesized motion will improve, opening up new possibilities for creative and practical applications.

Annex

Annex A

TEACH: Temporal Action Compositions for 3D Humans

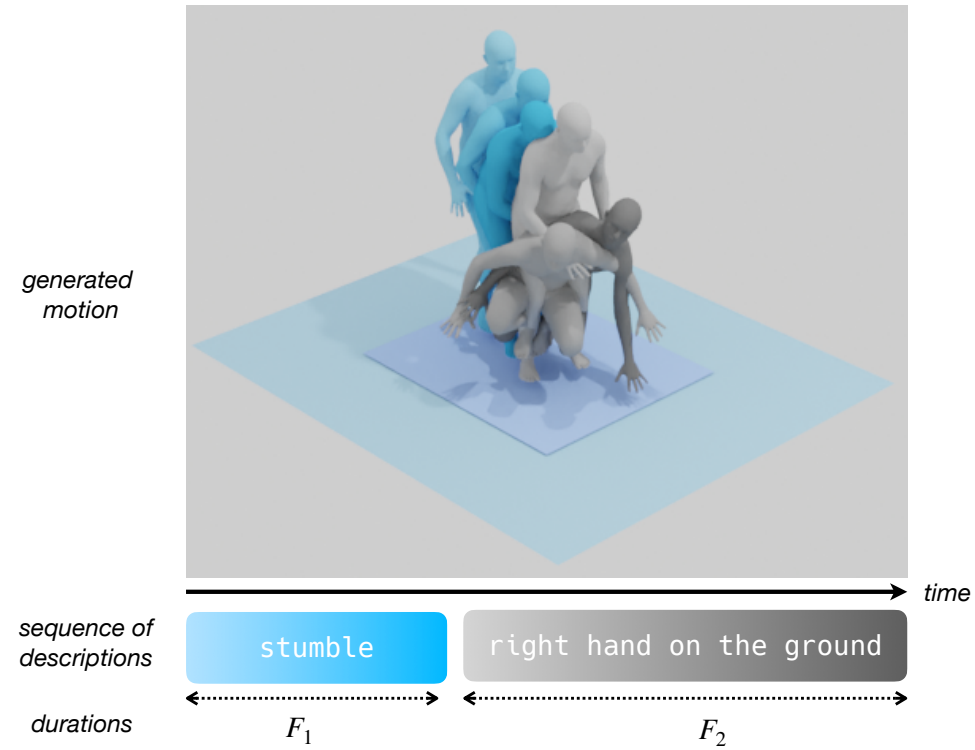


Figure A.1: **Goal:** Given a sequence of descriptions and durations as input, our goal is to generate a 3D human motion respecting the instruction and achieving temporal action compositionality. We design a recursive approach, TEACH, that can produce a variable number of actions given a stream of textual prompts. Note that the color saturation is aligned with the progress of each action.

This first chapter of the Annex extends Chapter 4. Instead of a single text as input, it takes a series of natural language descriptions, and the goal is to generate 3D human motions that correspond semantically to the texts, and follow the temporal order of the instructions.

In particular, our goal is to enable the synthesis of a series of actions, which we refer to as temporal action composition. The current state of the art in text-conditioned motion synthesis only takes a single action or a single sentence as input. This is partially due to lack of suitable training data containing action sequences, but also due to the computational complexity of their non-autoregressive model formulation, which does not scale well to long sequences. In this work, we address both issues. First, we exploit the recent BABEL motion-text collection, which has a wide range of labeled actions, many of which occur in a sequence with transitions between them. Next, we design a Transformer-based approach that operates non-autoregressively within an action, but autoregressively within the sequence of actions. This hierarchical formulation proves effective in our experiments when compared with multiple baselines. Our approach, called `TEACH` for "TEmporal Action Compositions for Human motions", produces realistic human motions for a wide variety of actions and temporal compositions from language descriptions. To encourage work on this new task, we make our code available for research purposes, and provide an additional video at `teach.is.tue.mpg.de`.

A.1 Introduction

The generation of realistic 3D human motions has applications in virtual reality, the games industry, and any application that requires motion capture data. Recently, controlling 3D human motion synthesis with semantics has received increasing attention [Guo et al. 2020; Ghosh et al. 2021; Petrovich et al. 2021; Petrovich et al. 2022]. The task concerns inputting semantics in the form of categorical actions, or free-form natural language descriptions, and outputting a series of 3D body poses. In this work, we address the latter, i.e., text-conditioned motion generation, which is more flexible compared to pre-defining a set of categories. More specifically, our goal is to animate a sequence of

actions given a sequence of textual prompts. Generating such *temporal action compositions* has not previously been studied.

Humans move in complex ways that involve different simultaneous and/or sequential actions. Hence, *compositionality* in time must be modeled to generate everyday motions that contain a series of different actions and that last longer than a few seconds. Compositionality in space, i.e., simultaneous actions, is another interesting direction outside the scope of this work. Generative models are popular for synthesizing images conditioned on textual descriptions [Ramesh et al. 2022; Saharia et al. 2022]. Their success can be largely attributed to massive training datasets. The same breakthrough has not happened for 3D human motion generation due to lack of motion-text data. Standard benchmarks for the text-conditioned motion synthesis task (e.g., the KIT Motion-Language dataset [Plappert et al. 2016]) are limited in the vocabulary of actions and the number of motion sequences. In this work, we use the recently released textual annotations of the BABEL dataset [Punnakkal et al. 2021], providing English descriptions for the AMASS motion capture collection [Mahmood et al. 2019]. This dataset is both larger and more diverse than previous datasets. While previous work uses BABEL for its categorical action annotations (60/120 classes) focusing mostly on classification settings [Tevet et al. 2022] or motion generation [Song et al. 2023], we directly train with its free-form language descriptions, which have not been used before. Thus, our generated motions cover a significantly wider variety of actions compared to the state of the art [Petrovich et al. 2022].

Recently, `TEMOS` [Petrovich et al. 2022] established a new baseline in text-conditioned 3D motion synthesis using Transformer-based VAEs [Petrovich et al. 2021] and pretrained language models [Sanh et al. 2019] to sample realistic motions corresponding to a text input. This model is limited in several ways besides being trained on the KIT data. `TEMOS` is not directly applicable for our task of generating a *sequence* of actions. While its non-autoregressive formulation generates high-quality motions, the approach does not readily scale to long sequences of multiple motions due to the quadratic time complexity of Transformers. Moreover, to embed complex sequences of actions in the latent space would require seeing a combinatorial number of action combinations during training. With existing training data, generalization to new sequences

would be challenging. In our work, we combine the best of both worlds, by designing an iterative model that generates one motion per action at a time, by conditioning on the previous motion. Within each iteration, we keep the non-autoregressive action generation approach, which probabilistically generates diverse and high-quality motions (see Figure A.1). We experimentally show that our iterative method compares favorably against baselines that jointly or independently generate pairs of actions in a single shot (Figure A.4).

One of the key challenges in synthesizing long action sequences given a stream of textual prompts is how to ensure *continuity* within the transitions between actions. Independently generating one motion per action would not guarantee temporal smoothness. In our framework, we find that encoding the next action conditioned on the last few frames of the previous action is a simple and effective solution. To account for any remaining discontinuities still present with this solution, we apply spherical linear interpolation (Slerp) over a short time window. Note that our approach treats transitions significantly better than a baseline that uses Slerp to interpolate between independently generated actions.

Our contributions are the following: (i) We introduce and establish a new benchmark for temporal action composition of 3D motions on the BABEL dataset; (ii) We design a new hybrid neural network model, `TEACH` (TEmporal Action Composition for 3D Humans), that addresses the limitations of previous state of the art by iteratively generating infinitely many actions with smooth transitions; (iii) We obtain promising results for text-to-motion synthesis from a large-vocabulary of actions.

A.2 Method

After defining the task (Section A.2.1), we present the `TEACH` architecture and explain the different baselines and architectural components of our method (Section A.2.2). Finally, we describe the training procedure and the losses (Section A.2.3).

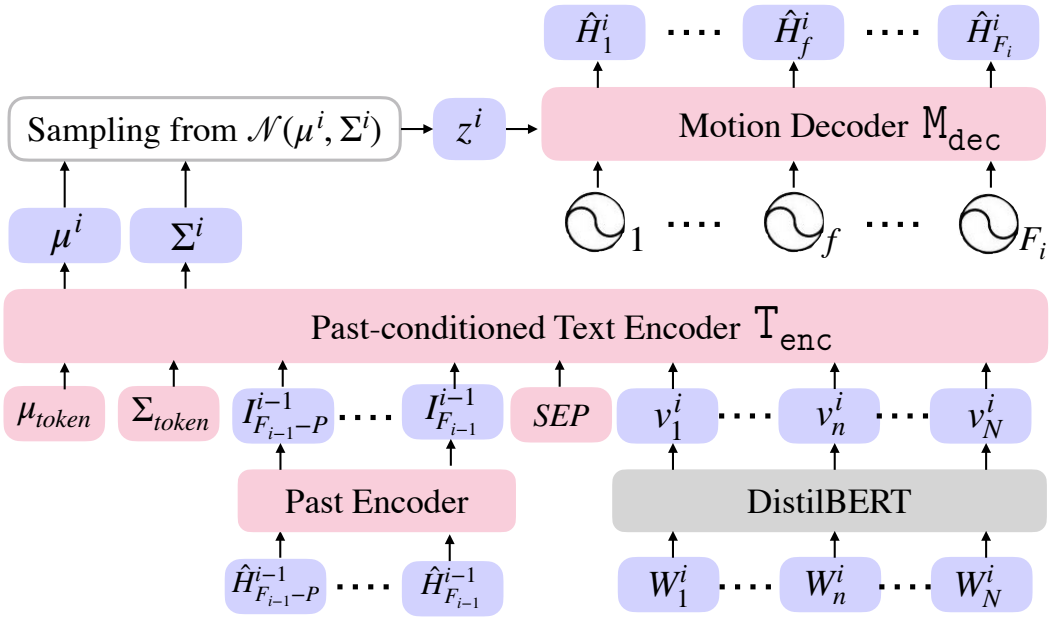


Figure A.2: **Method overview:** Our TEACH model is a variational encoder-decoder neural network. The current text instruction and the past frames are encoded by the corresponding encoders and are fed to T_{enc} along with the additional tokens. T_{enc} produces the distribution parameters from which the latent vector is sampled and given to the decoder to generate a sequence of 3D human poses. In this figure, we omit the motion encoder for simplicity.

A.2.1 Task definition

Starting from a sequence of instructions in natural language (e.g., in English), our goal is to generate smooth and realistic human motions that correspond to the instructions. Here, we demonstrate results for pairs or triplets of actions but our model can autoregressively generate an arbitrary sequence of actions given the respective action descriptions. During training, TEACH takes as input: a sequence of English prompts $S_1, \ldots, S_K$, where each phrase corresponds to an action description and consists of a sequence of words $S_i = W^i_1, \ldots, W^i_N$, *e.g.* "turn to the right", "walk forward", "sit down", etc. and a 3D human motion sequence. Each sequence consists of poses, $H_1, \ldots, H_F$, parametrized by the SMPL body model [Loper et al. 2015]. In this work, we follow the representation described in [Petrovich et al. 2022] that converts SMPL parameters to a $6D$ rotation representation [Zhou et al. 2019] together with root translation. Moreover, we use the same normalization and canonicalization process as in [Petrovich et al. 2022].

A.2.2 Architecture

Our architecture is inspired by TEMOS [Petrovich et al. 2022]. We use the same language encoder (DistillBERT) and motion encoder. We omit the details of the motion encoder as it is the same used in TEMOS. However, TEMOS is constrained to output a sampled motion given a language description, without being able to handle sequences of actions. Thus, we design a new text encoder architecture, which includes a Past Encoder (PC) that provides our method with the context of the previous action when generating the second action in each pair. For the first motion in each pair, we disable the Past Encoder and only use the learnable tokens and the encoded text. A separation token is used to facilitate disambiguation of motion and text modalities in the model [Devlin et al. 2019]. As illustrated in Figure A.2, we encode the current text instructions $W_1^i, \dots, W_N^i$ using a pre-trained, frozen text model (DistilBERT) into text features $\mathrm{v}_1^i, \dots, \mathrm{v}_N^i$. Moreover, the last P frames of the previous generated motion, $\widehat{H}^{i-1}_{F_{i-1}-P:F_{i-1}}$, are encoded into motion features $I_{F_{i-1}-P:F_{i-1}}$ (Past Encoder). Then, we combine the features from the previous action, $I^{i-1}_{F_{i-1}-j}, j \in \mathbb{N}$, and the current text features along with learnable tokens (μ_{token}, Σ_{token} and SEP), and pass them as inputs to the Past-conditioned Text-Encoder, which generates the distribution parameters μ^i and Σ^i. μ^i and Σ^i are treated as parameters of a Gaussian distribution, from which we sample and decode the final motion. Next we explain each module separately.

Past-conditioned Text Encoder. We first encode the natural language descriptions with a frozen DistilBERT [Sanh et al. 2019] which takes as input the current text instruction $S_i = W_1^i, \dots, W_N^i$ and outputs text features $\mathrm{v}_1^i, \dots, \mathrm{v}_N^i$. We use a Transformer encoder architecture to encode the past motion corresponding to the last P frames of the previous action. The past motion $\hat{H}^{i-1}_{F_{i-1}-P}, \dots, \hat{H}^{i-1}_{F_{i-1}}$ is transformed into pose features $I^{i-1}_{F_{i-1}-P}, \dots, I^{i-1}_{F_{i-1}}$. Finally, we use a transformer encoder (the Past-conditioned Text Encoder module T_{enc} to jointly encode the past motion features and the current text features into μ^i and Σ^i, parameters of a Gaussian distribution. This network takes as extra inputs, the μ_{token} and Σ_{token} as in ACTOR [Petrovich et al. 2021], and a special token (SEP) to separate both modalities. From the Gaussian distribution $\mathcal{N}(\mu^i, \Sigma^i)$, we sample a latent vector z^i. For the first motion we

disable PC since there is no previous motion.

Motion decoder. We use the same decoder architecture as in TEMOS [Petrovich et al. 2022], which generates a sequence of poses from a single embedding. This Transformer-based motion decoder takes the current latent vector z^i and F_i positional encodings (in the form of sinusoidal functions) as input, and generates the sequence of human motions.

Baselines. As there is no prior work that explicitly deals with sequences of actions, we design several baselines using TEMOS, Slerp [Shoemake 1985], and geometric transformations. Note that our work is different from the action-driven motion prediction proposed concurrently in [Mao et al. 2022], as we deal with full motion generation and free-form language descriptions and do not explicitly use only pairs formed by transitions. Specifically (Section A.4), we employ two baselines: "Independent", which is based on TEMOS and is trained on single action segments, and "Joint", which is also based on TEMOS, but takes as input the both motions (i.e., concatenation of the respective segments) and the corresponding language labels separated with a comma. For the case of Independent, the generated motions are fused into a pair of actions by: (1) aligning the last generated frame of the first action with the first frame of the second action by orientation and translation and (2) applying spherical interpolation to fill in the remaining transition between the two actions.

A.2.3 Training

Data handling. BABEL consists of language descriptions and action categories for the majority of sequences in AMASS [Mahmood et al. 2019]. Each sequence is separated in segments that can overlap without any constraint, except that all the frames of the sequence must be labeled. To train the Independent baseline, we use the training set segments from BABEL. Furthermore, we extract pairs of actions to train the remaining models. To achieve this, we process each sequence and, for each segment, $s_i = [t_s^i, t_e^i]$, we calculate all the segments s_j : $s_j \cap s_i \neq \varnothing, s_j \nsubseteq s_i, s_j \nsupseteq s_i$, except if a segment is a "transition". We think of transitions happening *between* actions so a transition and an action cannot form a pair. Simply, we use all the segments that are not a superset or subset

of each other and have an overlap. Next, if a segment is connected to another segment via a transition (i.e., the same transition overlaps with both), that triple is considered a pair of actions. For the rest of the cases, the pairs are formed by the segment overlaps and not transitions.

A training iteration consists of two forward passes. The first action, corresponding to sentence description S_1, and the given length F_1, will produce μ^1, Σ^1 and the generated motion $\hat{H}^1_1, \dots \hat{H}^1_{F_1}$. Then, given the second instruction S_2, the given length F_2, and the P last frames of the previous generated motion, we produce μ^2, Σ^2 and the generated motion of the second segment $\hat{H}^2_1, \dots \hat{H}^2_{F_2}$. We do one backward pass, which optimizes the reconstruction loss and the KL loss on the two segments jointly.

Reconstruction loss. From the two forward passes, we generate the motions $\widehat{H}^1_{1:F_1}$ and $\widehat{H}^2_{1:F_2}$. We enforce them to be close to the corresponding ground truth motions $H^1_{1:F_1}$ and $H^2_{1:F_2}$ via the following loss terms:

$$\mathcal{L}_R = \mathscr{L}(H^1_{1:F_1}, \widehat{H}^1_{1:F_1}) + \mathscr{L}(H^2_{1:F_2}, \widehat{H}^2_{1:F_2}), \tag{A.1}$$

where $\mathscr{L}$ is the smooth L1 loss.

KL loss. By using the notation $\phi^i = \mathcal{N}(\mu^i, \Sigma^i)$, and $\psi = \mathcal{N}(0, I)$, this loss regularizes the two Gaussian distributions ϕ^1 and ϕ^2 to be close to ψ as in the VAE formulation. We minimize the two Kullback-Leibler (KL) divergences

$$\mathcal{L}_{KL} = KL(\phi^1, \psi) + KL(\phi^2, \psi). \tag{A.2}$$

We also use the same additional KL losses as `TEMOS`, to enforce the latent vectors to follow the same distributions and the same L_1 loss to keep them as close as possible. We omit them from the description for simplicity and to highlight our technical contributions. The total loss is a weighted sum of the two terms: $\mathcal{L} = \mathcal{L}_R + \lambda_{KL}\mathcal{L}_{KL}$. In practice, we use $\lambda_{KL} = 10^{-5}$ as in `TEMOS` [Petrovich et al. 2022] and `ACTOR` [Petrovich et al. 2021].

Implementation details. For both the Past-Conditioned Text Encoder and the Past Encoder, we use a Transformer encoder model with 6 layers and 6 heads, a dropout of 0.1 and a feed-forward size of 1024. The latent vector dimension

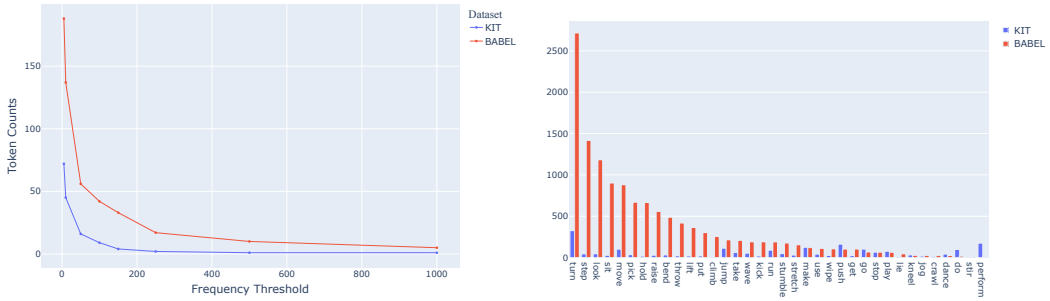


Figure A.3: **BABEL vs KIT:** We provide a comparative analysis of the amount of data and the vocabulary of verbs. On the top, the number of tokens (i.e. different words) in each dataset is plotted against various frequency thresholds, i.e. the number of words that appear at least freq. threshold times. We see that BABEL consistently has at least twice as many tokens as KIT. On the bottom, the verb histogram shows that BABEL has more samples across a wide range of actions. Note that there are differences in how the datasets label actions with generic words like "do" and "perform" being common in KIT and rare in BABEL, which is more specific.

is 256. The whole model is trained with the AdamW optimizer [Loshchilov et al. 2019] with a fixed learning rate of 10^{-4} with a batch size of 32 or 16. Both during training and test time, we use ground truth durations (F_i). We also use Slerp [Shoemake 1985] and alignment between the first and the second action for `TEACH` as well as for the Independent baseline. We apply Slerp to interpolate for 8 frames at the beginning of the second motion which includes the transition.

A.3 Experiments

We first describe the dataset (Section A.3.1) and evaluation metrics (Section A.3.2) used in our experiments. We then report our main results by comparing our method with multiple baselines (Section A.3.3). Next, we present an ablation study to investigate the contribution of motion interpolation (Section A.3.4) and different numbers of past frames used by PC (Section A.3.5). Finally, we provide qualitative results (Section A.3.6) and a discussion on limitations (Section A.3.7).

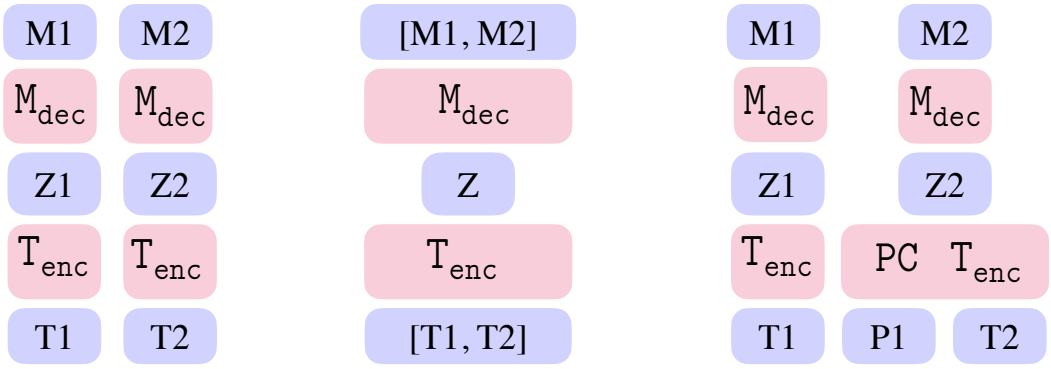
A.3.1 BABEL dataset

In our work, we train and evaluate on BABEL [Punnakkal et al. 2021], which provides textual descriptions for the motions in the AMASS collection [Mahmood et al. 2019]. In particular, we use the processed text version (lemmatized etc. as opposed to the raw version which is also provided). We do not use the categorical action labels. In total there are 10881 motion sequences, with 65926 textual labels and the corresponding segments. The unique property of BABEL is that it has annotated segments that overlap in each sequence, which allows us to investigate generation of a *sequence of actions*. In contrast, a textual label in KIT [Plappert et al. 2016] covers the entire sequence. Moreover, KIT is smaller both in terms of the number of motion sequences and the number of actions. Figure A.3 shows the distribution of verbs according to the most frequent verbs in BABEL.

Refer to the supplementary material of the main paper for additional analysis of KIT's most frequent verbs compared with BABEL and also analysis about other part-of-speech categories. There are approximately 5.7k and 23.4k pairs in the validation and training sets respectively. We consider pairs of actions for simplicity but `TEACH` is applicable to sequence of actions of arbitrary length. Note that, we do not use "t-pose" or "a-pose" actions during training. We use transitions only to identify possible pairs of actions. During training, in the case of segment overlap, we uniformly distribute the overlapping frames across the two segments that constitute the pair. Also, note that the majority of the pair data ($\sim 70\%$) is created by overlapping segments and not by transitions. In the case of a transition, we concatenate the transition with the second segment.

Methods	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
(a) Independent	0.729	0.707	0.169	0.770	0.255	0.253	0.016	0.267
(b) Joint	0.790	0.773	0.163	0.832	0.306	0.305	0.014	0.317
(c) Past-conditioned (`TEACH`)	0.674	0.654	0.159	0.717	0.222	0.220	0.014	0.234

Table A.1: **Comparison against baselines on pairs of actions:** We benchmark the 3 different approaches on pairs of BABEL [Punnakkal et al. 2021]. As we can see `TEACH` outperforms Joint and Independent baselines in all the metrics.



(a) Independent (b) Joint (c) Past-Conditioned

Figure A.4: **Variants:** We illustrate the baselines for independent single-action training (a) and joint two-action training (b). Our method on the other hand is recursive, and is conditioned on the past motion (c). T1 and T2 denote the sequence of two textual descriptions. M stands for motion, and Z stands for the latent vector.

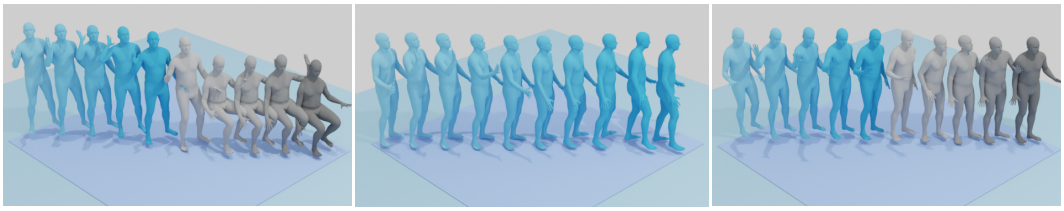
A.3.2 Evaluation metrics

We follow the evaluation metrics employed by [Ghosh et al. 2021; Petrovich et al. 2022], namely Average Positional Error (APE) and Average Variational Error (AVE), measured on the root joint and the rest of the body joints separately. Mean *local* and *global* refer to the joint position in the local (with respect to the root) or global coordinate systems, respectively. As in [Petrovich et al. 2022], we sample one random motion generation from our variational model and compare it against the ground truth motion corresponding to the test description. While we quantitatively evaluate on pairs of actions, we qualitatively show the ability of our model to generate two or more actions in Figure A.6 and the supplementary video.

A.3.3 Comparison with baselines

Here, we first describe the baselines we created by adapting `TEMOS` [Petrovich et al. 2022] to the action sequence synthesis task without any architectural changes. Figure A.4 summarizes the two variants (a) independent and (b) joint training.

Independent training, in Figure A.4 (a), refers to inputting a single text and



(a) Independent (b) Joint (c) TEACH

Figure A.5: **Qualitative comparison:** We show an illustrative example for (a) Independent, (b) Joint and (c) `TEACH` for the sequence of actions [`wave the right hand`, `raise the left hand`]. While the individual waving and raising hand actions are correctly generated, the single-action independent baseline (a) transitions from standing to sitting incoherently as the next action is not conditioned on the past. Joint baseline (b) on the other hand, waves with the right hand but does not raise the left one, probably because such an action combination was not present in the training set. On the other hand, `TEACH` learns about both single action variation and and autoregressive transitions between actions, and thus completes both actions naturally. Note that, while these motions are performed in place, we artificially translate each pose to show the motion frame-by-frame such that the transition and action details are easier to see.

outputting a single motion, as is the case for `TEMOS`. To adapt this model to a sequence of actions, we generate the two actions and perform an interpolation operation (i.e., Slerp) to obtain smooth transitions between the independently generated motions. However, a naive interpolation results in poor transitions since the second motion may start at a different global location, with a different global rotation. To account for this mismatch, we translate the root of the second motion to have the same x,y coordinates as the first motion. Then, we rotate it to match the first motion's global orientation. The advantage of this model is its ability to scale up to any number of action compositions. However, despite the interpolation, we observe unnatural motions due to large changes between body poses during transitions. For example in Figure A.5, the model generates two motions that are not compatible in terms of pose, creating an unrealistic transition. This is expected as the independent baseline has no notion of the previous motions.

The joint training, in Figure A.4 (b), is another alternative to extend `TEMOS` to multiple descriptions without further modifications. We simply combine the sequence of descriptions into a single text with a comma punctuation in between, and train the model with pairs of motions corresponding to consecutive actions.

The advantage of this model is the ability to produce smooth motions, including the transitions. However, the major disadvantage concerns scalability. Due to quadratic complexity with respect to the motion duration, the joint training does not scale well to a large number of actions. Moreover, it would require many action combinations, i.e., a concatenation of more than pairs of actions, at training to produce a variable number of actions. Such data are not easy to capture, making it challenging to train such a model. Finally, it may be difficult to generalize to unseen action combinations. In our experiments, we train this model with 2-action pairs (which is a relatively easy setting compared to more actions).

In contrast to independent and joint training, our model (Figure A.4 (c)) is recursive and the future action is conditioned on the previous action. In Table A.1, we summarize the performance of these three variants on the BABEL validation set. Our past-conditioned `TEACH`, which uses the last 5 frames from the previous action, outperforms the baselines. Due to the difficulty of quantitative evaluation of generative models, we also rely on qualitative comparisons, provided in the supplementary video. An illustration of our arguments can be seen in Figure A.5.

Methods	Transition Dist.		Average Positional Error ↓				Average Variance Error ↓			
	w/ align.	w/out align.	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
Independent (no Slerp)	0.151	0.177	0.762	0.740	0.170	0.805	0.255	0.253	0.016	0.267
`TEACH` (no Slerp)	0.107	0.122	0.677	0.658	0.159	0.722	0.227	0.225	0.015	0.239
Independent	n/a	n/a	0.729	0.707	0.169	0.770	0.255	0.253	0.016	0.267
`TEACH`	n/a	n/a	0.674	0.654	0.159	0.717	0.222	0.220	0.014	0.234

Table A.2: **Effect of Slerp:** We measure transition distance for generated samples given all the test set pairs. We define transition distance as the Euclidean distance between the last frame of the first action and the first frame of the second action, calculated on joint positions, when the last pose of the first action is aligned with the first pose of the next action and when it is not. `TEACH` better captures the transition between the two actions compared to the previous-action-agnostic `TEMOS`. Moreover, the Independent baseline cannot be benchmarked without orienting and aligning the poses as it is trained on single actions that are canonicalized to face in the forward direction.

P	Average Positional Error ↓				Average Variance Error ↓			
	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
1	0.725	0.704	0.160	0.766	0.222	0.220	0.015	0.234
5	0.674	0.654	0.159	0.717	0.222	0.220	0.015	0.234
10	0.718	0.698	0.157	0.759	0.238	0.237	0.015	0.250
15	0.719	0.699	0.163	0.761	0.238	0.236	0.014	0.250

Table A.3: **Ablation on the number of past frames:** Here, we change the number of past frame, while keeping the other training settings identical and report the different metrics. We observe the best performance when using 5 past frames.

A.3.4 Effect of interpolating the action transitions

As explained in Sections A.2 and A.3.3, we use Slerp interpolation between actions both for the independent training baseline, and our method. We justify the use of such interpolation with the experiment in Table A.2. Removing Slerp causes discontinuities which are easier to see visually from our supplementary video. However, the discontinuity is higher for the independent generations than in `TEACH`. To measure the degree of discontinuities, we report the average transition distance, i.e., the Euclidean distance between the two body poses corresponding to the last frame of the previous action, and the first frame of the next action. We see a clear decrease in discontinuity with `TEACH` (0.107 vs 0.151 m), even when the bodies are aligned. Moreover, we measure the same metric in the absence of alignment for the global orientation of the second motion (see Section A.3.3). We see that this alignment step is crucial for the independent baseline, as the transition distance compared to `TEACH` is even worse if we do not apply any alignment at all (0.177 to 0.122), demonstrating that `TEACH` models generate smoother transitions than the baseline.

A.3.5 Past conditioning duration

In Table A.3, we investigate the influence of the hyperparameter P, the number of frames from the past motion to input to the past-conditioned text encoder. While the performance is similar across 1, 5, 10, or 15 frames, we observe a slight improvement when using 5 frames as opposed to 1 frame, potentially because a single frame does not capture enough past information. However,

further increasing the number of past frames does not improve the results.

A.3.6 Qualitative analysis

We present qualitative motion generation results in Figure A.6. In contrast to previous work that trains models on the KIT dataset [Plappert et al. 2016], our model is able to go beyond locomotive motions, and covers a wider variety of actions, such as `right hand on the ground`. Finally, we show examples of more than 2 actions in the last row of Figure A.6. We refer to the supplementary video on our webpage for viewing the motions, providing analyses of the effect of interpolation, presenting motions beyond pairs of actions and failure cases. Moreover, we give a quick overview of our method and baselines and explain visually how the alignment of the different actions is performed;

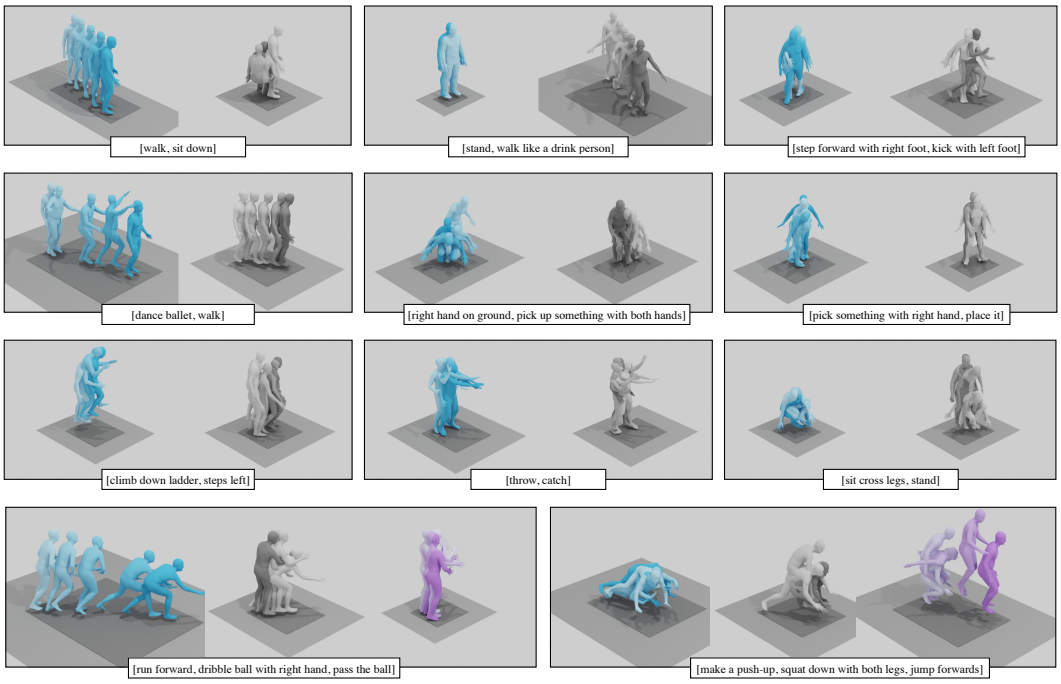


Figure A.6: **Qualitative results:** In the first 3 rows, we visualize `TEACH` results for pairs of actions. We see how `TEACH` goes beyond walking and that in all the cases there are two actions being performed. Even fine-grained sequences of action like 'step forward with the right foot' and 'kick with the left foot' are generated accurately. In the final row, we show triplets of actions. We use a separate image for each action in the sequence to make the performed action clearly visible. We denote the ending of the first action with the most saturated version of cyan, while the starting of the second is the less saturated version of gray.

A.3.7 Limitations

Our work does not come without limitations. TEACH is susceptible to acceleration peaks when transitioning from the first action to the second one. There is still the need to apply Slerp to smooth out these discontinuities but, as we see in Table A.2, the starting/ending poses of the two actions are not far away. This behavior may be also attributed to the variational nature of the model, which makes it difficult to precisely match the previous motion without any explicit pose-level autoregressive constraints. Moreover, BABEL has a lot of overlapping segments of actions which makes it difficult sometimes to have a visually "clear" sequence of actions, as some actions might mix with others.

A.4 Conclusions

We presented a new task on motion generation from a sequence of textual prompts, which we refer to as action compositions in time. We established a new benchmark on the BABEL dataset for this task, and explored a variety of strong baselines, including independently or jointly training pairs of actions. Our recursive approach, TEACH, improves over the baselines quantitatively, while addressing the past limitations by allowing variable numbers of actions and producing fewer discontinuities at transitions. While we obtain promising results within this new direction, there is still room for improvement. Motion realism can be improved and contact with the world could be explicitly modeled. Here, we assume that the character does not know what it will do in the future; that is, it only looks backwards in time. In contrast, humans have goals and know what they will do next. This knowledge about the future can affect the present. Future work can explore such "looking ahead" to better generate realistic sequences of actions. We hope that TEACH will encourage further research on combining language and 3D motion, much like the field has done with language and 2D images [Ramesh et al. 2022; Saharia et al. 2022].

Annex B

SINC: Spatial Composition of 3D Human Motions for Simultaneous Action Generation

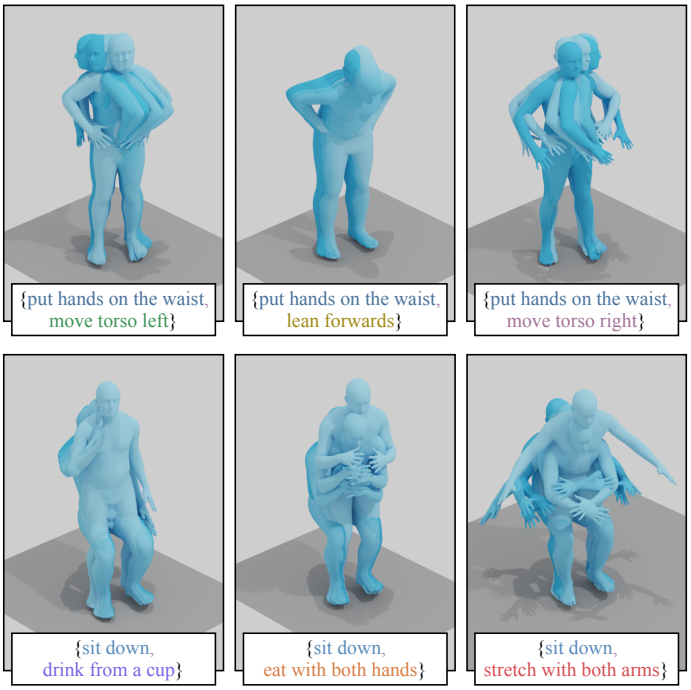


Figure B.1: **Goal:** We demonstrate the task of *spatial compositions* in human motion synthesis. We generate 3D motions for a pair of actions, defined by a pair of textual descriptions. Here, we provide six sample input-output illustrations from our model. For example, we input the set of actions {'put hands on the waist', 'move torso left'} and generate one motion that simultaneously performs both.

This second chapter of the appendix also extends Chapter 4. Instead of a single text as input, our goal is to synthesize 3D human motions given textual inputs describing simultaneous actions, for example 'waving hand' while 'walking' at the same time. We refer to generating such simultaneous movements as performing *spatial compositions*.

In contrast to *temporal compositions* (Annex A) that seek to transition from one action to another, spatial compositing requires understanding which body parts are involved in which action, to be able to move them simultaneously. Motivated by the observation that the correspondence between actions and body parts is encoded in powerful language models, we extract this knowledge by prompting GPT-3 with text such as "what are the body parts involved in the action <action name>?", while also providing the parts list and few-shot examples. Given this action-part mapping, we combine body parts from two motions together and establish the first automated method to spatially compose two actions. However, training data with compositional actions is always limited by the combinatorics. Hence, we further create synthetic data with this approach, and use it to train a new state-of-the-art text-to-motion generation model, called `SINC` ("SImultaneous actioN Compositions for 3D human motions"). In our experiments, we find that training with such GPT-guided synthetic data improves spatial composition generation over baselines. Code, models and an illustrative video are publicly available at sinc.is.tue.mpg.de.

B.1 Introduction

Text-conditioned 3D human motion generation has recently attracted increasing interest in the research community [Athanasiou et al. 2022; Guo et al. 2022a; Petrovich et al. 2022], where the task is to input natural language descriptions of actions and to output motion sequences that semantically correspond to the text. Such controlled motion synthesis has a variety of applications in fields that rely on motion capture data, such as special effects, games, and virtual reality. While there have been promising results in this direction, *fine-grained* descriptions remain out of reach. Consider the scenario in which a movie

production needs a particular motion of someone jumping down from a building. One may generate an initial motion with one description, and then gradually refine it until the desired motion is obtained, e.g., {'jumping down', 'with arms behind the back', 'while bending the knees'}. State-of-the-art methods [Petrovich et al. 2022; Chen et al. 2023] often fail to produce reasonable motions when conditioned on fine-grained text describing multiple actions. In this work, we take a step towards this goal by focusing on the *spatial composition* of motions. In other words, we aim to generate one motion depicting multiple simultaneous actions; see Figure B.1. This paves the way for further research on fine-grained human motion generation.

Previous work [Lin et al. 2018a; Ahuja et al. 2019; Ghosh et al. 2021; Petrovich et al. 2022] initially explored the text-conditioned motion synthesis problem on the small-scale KIT Motion-Language dataset [Plappert et al. 2016]. Recently, work [Athanasiou et al. 2022; Guo et al. 2022a] has shifted to the large-scale motion capture collection AMASS [Mahmood et al. 2019], and its language labels from BABEL [Punnakkal et al. 2021] or HumanML3D [Guo et al. 2022a]. In particular, similar to this work, `TEACH` [Athanasiou et al. 2022] focuses on fine-grained descriptions by addressing temporal compositionality, that is, generating a sequence of actions, one *after* the other. We argue that composition in time is simpler for a model to learn since the main challenge is to smoothly transition between actions. This does not necessarily require action-specific knowledge, and a simple interpolation method such as Slerp [Shoemake 1985] may provide a decent solution. On the other hand, there is no such trivial solution for compositions in *space*, since one needs to know action-specific body parts to combine two motions. If one knows that 'waving' involves the hand and 'walking' involves the legs, then compositing the two actions can be performed by cutting and pasting the hand motion into the walking motion. This is often done manually in the animation industry.

To automate this process, we observe that pretrained language models such as GPT-3 [Brown et al. 2020] encode knowledge about which body parts are involved in different actions. This allows us to first establish a spatial composition baseline (analogous to the Slerp baseline for temporal compositions); i.e., independently generating actions then combining with heuristics. Not surprisingly, we find that this is suboptimal. Instead, we use the synthesized

compositions of actions as additional training data for a text-to-motion network. This enriched dataset enables our model, called SINC ("SImultaneous actioN Compositions for 3D human motions"), to outperform the baseline. Our GPT-based approach is similar in spirit to work that incorporates external linguistic knowledge into visual tasks [Yu et al. 2017; Wang et al. 2022b; Brooks et al. 2023].

While BABEL [Punnakkal et al. 2021] and HumanML3D [Guo et al. 2022a] have relatively large vocabularies of actions, they contain a limited number of *simultaneous* actions. A single temporal segment is rarely annotated with multiple texts. For example, BABEL contains only roughly 2.5K segments with simultaneous actions, while it has ∼25K segments with only one action. This highlights the difficulty of obtaining compositional data at scale. Moreover, for any reasonably large set of actions, it is impractical to collect data for all possible pairwise, or greater, combinations of actions such that there exists no unseen combination at test time [Yu et al. 2017; Yang et al. 2018]. With existing datasets, it is easy to learn spurious correlations. For example, if waving is only ever observed by someone standing, a model will learn that waving involves moving the arm with straight legs. Thus generating waving and sitting would be highly unlikely. In our work, we address this challenge by artificially creating compositional data for training using GPT-3. By introducing more variety, our generative model is better able to understand what is essential to an action like 'waving'.

Our method, SINC, extends the generative text-to-motion model TEMOS [Petrovich et al. 2022] such that it becomes robust to input text describing more than one action, thanks to our synthetic training. We intentionally build on an existing model to focus the analysis on our proposed synthetic data. Given a mix of real single actions, real pairs of actions, and synthetic pairs of actions, we train a probabilistic text-conditioned motion generation model. We introduce several baselines to measure sensitivity to the model design, as well as to check whether our learned motion decoder outperforms a simpler compositing technique (i.e., simply using our GPT-guided data creation approach, along with a single-action generation model). We observe limited realism when compositing different body parts together, and need to incorporate several heuristics, for example when merging motions whose body parts overlap. While such synthetic data is

imperfect, it helps the model disentangle the body parts that are relevant for an action and avoid learning spurious correlations. Moreover, since our motion decoder has also access to real motions, it learns to generate realistic motions, eliminating the realism problem of the synthetic composition baseline.

Our contributions are the following: (i) We establish a new benchmark on the problem of spatial compositions for 3D human motions, compare a number of baseline models on this new problem, and introduce a new evaluation metric that is based on a motion encoder that has been trained with text supervision. (ii) To address the data scarcity problem, we propose a GPT-guided synthetic data generation scheme by combining action-relevant body parts from two motions. (iii) We provide an extensive set of experiments on the BABEL dataset, including ablations that demonstrate the advantages of our synthetic training, as well as an analysis quantifying the ability of GPT-3 to assign part labels to actions.

B.2 Method

Given a set of action descriptions in the form of text, such as {"walk in a circle", "wave with the right hand"}, and a desired motion duration F, the goal is to probabilistically generate realistic 3D human motions such that all the given actions are performed simultaneously in each generated sequence. We refer to this problem as spatial composition. Note that as a proof of concept, we perform our experiments mainly with pairs of actions, but the framework is applicable beyond pairs.

In the following, we first introduce our framework to generate synthetic training data by extracting correspondence between actions and body parts from large language models (Section B.2.1). Then, we describe our model training with synthetically augmented data (Section B.2.2), and finally present implementation details (Section B.2.3).

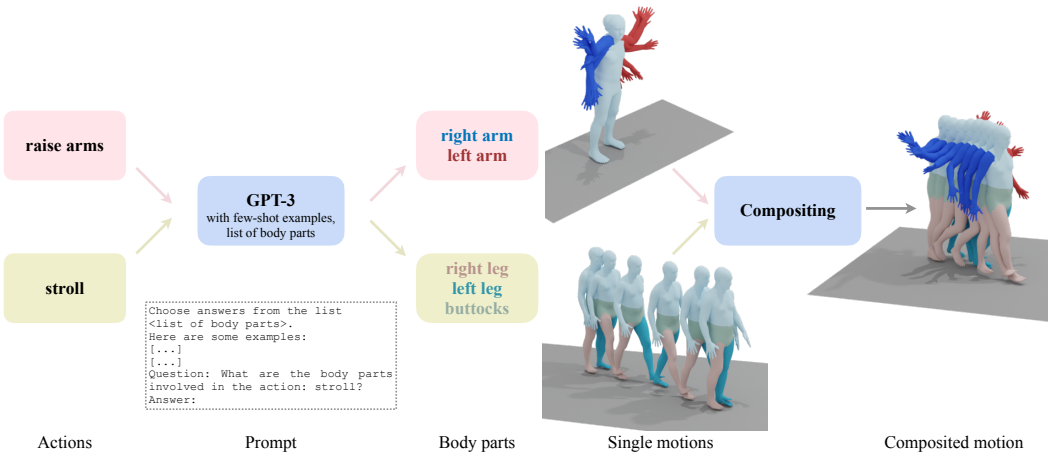


Figure B.2: **GPT-guided synthetic training data creation:** We illustrate our procedure to generate Synth-Pairs. Here, we combine two motion sequences from the training set with the corresponding labels 'stroll' and 'raise arms'. We first prompt GPT-3 with the instructions, few-shot examples containing question-answer pairs, and giving the action of interest in the last question without the answer. We minimally post-process the output of GPT-3 to assign this action to a set of body parts. The relevant body parts from each motion are then stitched together to form a new synthetically composited motion.

B.2.1 GPT-guided synthetic training data creation

As explained in Section B.1, we leverage a large language model, GPT-3 [Brown et al. 2020], to automatically assign a given action description to a set of body parts from a predefined list. Given such correspondence, we then synthetically combine existing motions together to create compositional training data. This process is illustrated in Figure B.2.

Body part label extraction from GPT-3. We process the entire set of motion descriptions in the dataset to associate each action description to a set of body parts. We use the Text-Completion tool from OpenAI's API of GPT-3 [Brown et al. 2020] to extract the body part correspondence for a given language description. Specifically, for each individual action description in the dataset, we construct a prompt consisting of three parts. (i) We specify the instruction in the form of "choose answers from the list <list of body parts>', where the list is ['left arm', 'right arm', 'left leg', 'right leg', 'torso', 'neck', 'buttocks', 'waist']". (ii) We provide few-shot examples as question-answer pairs, where the question is 'What are the body parts involved in the action:

<action>?', and the answer is the list of manually labeled body parts. (iii) The last part has the same form as the question, but we do not give the answer.

With this approach, GPT-3 outputs require minimal processing, i.e., the responses are words that correspond almost always to the provided list in (i). We post-process GPT-3's responses by removing punctuation, lowercasing, and mapping to a list of SMPL [Loper et al. 2015] body parts that we define separately, and use in the subsequent steps of our approach to generate synthetic data. We take a subset of SMPL body parts: ['left arm', 'right arm', 'left leg', 'right leg', 'torso', 'global orientation']. We coarsely define these six different body parts, but dealing with more fine-grained body parts is certainly possible.

From the first list, 'neck' is mapped to 'torso', and ['waist', buttocks'] are mapped to 'global orientation'. This is because, when prompting for free-form outputs without providing a list (i) or few-shot examples (ii), we qualitatively observe that GPT-3 refers to changes in global orientation of the body using words such as 'waist' or 'buttocks'. Hence, we replace 'global orientation' with these two words instead. GPT-3 also outputs the word 'neck' in some cases even when it is not included in the list, which motivated us to add it to our list.

To evaluate our choices for the prompt, in Table B.1 we measure the contribution of providing (i) the list, and (ii) few-shot examples in the prompt. For this, we manually label 100 action descriptions from BABEL. For each action, we annotate each body part as Yes/No/Sometimes to mark whether that body part is involved with that action. Note that we use 'Sometimes' for ambiguous cases, where it is acceptable to include, but not necessarily mandatory. For example 'hands' may or may not be involved in 'walking'. We then check the accuracy of GPT-3 body part labeling, by counting Yes/No as 1/0, ignoring optional body parts to not bias our evaluation.

A prompt asking for a free-form answer (i.e., "List the body parts involved in this action: <action>") complicates the required post-processing as one needs to handle over-detailed answers such as 'deltoids', 'triceps', or different ways of referring to the same body part. We manually built a lookup table to map from GPT-3 outputs to SMPL body parts but obtained suboptimal results. As can be seen from Table B.1, providing the list (rows a vs b) significantly boosts the

Body part labeling	Global	Torso	Left arm	Right arm	Left leg	Right leg	Mean
Part velocity magnitude	0.72	0.68	0.60	0.55	0.58	0.67	0.65
GPT-based (a) free-form	0.72	0.70	0.85	0.86	0.80	0.83	0.79
GPT-based (b) choose from list	0.79	0.68	0.89	0.90	0.88	0.89	0.84
GPT-based (c) choose from list + few-shot examples	**0.84**	**0.72**	**0.89**	**0.89**	**0.89**	**0.90**	**0.85**

Table B.1: **GPT body part labeling performance:** We report the part-labeling accuracy of GPT-3, as well as a simpler baseline based on part velocity magnitudes. For GPT-3, we experiment with various types of prompts on 100 manually annotated actions. (a) Asking which body parts are involved with an action, and post-processing free-form language outputs to associate to part labels. (b) Asking to choose from a given list of body parts, and (c) additionally also providing few-shot examples. See Section B.2.1 for more details on these prompts.

labeling accuracy, especially for picking the correct left/right arm/leg, which is further improved by providing few-shot examples (row c). We provide examples from GPT-3's responses for various prompts in the appendix of our original paper [Athanasiou et al. 2023].

Could we extract body part labels without GPT-3? To test the effectiveness of our GPT-based body part labeling, we also implement an alternative body-part labeling approach based on part velocity magnitude. The assumption is that we have action-motion pairs, and if a body part movement is above a threshold, that part should be involved with the associated action. Specifically, we compute average positional velocities across frames for each body part, standardize (subtracting the mean, dividing by the standard deviation over frames), and determine a threshold (by visual inspection) to decide if a body part is involved in a given motion. This heuristic baseline has the disadvantage that it may suffer from spurious correlations (e.g., if we only see waving while walking, we will think that leg motion is critical to waving). From the first row of Table B.1, we observe that the accuracy of this approach is significantly lower than the GPT-based approaches.

Body part composition to create new motions. Given a set of labeled motions to combine, and the extracted GPT-3 body parts involved, we first determine if the actions are compatible; i.e., whether a valid motion can be composited, based on the descriptions. For example, the actions ['walking', 'kicking with the right leg'] may not be performed at the same time as they

both include the body part 'right leg'. For the synthetic training data, we only create compositions for valid pairs that are compatible in terms of their body part involvement, and use *real* motions from the database. Next, we detail the data creation procedure.

Given two motions A and B, along with the corresponding selected body parts extracted by GPT-3, we compose these motions into a new one by performing the following steps: (1) We trim the longer motion to match the length of the shorter one; (2) We order the motions A and B such that motion B always has fewer body parts than motion A; (3) If motion B involves at least one leg or the global orientation, we also select both legs, the global orientation, and translation from motion B (otherwise, we obtain these 4 values from motion A); (4) The remaining unselected body parts (if any) are taken from motion A; (5) The composited motion is obtained by combining selected body parts from motion A and B, along with the translation according to step 3. We perform step 3 to retain plausibility as much as possible, as the leg motions are highly correlated with changes in global translation and orientation. This procedure ensures realism and accuracy of the compositions to some extent; but does not provide a guarantee.

Note that we also employ this approach as a baseline in our experiments, where we combine the motions under these assumptions using two *generated* motions from a single-action trained model. In this case, body part incompatibilities may occur ('walking' and 'kicking' both involve the leg), and body parts from motion B override the conflicting parts from motion A (see the appendix of our main paper for further details).

B.2.2 Learning to generate spatial compositions

We employ the recent architecture `TEMOS` [Petrovich et al. 2022], which encodes the text into a distribution via a Transformer encoder (text encoder $\mathcal{T}_{enc}$), and produces motions by using a Transformer decoder (motion decoder $\mathcal{M}_{dec}$). Similar to Language2Pose [Ahuja et al. 2019], `TEMOS` contains a motion encoder ($\mathcal{M}_{enc}$) and encourages a cross-modal joint space between text and motion embeddings. A simplified overview of the architecture can be seen in Figure B.3.

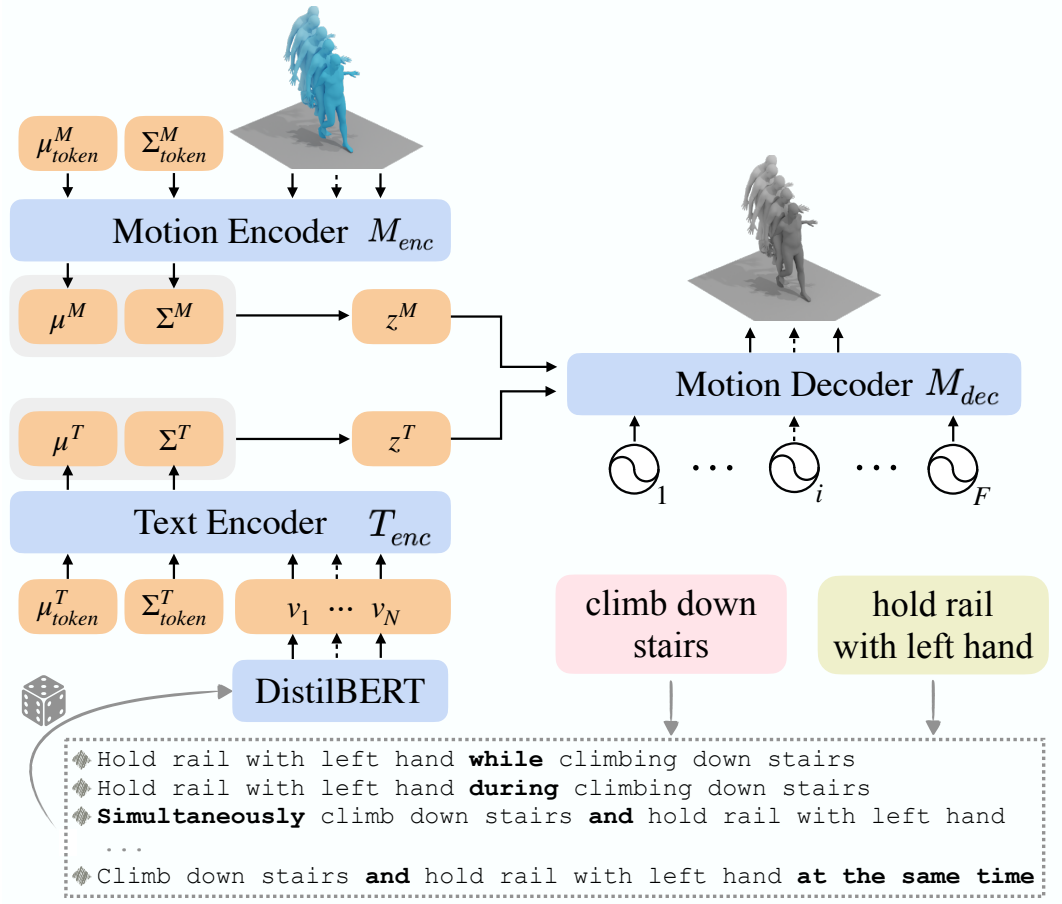


Figure B.3: **Model architecture:** We extend TEMOS [Petrovich et al. 2022] such that it is trained with compositional actions. We build multiple descriptions given two action labels, by adding words such as 'while', 'during', etc. We then randomly sample one version during training as input to the text encoder.

The motion encoder takes as input a body motion sequence $B \in \mathbb{R}^{l \times d_f}$ where d_f is the feature dimension and l the maximum motion length and outputs a single latent vector z^M and a distribution $\mathcal{N}(\mu^M, \Sigma^M)$. Similarly, the text encoder outputs z^T, which is sampled from the distribution $\mathcal{N}(\mu^T, \Sigma^T)$. These distribution parameters are obtained by appending two extra learnable tokens in the transformer encoder, and taking their corresponding outputs [Petrovich et al. 2021]. The latent vectors are sampled using the re-parametrization trick [Kingma et al. 2014]. The motion decoder then takes as input (a) the duration encoded by positional encodings $F \in \mathbb{R}^{l \times d}$, where l is the maximum motion length and d the latent dimension, (b) along with either the motion z^M or text z^T latent vector. At test time, the motion encoder is not used.

The model is supervised with the standard normal distribution losses, $\mathcal{L}_{\mathcal{KL}}^{\mathcal{T}}$=$\mathcal{KL}(\mathcal{N}(\mu^T, \Sigma^T), \mathcal{N}(0, I))$ and $\mathcal{L}_{\mathcal{KL}}^{\mathcal{M}}$=$\mathcal{KL}(\mathcal{N}(\mu^M, \Sigma^M), \mathcal{N}(0, I))$ for the text and motion distributions, respectively. Moreover, $\mathcal{L}_{\mathcal{Z}} = \tilde{\mathcal{L}}_1(z^T, z^M)$ is used to force the text latent vectors to be close to the motion latent vector, where $\tilde{\mathcal{L}}_1$ is the smooth L1 loss. Finally, the distributions of different texts and the motion are supervised via $\mathcal{L}_{\mathcal{KL}}^{\mathcal{M}\|\mathcal{T}}$=$\mathcal{KL}(\mathcal{N}(\mu^T, \Sigma^T),\ \mathcal{N}(\mu^M, \Sigma^M))$ and its symmetric version $\mathcal{L}_{\mathcal{KL}}^{\mathcal{T}\|\mathcal{M}}$. The reconstruction losses for the generated motions, $\hat{B}^M$ and $\hat{B}^T$, from both the motion and the text branches, $\mathcal{L}_{\mathcal{R}} = \tilde{\mathcal{L}}_1(B, \hat{B}^T)+\tilde{\mathcal{L}}_1(B, \hat{B}^M)$, are added to the total loss:

$$\mathcal{L} = \mathcal{L}_{\mathcal{KL}}^{\mathcal{T}} + \mathcal{L}_{\mathcal{KL}}^{\mathcal{M}} + \mathcal{L}_{\mathcal{KL}}^{\mathcal{M}\|\mathcal{T}} + \mathcal{L}_{\mathcal{KL}}^{\mathcal{T}\|\mathcal{M}} + \mathcal{L}_{\mathcal{R}} + \mathcal{L}_{\mathcal{Z}}. \tag{B.1}$$

While our experiments use `TEMOS` [Petrovich et al. 2022], our synthetic data strategy is applicable to any text-to-motion generation model. We provide further evidence on the benefits of synthetic training on a diffusion-based approach (similar to MLD [Chen et al. 2023]) in the appendix of our original paper.

Input text format and augmentations. Here, we describe how we provide the input to the text encoder. In case of a single motion that is described by one action label, we simply input the original label as in [Petrovich et al. 2022]. In case of two or more descriptions, which is the focus of this work, we combine multiple descriptions into a single text. Specifically, we use several keywords to describe simultaneous actions (e.g., 'while', 'at the same time', 'simultaneously', 'during', etc.), and randomly place them in the text description to form an input that imitates a free-form input. Moreover, we shuffle the order of the labels, and add inflections to verbs such as gerunds when grammatically applicable; e.g., when using 'while'. Figure B.3 shows some examples. Such and input formation allows users to enter free-form language descriptions at test time, which is a natural interface for humans. During training, we pick a random text augmentation, and at test time, we evaluate all the models using the conjunction word 'while'. In the appendix of our paper, we provide results with more conjunction words both seen and unseen during training.

Model	Tr. Data		TEMOS ↑	Average Positional Error ↓				Average Variance Error ↓			
	Real-P	Real-S	score	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
Single-action	✗	✓	0.601	0.592	0.551	0.286	0.712	0.076	0.075	0.013	0.083
Single-action GPT-compositing	✗	✓	0.618	0.546	0.507	0.282	0.666	0.076	0.075	0.013	0.082
SINC-STE	✓	✗	0.614	0.636	0.615	0.275	0.743	0.082	0.081	0.014	0.090
SINC	✓	✗	0.631	0.703	0.682	0.269	0.815	0.107	0.106	0.013	0.114
SINC	✓	✓	**0.640**	0.601	0.573	0.268	0.724	0.093	0.092	0.012	0.100

Table B.2: **Baseline comparison:** We train only with Real-Pairs of the BABEL dataset and report performance when compositing naively or with GPT-3 annotations. Furthermore, we ablate the model design for handling multiple textual inputs when extending TEMOS [Petrovich et al. 2022]. We observe better performance at handling action pairs with a single text encoder (SINC) that takes as input the two text labels as a single free-form description with various augmentations, as described in Section B.2.2, compared to separate text encodings of the labels (SINC-STE). Moreover, we report the performance of SINC when adding Real-Singles, as well.

B.2.3 Implementation details

We define a 3D human motion as a sequence of human poses using the SMPL body model [Loper et al. 2015]. As in TEMOS [Holden et al. 2016; Petrovich et al. 2022], we represent the motion using the 6D rotations [Zhou et al. 2019] for body joints and the 2D-projection of the x, y trajectory along with the z translation. This results in $d_f = 135$ for each body pose in each motion sequence. All the motions are canonicalized to face the same forward direction and are standardized.

The input text is encoded with DistilBERT [Sanh et al. 2019] (whose parameters are frozen), followed by a learnable linear projection. The latent dimension is fixed to $d = 256$. We use 6 layers and heads in the transformers with a linear projection of size 1024. We set the batch size to 64 and the learning rate to $3 \cdot 10^{-4}$ for all our experiments.

Our model is applicable to arbitrary numbers of actions for a given motion. Therefore, we jointly train on single actions, and multiple actions. Single actions are from real data. Multiple actions can be (i) from synthetic pairs that are randomly generated 'on the fly' or (ii) from real data where most such motions have two labels, but we also include those with more than two; see the supplementary video on our project page for more details. For each sequence in a mini-batch, if it is a real single action, with probability p, we combine it randomly with another compatible action.

Synthetic data	Training Data			TEMOS ↑	Average Positional Error ↓				Average Variance Error ↓			
	Real-P	Real-S %	Synth-P %	score	root joint	global traj.	mean local	mean global	root joint	global traj.	mean local	mean global
N/A	✓	0	0	0.631	0.703	0.682	0.269	0.815	0.107	0.106	0.013	0.114
	✓	100	0	0.640	0.601	0.573	0.268	0.724	0.093	0.092	0.012	0.100
Random composition	✗	0	100	0.539	0.489	0.434	0.291	0.595	0.075	0.074	0.012	0.082
	✗	50	50	0.540	0.587	0.535	0.288	0.687	0.077	0.076	0.012	0.083
	✓	0	100	0.619	0.485	0.438	0.272	0.602	0.074	0.073	0.011	0.081
	✓	50	50	0.617	0.454	0.394	0.272	0.560	0.069	0.068	0.011	0.075
GPT composition	✗	0	100	0.618	0.478	0.451	0.265	0.610	0.063	0.062	0.012	0.070
	✗	50	50	0.541	0.646	0.598	0.290	0.747	0.078	0.077	0.012	0.085
	✓	0	100	0.642	0.553	0.527	0.266	0.671	0.061	0.060	0.011	0.068
	✓	50	50	**0.644**	0.481	0.452	0.261	0.605	0.064	0.062	0.011	0.070

Table B.3: **Contribution of the synthetic data:** We report performance when including two types of synthetic data created by body part combination, either determined by GPT or randomly. We further experiment (i) with different percentages of sampling ratios between the Real-Singles and Synth-Pairs, and (ii) with the inclusion of Real-Pairs.

B.3 Experiments

We present data and evaluation metrics (Section B.3.1), followed by the baselines we introduce (Section B.3.2). We report quantitative experimental results with ablations (Sections B.3.3 and B.3.4). We conclude with a qualitative analysis (Section B.3.5) and a discussion of limitations (Section B.3.6).

B.3.1 Data and evaluation metrics

We use the **BABEL** dataset [Punnakkal et al. 2021], to exploit its unique potential to study simultaneous actions. Some BABEL motions come with multiple language descriptions where annotations can overlap in time. We extract all such simultaneous action pairs for both training (2851 motions), and validation sets (1232 motions). We only consider the sequences that have a length between 600 (20 sec.) and 15 (0.5 sec.) frames. From the validation set, we exclude redundant pairs with the label 'stand', because this commonly occurs in the data while not representing challenging cases. We also remove pairs that are *seen* in the training set, and end up with 667 sequences that contain two simultaneous actions. The results on the full validation set are provided in the appendix of our original paper. Besides the simultaneous pairs, we include the single-action data from BABEL in training. Specifically, there are 24066 and 8711 single-action motions for training and validation sets, respectively. In our experiments, we denote the simultaneous actions from

BABEL with **Real-Pairs**, the single-motion segments from BABEL with **Real-Singles**, and our synthetic data created by using body-part labels from GPT with **Synth-Pairs**. We perform evaluation only on the real spatial pairs of the BABEL validation set to assess the quality of simultaneous action generation. We use the validation set as test set and train all of our models for 500 epochs.

We report evaluation metrics adopted by [Ghosh et al. 2021; Athanasiou et al. 2022; Petrovich et al. 2022]: Average Positional Error (APE), and Average Variational Error (AVE). However, we observe that these metrics do not always correlate well with the visual quality of motions, nor their semantic correspondence. We introduce, and additionally report, a new `TEMOS` **score**, which compares the cosine similarity between the generated motion and the ground truth after encoding them into the motion encoder of `TEMOS` [Petrovich et al. 2022], which is trained on BABEL Real-Singles (we do not observe significant changes when altering this model with `TEMOS` trained on different data; see the appendix of our paper). This is similar in spirit to BERTScore [Zhang et al. 2020a], which evaluates text generation quality by comparing to the ground truth in the text embedding space. More details can be found in the appendix of our paper. While this metric is also imperfect (e.g., it still assumes a single ground truth action), we observe that it better correlates with realism and motion semantics as it has been trained to encode motions controlled by text descriptions. An alternative performance measure is adopted by [Guo et al. 2022a] that reports motion-to-text retrieval metrics, randomly selecting for each motion 31 negative text descriptions along with the ground truth. Finally, we include diversity metrics in the appendix of our main paper.

B.3.2 Single-action baselines

In the following, we introduce and describe two baselines using a model trained with one description per motion: (i) A naive single-action baseline that relies on a text-to-motion synthesis model trained on single actions, tested on pairs of actions. (ii) Our proposed GPT-compositing applied on independent motion generations from a single-action model.

Single-action model. Our first baseline tests the ability of single-action

models to synthesize compositions by only modifying the input text. We train with Real-Singles from BABEL. At test time, we concatenate the text descriptions using 'while' as a keyword and evaluate the generated motions.

Single-action GPT-compositing. Another single-action baseline generates two independent motions given two texts, which are then combined using our proposed GPT-guided composition, stitching body parts from two motions (as described in our synthetic data creation; see Section B.2.1). Note that unlike the synthetic data, which combines real motions, this baseline combines generated motions. The disadvantage of this model is that it requires GPT at test time, and is based on heuristics that may be error-prone, such as trimming the motions to the same duration, and resolving common body part labels (see the supplementary video on our project page for details). In the presence of a model that is trained only on individual actions (Real-Singles), we observe that the GPT-based compositing of two independent generations improves the performance over the single-action baseline (as shown in Table B.2 top). Based on qualitative observation (see Section B.3.5), the single-action baseline often generates one out of the two actions. The GPT-compositing baseline better captures both actions; however, lacks realism due to composing actions with heuristics. `SINC`, which trains on compositional data, alleviates both issues.

B.3.3 The effect of the input text format

To confirm whether our free-form input format sacrifices performance compared to a more controlled alternative of keeping the two action texts separate, we experiment with a variant of our `SINC` model by changing the text encoding. Instead of a single text combining two actions, we concatenate them together with a learnable separation token in between after independently encoding the actions with DistilBERT. We refer to this separate text encoding variant as `SINC`-STE. In Table B.2, we compare `SINC` with `SINC`-STE when trained only with Real-Pairs, and observe a better `TEMOS` score with the free-form text augmentations, at the cost of worse positional errors. We observe that metrics based on joint positions may score high even in the absence of the second action, especially if it involves a fine-grained motion (see supplementary video).

Besides quantitative performance, SINC has the advantage of allowing more flexible inputs.

B.3.4 Training with different sets of data

Contribution of Real-Singles and Real-Pairs. In Table B.2, we report the performance of SINC when adding both Real-Pairs and Real-Singles to training. We see that training with the large number of single actions of BABEL, in addition to the small amount of action pairs, improves performance, and highlights the limited scale of the available pairs.

Contribution of GPT-guided Synth-Pairs. We experiment with different training sources in Table B.3, mainly to assess the effect of adding synthetic training data. The percentages (0, 50, or 100) reflect the probability p that a real-single action is composited synthetically with another action (see Section B.2.3). When using all training data (i.e., Real-P, Real-S 50%, Synth-P 50%), we obtain the best TEMOS score, and more importantly observe better qualitative results (see Figure B.5). In particular, the model trained with GPT-guided synthetic data demonstrates superior generalization capability to unseen combinations. In the supplementary video, we provide results with input combinations that are unseen both in the real training and validation sets.

Synthetic data without GPT guidance. We further test whether our GPT-guidance to generate synthetic data is better than just randomly mixing body parts (Random composition). In Table B.3, GPT compositions outperform Random compositions, especially when training only on synthetic data (0.539 vs 0.618 TEMOS score).

B.3.5 Qualitative analysis

In Figure B.5 (a), we present simultaneous action generations using SINCfor the validation set of BABEL. We show one random generation from our model for each description pair (left), along with the ground truth (right). Note that we display one sample due to space constraints, but the model can

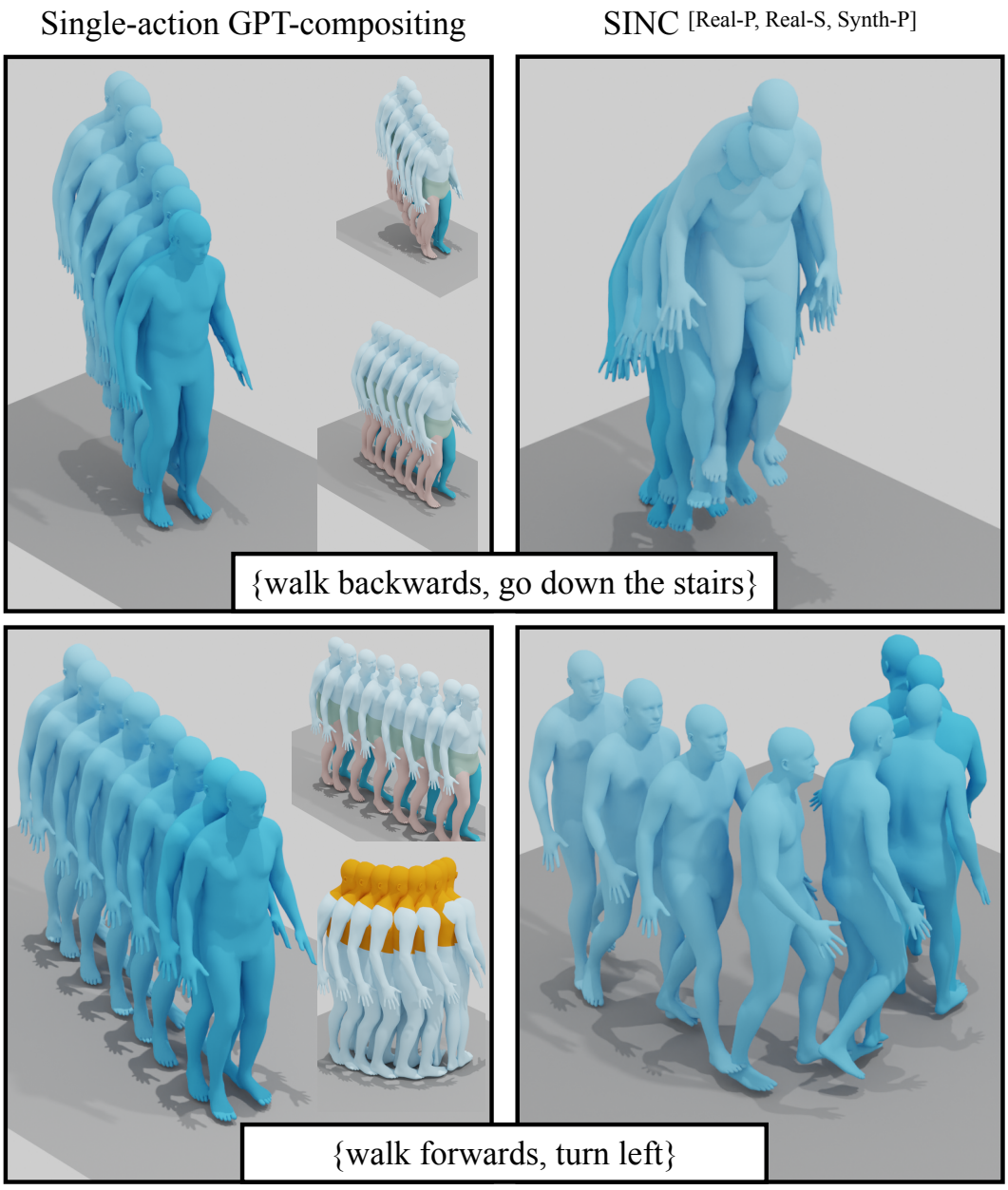


Figure B.4: **Single-action GPT-compositing vs SINC:** We show two examples that highlight the advantage of our model compared to GPT compositions. Top: The detected body parts overlap causing the stitching to generate a forwards movement. Bottom: The global orientation is taken from the 'walk forwards' failing to generate a left turn.

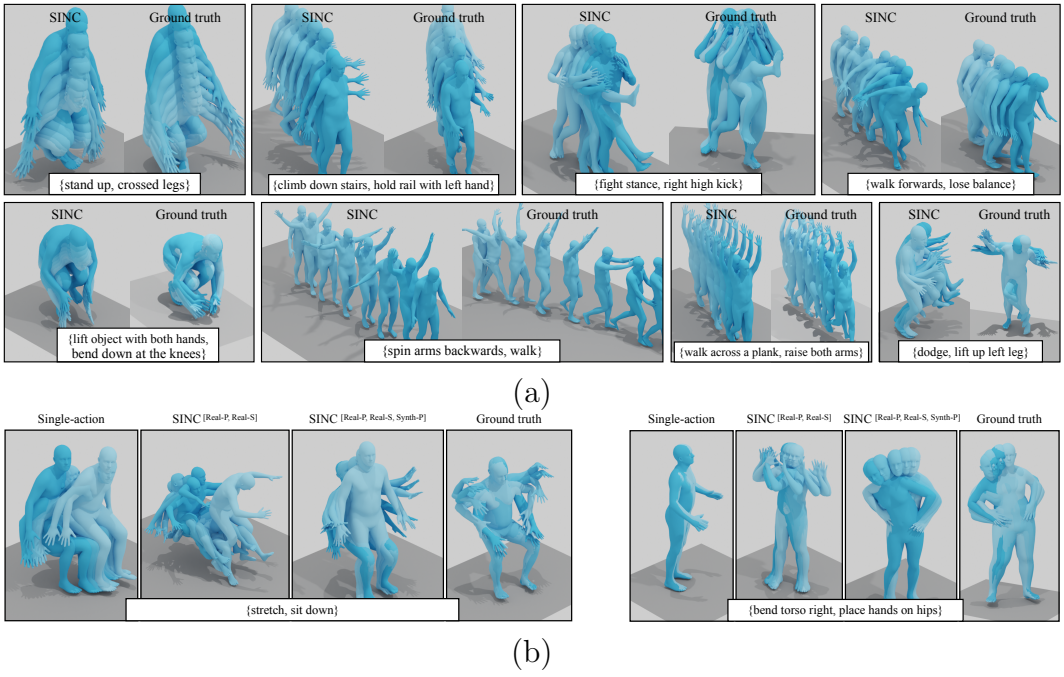


Figure B.5: **Qualitative analysis:** (a) We present qualitative results for our final model, SINC, for various description pairs from the validation set. Our generations correctly correspond to the input semantics even when they are different from the ground truth, highlighting the challenge of coordinate-based (positional) performance measures. We display the ground truth (GT) for reference to define what the given actions mean. (b) We compare different models on two simultaneous action pairs. Both the Single-action model and the model not trained on synthetic data fail to generate those two compositions. Our model trained with the synthetic data successfully generates the composition in both cases. We include more comparisons in the supplementary video on our project page.

synthesize multiple diverse motions per input. We observe that, while being sometimes different from the ground-truth motion, our generations follow the semantics of *both* actions, achieving spatial compositionality. Moreover, we qualitatively compare different models trained with and without synthetic data in Figure B.5 (b), for the pair {'stretch', 'sit down'} and {'bend torso right', 'put hands on hips'}. This action pair combination is unseen in Real-Pairs, but is seen in the Synthetic-Pairs data. In both cases, the Single-action model and the model that has not been trained on Synthetic-Pairs (first two columns) fail to generate the motion in contrast to `SINC`which is trained on spatial compositions.

Finally, in Figure B.4 we show failure cases of GPT-composition. Our baseline fails to generate a motion that corresponds to the instruction when the body parts are overlapping (top row). Another failure case happens when global orientation is important for the semantics of an action ('turn left') and is assigned to the walking action since it involves both feet (bottom row).

B.3.6 Limitations

Our framework relies on synthetic data creation by combining arbitrary motions together. Even if the body parts are compatible, in real life, not all actions appear simultaneously together. Future work should also explore the *semantic* compatibility between actions by extracting this knowledge from language models to construct semantically meaningful compositions. However, language models are also prone to mistakes. In particular, GPT-3 body part labels may be insufficient or ambiguous (e.g., 'walking' may or may not involve hands). Additionally, going beyond our 6 course parts to obtain fine-grained body part label association is important. In particular, this could involve the fingers and even facial expressions. Another limitation of our work (and the whole field) concerns the evaluation metrics. Despite introducing a new `TEMOS` score, perceptually meaningful performance measures are still missing. Finally, our model is conceptually not limited to pairs, but since it is rare to simultaneously perform more than two actions, we only focus on pairs in this work.

B.4 Conclusions

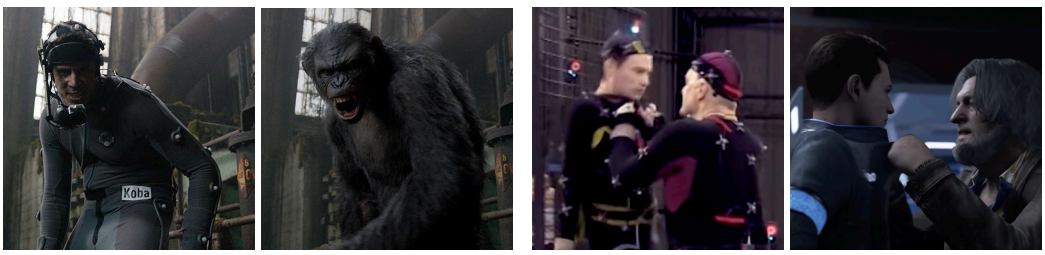
In this work, we established a new method to create spatial compositions of 3D human motions. Given a set of textual descriptions, our `SINC` model is able to generate motions that simultaneously perform multiple actions presented as textual input. We make use of the GPT-3 language model to obtain a mapping between actions and body parts to automatically create synthetic combinations of compatible actions. We use these synthetic motions to enrich the training of our model and find that it helps it generalize to new, complex, motions. We introduce multiple baselines and experiment with different data sources for this new problem. Our findings will open up possibilities for further research in fine-grained motion synthesis. While here we focus on spatial composition, future work should explore jointly modeling spatial and temporal action composition.

Résumé long en français

Contrôle en langage naturel pour la synthèse de mouvements humains en 3D

Introduction

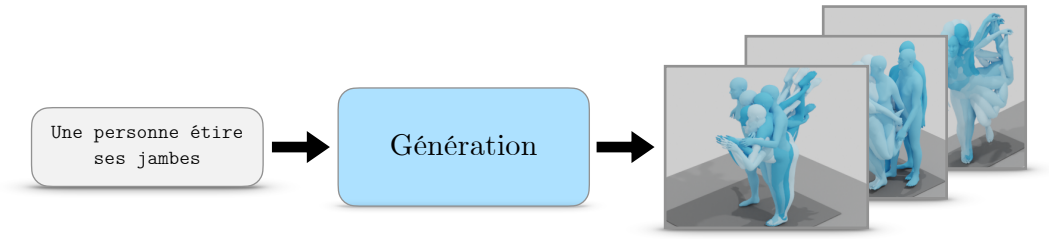
Les mouvements humains 3D jouent un rôle clé dans divers domaines, tels que le cinéma, le secteur médical, la réalité augmentée, la réalité virtuelle et l'industrie du jeu vidéo. Toutefois, ces utilisations reposent souvent sur des données de capture de mouvements coûteuses et chronophages (illustré en Figure B.6).



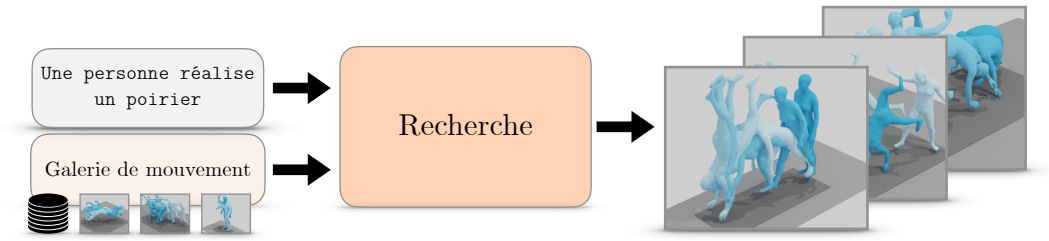
(a) Films (b) Jeux vidéos

Figure B.6: **La capture de mouvements** a transformé la création et l'interaction avec les personnages numériques, étant cruciale dans plusieurs secteurs. Elle rend possible la conversion des performances humaines en animations pour des créatures non-humaines, illustrée par la transformation d'acteurs en singes dans "La Planète des Singes" (a). Également indispensable dans les jeux vidéo pour des expériences immersives, "Detroit: Become Human" permet de contrôler des personnages qui se déplacent et expriment des émotions humaines (b).

L'objectif de cette thèse est d'explorer les modèles génératifs en tant que voie alternative pour obtenir des mouvements humains 3D. Plus spécifiquement, notre objectif est de contrôler le processus génératif par le biais d'une interface en langage naturel. Pour cela, nous développons une série de modèles qui synthétisent des mouvements réalistes et variés en suivant des entrées sémantiques. Un autre objectif de cette thèse est, à partir de textes, de réaliser une recherche dans une galerie de mouvements humains. Ces deux objectifs sont illustrés dans la Figure B.7.



(a) Génération de mouvements humains en 3D à partir de texte



(b) Recherche de mouvements humains dans une galerie à partir de texte

Figure B.7: **Objectifs :** Dans cette thèse, nous poursuivons deux objectifs principaux qui sont (a) de générer des mouvements humains à partir d'entrées sémantiques comme du texte et (b) de rechercher dans une galerie de mouvements à partir d'une requête textuelle.

S'agissant de la motivation, générer des mouvements humains de manière synthétique offre une alternative moins coûteuse par rapport à la capture de mouvements traditionnelle, permettant la création de nouveaux mouvements à la demande. Ces mouvements synthétiques peuvent également servir à entraîner des modèles d'apprentissage profond, en particulier là où l'obtention de données d'entraînements annotées est complexe. Il existe également d'autres applications, notamment dans le domaine médical ou potentiellement en robotique pour le contrôle de robots humanoïdes.

Les défis majeurs de la génération de mouvements humains sont la conception d'architectures de réseaux neuronaux capables de capturer l'aspect temporel des mouvements, la création de mouvements réalistes et physiquement possibles avec des transitions fluides ainsi que, le développement de modèles génératifs qui peuvent générer des séquences variées tout en restant fidèles aux entrées sémantiques. Aussi, la limitation de la taille des ensembles de données d'entraînement et les difficulés d'évaluation de ces modèles représentent des défis importants.

Ci-dessous, nous présentons brièvement les contributions de chaque chapitre ainsi qu'un résumé récapitulatif de ces contributions.

Génération à partir d'actions

Dans notre premier chapitre, nous relevons le défi de générer des séquences de mouvements humains conditionnées par des catégories d'actions spécifiques. Nous présentons `ACTOR`, un autoencodeur variationnel conditionnel (VAE) conçu et entraîné pour générer des mouvements humains conditionnés par des actions en utilisant la paramétrisation SMPL. Nous introduisons une représentation d'un mouvement entier dans un seul vecteur latent, ce qui constitue une nouveauté clé dans ce domaine. De cet espace latent peut être généré des mouvements humains de manière non autoregressive.

Pour surmonter le manque de données d'entraînements, nous utilisons l'estimation de mouvements humains à partir de vidéos monoculaires et nous montrons qu'il est possible d'entraîner notre générateur à partir de ces estimations bruitées. Nous présentons une étude approfondie de l'ablation de l'architecture et des fonctions de coûts et nous montrons des améliorations significatives sur plusieurs ensembles de données par rapport aux méthodes existantes grâce à notre nouvelle formulation VAE basée sur un Transformer. De plus, nous soulignons les applications pratiques de notre modèle dans deux domaines, celui de l'amélioration de la reconnaissance d'actions et celui du débruitage de données de capture de mouvements humains.

Génération à partir de textes

Dans le deuxième chapitre, nous allons au-delà des actions catégorielles et nous nous intéressons à la synthèse de divers mouvements humains 3D à partir de *descriptions textuelles*. Cela permet d'élargir le vocabulaire et d'obtenir un contrôle potentiellement plus fin. Notre travail se distingue des recherches précédentes en ne générant pas de manière déterministe une séquence de mouvement unique, mais en synthétisant des séquences multiples et variées à partir d'un texte donné. Nous proposons `TEMOS`, reposant sur notre architecture `ACTOR` basée sur un VAE, mais qui intègre cette fois un encodeur de texte pré-entraîné pour traiter les entrées en langage naturel à large vocabulaire. Nous nous assurons que l'espace latent est cohérent entre les modalités de mouvements et de textes en utilisant une fonction de coût entre les vecteurs latents des différentes modalités.

Nous fournissons une étude d'ablation complète des composants du modèle et nous montrons une performance bien supérieure à celle de l'état de l'art, tant sur les métriques qualitatives que sur les études perceptuelles. Outre notre performance compétitive, notre modèle est capable de générer plusieurs mouvements différents par description textuelle d'entrée, contrairement aux travaux déterministes précédents. Nous montrons également que notre modèle est compatible avec différentes représentations de mouvements, celles basées sur les emplacements des articulations et celles basées sur le modèle de corps paramétrique SMPL.

Recherche de mouvements à partir de textes

Dans le troisième chapitre, nous abordons la tâche adjacente de la *recherche* de mouvements humains 3D à partir de texte, où l'objectif est, par le biais d'une requête textuelle, de rechercher à l'intérieur d'une collection de mouvements. Nous présentons une approche simple et efficace, appelée `TMR`, qui s'appuie sur notre modèle précédent `TEMOS`, en intégrant une fonction de coût contrastive pour améliorer la structure de l'espace latent multimodal. Nos résultats soulignent l'importance de conserver la génération de mouvements

avec l'entraînement contrastif pour améliorer les résultats.

De plus, étant donné la similitude des descriptions textuelles à travers les mouvements dans l'ensemble de données, nous montrons qu'une simple stratégie de filtrage peut améliorer la performance du modèle. De surcroît, la tâche de recherche n'étant pas étudiée de manière exhaustive, nous introduisons une série de benchmarks de difficultés variées et nous effectuons des analyses sur plusieurs protocoles. Enfin, nous fournissons des expériences étendues pour analyser les effets de chaque composant dans des environnements contrôlés et nous montrons que nos résultats surpassent l'état de l'art. Ainsi, ce modèle de recherche haute performance peut être utilisé comme évaluateur des méthodes de génération de mouvements humains 3D.

Génération à partir de chronologies multi-pistes

Dans le quatrième chapitre, nous exposons un nouveau problème appelé "contrôle par chronologie multi-pistes" pour la synthèse de mouvements humains 3D pilotée par du texte. En d'autres termes, au lieu d'une seule description textuelle, les utilisateurs organisent plusieurs textes dans des intervalles temporels qui peuvent se chevaucher. Nous présentons `STMC`, une méthode de débruitage en temps de test pouvant être intégrée à n'importe quel modèle de diffusion de mouvements humains pré-entraîné. Cette méthode est capable de gérer les compositions spatiales et temporelles présentes dans les chronologies en entrée.

Nos évaluations démontrent que notre méthode génère des mouvements qui correspondent étroitement aux aspects sémantiques et temporels de la chronologie d'entrée et fonctionne mieux que des méthodes de références soigneusement élaborées. De plus, nous entraînons un modèle basé sur la diffusion pour générer directement les paramètres de pose SMPL. Cela évite que nous recourions à une optimisation au moment du test, étape obligatoire pour les méthodes traditionnelles.

Résumé

En résumé, les contributions de cette thèse sont les suivantes (i) nous développons un autoencodeur variationnel génératif, ACTOR, pour la génération de séquences de mouvements humains conditionnée par l'action, (ii) nous présentons TEMOS, un modèle génératif conditionné par le texte qui synthétise des mouvements humains diversifiés, (iii) nous exposons TMR, une nouvelle approche pour la recherche de mouvements humains 3D à partir de texte, (iv) enfin, nous proposons STMC, une méthode pour la génération de mouvements humains controlés par une chronologie à plusieurs pistes.

Bibliography

Aberman, Kfir et al. (2020). "Skeleton-Aware Networks for Deep Motion Retargeting." In: *ACM Transactions on Graphics (TOG)* (page 87).

Advanced Computing Center for the Arts and Design (n.d.). *ACCAD MoCap Dataset*. URL: https://accad.osu.edu/research/motion-lab/mocap-system-and-data (page 30).

Ahn, Hyemin et al. (2018). "Text2Action: Generative Adversarial Synthesis from Language to Action." In: *International Conference on Robotics and Automation (ICRA)* (pages 18, 19, 23).

Ahuja, Chaitanya and Louis-Philippe Morency (2019). "Language2Pose: Natural Language Grounded Pose Forecasting." In: *International Conference on 3D Vision (3DV)* (pages 10, 18, 19, 58, 59, 61, 63, 66–71, 73, 88, 162, 168).

Aïder, Méziane, Oussama Gacem, and Mhand Hifi (2022). "Branch and solve strategies-based algorithm for the quadratic multiple knapsack problem." In: *Journal of the Operational Research Society* (page 94).

Akhter, Ijaz and Michael J. Black (2015). "Pose-Conditioned Joint Angle Limits for 3D Human Pose Reconstruction." In: *Computer Vision and Pattern Recognition (CVPR)* (page 30).

Aksan, Emre, Manuel Kaufmann, and Otmar Hilliges (2019). "Structured Prediction Helps 3D Human Motion Modelling." In: *International Conference on Computer Vision (ICCV)* (pages 22, 35).

Alexanderson, Simon et al. (2023). "Listen, Denoise, Action! Audio-Driven Motion Synthesis with Diffusion Models." In: *ACM Transactions on Graphics (TOG)* (page 23).

Anguelov, Dragomir et al. (2005). "SCAPE: shape completion and animation of people." In: *SIGGRAPH* (pages 28, 29).

Arikan, Okan, David A. Forsyth, and James F. O'Brien (2003). "Motion Synthesis from Annotations." In: *ACM Transactions on Graphics (TOG)* (page 84).

Aristidou, Andreas, Ariel Shamir, and Yiorgos Chrysanthou (2019). "Digital Dance Ethnography: Organizing Large Dance Collections." In: *Computing and Cultural Heritage* (page 30).

Athanasiou, Nikos et al. (2022). "TEACH: Temporal Action Composition for 3D Humans." In: *International Conference on 3D Vision (3DV)* (pages 9, 20, 113, 114, 125, 126, 128, 129, 161, 162, 173).

Athanasiou, Nikos et al. (2023). "SINC: Spatial Composition of 3D Human Motions for Simultaneous Action Generation." In: *International Conference on Computer Vision (ICCV)* (pages 9, 16, 109, 113–115, 118–120, 128, 129, 167).

Badler, Norman I. (1975). "Temporal Scene Analysis: Conceptual Descriptions of Object Movements." PhD thesis. University of Toronto (page 35).

Badler, Norman I., Cary B. Phillips, and Bonnie Lynn Webber (1993). *Simulating Humans: Computer Graphics Animation and Control*. Oxford University Press (pages 3, 16, 35).

Bain, Max et al. (2021). "Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval." In: *International Conference on Computer Vision (ICCV)* (pages 25, 58, 90).

Bard, Google (2023). *Bard: Conversational AI developed by Google*. URL: https://bard.google.com/chat (page 13).

Barsoum, Emad, John Kender, and Zicheng Liu (2018). "HP-GAN: Probabilistic 3D Human Motion Prediction via GAN." In: *Computer Vision and Pattern Recognition Workshops (CVPRW)* (pages 21, 35).

Bhattacharya, Uttaran et al. (2021). "Speech2AffectiveGestures: Synthesizing Co-Speech Gestures with Generative Adversarial Affective Expression Learning." In: *ACM International Conference on Multimedia (ACMMM)* (page 23).

Black, Michael J. et al. (2023). "BEDLAM: A Synthetic Dataset of Bodies Exhibiting Detailed Lifelike Animated Motion." In: *Computer Vision and Pattern Recognition (CVPR)* (page 4).

Bogo, Federica et al. (2016). "Keep it SMPL: Automatic Estimation of 3D Human Pose and Shape from a Single Image." In: *European Conference on Computer Vision (ECCV)* (pages 28, 122).

Bogo, Federica et al. (2017). "Dynamic FAUST: Registering Human Bodies in Motion." In: *Computer Vision and Pattern Recognition (CVPR)* (page 30).

Bowden, Richard (2000). "Learning Statistical Models of Human Motion." In: *Computer Vision and Pattern Recognition (CVPR), Workshop on Human Modeling, Analysis and Synthesis* (page 21).

Brooks, Tim, Aleksander Holynski, and Alexei A Efros (2023). "InstructPix2Pix: Learning to Follow Image Editing Instructions." In: *Computer Vision and Pattern Recognition (CVPR)* (page 163).

Brown, Tom et al. (2020). "Language Models are Few-Shot Learners." In: *Neural Information Processing Systems (NeurIPS)* (pages 21, 42, 119, 162, 165).

Büttner, Michael and Simon Clavet (2015). "Motion Matching - The Road to Next Gen Animation." In: *Nucl.ai,* youtu.be/watch?v=z_wpgHFSWss&t=658s (pages 24, 84).

Cai, Haoye et al. (2018). "Deep Video Generation, Prediction and Completion of Human Action Sequences." In: *European Conference on Computer Vision (ECCV)* (pages 49, 50).

Cai, Zhongang et al. (2021). "Playing for 3D human recovery." In: *arXiv:2110.07588* (page 4).

CMU Graphics Lab Motion Capture Database (2003). Carnegie-Mellon University. URL: http://mocap.cs.cmu.edu (pages 19, 30, 31, 37, 66).

Cascante-Bonilla, Paola et al. (2023). "Going Beyond Nouns With Vision & Language Models Using Synthetic Data." In: *International Conference on Computer Vision (ICCV)* (page 5).

Cervantes, Pablo et al. (2022). "Implicit Neural Representations for Variable Length Human Motion Generation." In: *European Conference on Computer Vision (ECCV)* (page 18).

Chatzitofis, Anargyros et al. (2020). "HUMAN4D: A Human-Centric Multimodal Dataset for Motions and Immersive Media." In: *IEEE Access* (page 30).

Chen, Ting (2023). "On the Importance of Noise Scheduling for Diffusion Models." In: *arXiv:2301.10972* (page 122).

Chen, Xin et al. (2023). "Executing your Commands via Motion Diffusion in Latent Space." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 16, 19, 25, 84, 114, 118, 162, 170).

Cheng, Xingyi et al. (2019). "Variational Semi-Supervised Aspect-Term Sentiment Analysis via Transformer." In: *Computational Natural Language Learning (CoNLL)* (page 36).

Corona, Enric et al. (2020). "Context-Aware Human Motion Prediction." In: *Computer Vision and Pattern Recognition (CVPR)* (page 22).

Cudeiro, Daniel et al. (2019). "Capture, Learning, and Synthesis of 3D Speaking Styles." In: *Computer Vision and Pattern Recognition (CVPR)* (page 23).

Dabral, Rishabh et al. (2023). "MoFusion: A Framework for Denoising-Diffusion-based Motion Synthesis." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 19, 85).

Darcet, Timothée et al. (2024). "Vision Transformers Need Registers." In: *International Conference on Learning Representations (ICLR)* (page 123).

Devlin, Jacob et al. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *North American Chapter of the Association for Computational Linguistics (NAACL)* (pages 18, 36, 40, 42, 58, 76, 77, 148).

Dhariwal, Prafulla and Alexander Nichol (2021). "Diffusion Models Beat GANs on Image Synthesis." In: *Neural Information Processing Systems (NeurIPS)* (page 15).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio (2017). "Density estimation using Real NVP." In: *International Conference on Learning Representations (ICLR)* (page 16).

Dosovitskiy, Alexey et al. (2021). "An image is worth 16x16 words: Transformers for image recognition at scale." In: *International Conference on Learning Representations (ICLR)* (pages 36, 40).

Doveh, Sivan et al. (2023). "Teaching Structured Vision & Language Concepts to Vision & Language Models." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 85, 96).

Duan, Yinglin et al. (2021). "Single-Shot Motion Completion with Transformer." In: *arXiv:2103.00776* (page 22).

Duan, Yinglin et al. (2022). "A Unified Framework for Real Time Motion Completion." In: *AAAI Conference on Artificial Intelligence* (page 22).

Escorcia, Victor et al. (2019). "Temporal localization of moments in video collections with natural language." In: *arXiv:1907.12763* (pages 86, 100).

Fan, Yingruo et al. (2022). "FaceFormer: Speech-Driven 3D Facial Animation With Transformers." In: *Computer Vision and Pattern Recognition (CVPR)* (page 23).

Fang, Le et al. (2021). "Transformer-based Conditional Variational Autoencoder for Controllable Story Generation." In: *arXiv:2101.00828* (page 36).

Fragkiadaki, Katerina et al. (2015). "Recurrent Network Models for Human Dynamics." In: *International Conference on Computer Vision (ICCV)* (pages 21, 36).

Futrelle, Robert P. and Glen C. Speckert (1978). "Extraction of motion data by interactive processing." In: *Conference Pattern Recognition and Image Processing (CP)* (page 16).

Galata, Aphrodite, Neil Johnson, and David Hogg (2001). "Learning Variable Length Markov Models of Behaviour." In: *Computer Vision and Image Understanding (CVIU)* (page 21).

Gao, Jiyang et al. (2017). "Tall: Temporal activity localization via language query." In: *International Conference on Computer Vision (ICCV)* (pages 86, 100).

Gao, Tong et al. (2015). "DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization." In: *ACM Symposium on User Interface Software & Technology* (page 58).

Gavrila, Dariu M. (1999). "The Visual Analysis of Human Movement: A Survey." In: *Computer Vision and Image Understanding (CVIU)* (page 16).

Ghorbani, Saeed et al. (2021). "MoVi: A large multi-purpose human motion and video dataset." In: *Plos One* (page 30).

Ghosh, Anindita et al. (2021). "Synthesis of Compositional Animations From Textual Descriptions." In: *International Conference on Computer Vision (ICCV)* (pages 18, 19, 58, 59, 61, 65–71, 144, 153, 162, 173).

Ginosar, S. et al. (2019). "Learning Individual Styles of Conversational Gesture." In: *Computer Vision and Pattern Recognition (CVPR)* (page 23).

Goodfellow, Ian et al. (2014). "Generative Adversarial Nets." In: *Neural Information Processing Systems (NeurIPS)* (pages 15, 21).

Gopalakrishnan, Anand et al. (2019). "A Neural Temporal Model for Human Motion Prediction." In: *Computer Vision and Pattern Recognition (CVPR)* (page 21).

Gu, Tianpei et al. (2022). "Stochastic Trajectory Prediction via Motion Indeterminacy Diffusion." In: *Computer Vision and Pattern Recognition (CVPR)* (page 21).

Guo, Chuan et al. (2020). "Action2Motion: Conditioned Generation of 3D Human Motions." In: *ACM International Conference on Multimedia (ACMMM)* (pages 17, 31, 32, 37, 42–44, 49, 50, 53, 58, 91, 123, 144).

Guo, Chuan et al. (2022a). "Generating Diverse and Natural 3D Human Motions From Text." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 6,

19, 24, 32, 84, 85, 87, 89, 91–96, 99, 100, 114, 116, 122, 123, 126, 161–163, 173).

Guo, Chuan et al. (2022b). "TM2T: Stochastic and Tokenized Modeling for the Reciprocal Generation of 3D Human Motions and Texts." In: *European Conference on Computer Vision (ECCV)* (pages 16, 19, 136).

Habibie, Ikhsanul et al. (2017). "A Recurrent Variational Autoencoder for Human Motion Synthesis." In: *British Machine Vision Conference (BMVC)* (pages 17, 21, 23, 35).

Habibie, Ikhsanul et al. (2022). "A Motion Matching-based Framework for Controllable Gesture Synthesis from Speech." In: *SIGGRAPH* (page 23).

Hadsell, R., S. Chopra, and Y. LeCun (2006). "Dimensionality Reduction by Learning an Invariant Mapping." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 25, 94, 95).

Harvey, Félix G. and Cristopher Joseph Pal (2018). "Recurrent transition networks for character locomotion." In: *SIGGRAPH* (page 22).

Harvey, Félix G. et al. (2020). "Robust motion in-betweening." In: *ACM Transactions on Graphics (TOG)* (page 22).

Hassan, Mohamed et al. (2021). "Stochastic scene-aware motion prediction." In: *International Conference on Computer Vision (ICCV)* (page 23).

He, Chengan et al. (2022). "NeMF: Neural Motion Fields for Kinematic Animation." In: *Neural Information Processing Systems (NeurIPS)* (page 25).

Hendricks, Lisa Anne et al. (2017). "Localizing moments in video with natural language." In: *International Conference on Computer Vision (ICCV)* (pages 86, 100).

Hendrycks, Dan and Kevin Gimpel (2016). "Gaussian error linear units (GELUs)." In: *arXiv:1606.08415* (page 42).

Henter, Gustav Eje, Simon Alexanderson, and Jonas Beskow (2020). "MoGlow: Probabilistic and Controllable Motion Synthesis Using Normalising Flows." In: *ACM Transactions on Graphics (TOG)* (page 17).

Hesse, Nikolas et al. (2018). "Learning an Infant Body Model from RGB-D Data for Accurate Full Body Motion Analysis." In: *Medical Image Computing and Computer Assisted Intervention (MICCAI)* (page 29).

Hessel, Jack et al. (2021). "CLIPScore: A Reference-free Evaluation Metric for Image Captioning." In: *Empirical Methods in Natural Language Processing (EMNLP)* (page 125).

Higgins, Irina et al. (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework." In: *International Conference on Learning Representations (ICLR)* (page 41).

Hill, I. (1983). "Natural language versus computer language." In: *Designing for Human-Computer Communication* (page 57).

Ho, Jonathan, Ajay Jain, and Pieter Abbeel (2020). "Denoising diffusion probabilistic models." In: *Neural Information Processing Systems (NeurIPS)* (pages 15, 19, 21, 116, 117).

Ho, Jonathan and Tim Salimans (2021). "Classifier-free diffusion guidance." In: *Deep Generative Models and Downstream Applications NeurIPS Workshop* (pages 15, 117).

Hogg, David (1983). "Model-based vision: a program to see a walking person." In: *Image and Vision Computing* (page 28).

Holden, Daniel, Taku Komura, and Jun Saito (2017). "Phase-Functioned Neural Networks for Character Control." In: *ACM Transactions on Graphics (TOG)* (pages 17, 23, 35, 93).

Holden, Daniel, Jun Saito, and Taku Komura (2016). "A Deep Learning Framework for Character Motion Synthesis and Editing." In: *ACM Transactions on Graphics (TOG)* (pages 27, 61, 112, 122, 171).

Holden, Daniel et al. (2020). "Learned motion matching." In: *ACM Transactions on Graphics (TOG)* (pages 23, 24).

Hoyet, Ludovic et al. (2012). "Sleight of Hand: Perception of Finger Motion from Reduced Marker Sets." In: *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (page 30).

Huang, Wencan et al. (2021). "Towards Fast and High-Quality Sign Language Production." In: *ACM International Conference on Multimedia (ACMMM)* (page 19).

Ionescu, Catalin, Fuxin Li, and Cristian Sminchisescu (2011). "Latent structured models for human pose estimation." In: *International Conference on Computer Vision (ICCV)* (pages 19, 30).

Ionescu, Catalin et al. (2014). "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments." In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (pages 19, 30, 37).

Ji, Yanli et al. (2018). "A Large-Scale RGB-D Database for Arbitrary-View Human Action Recognition." In: *ACM International Conference on Multimedia (ACMMM)* (pages 31, 37, 43, 52).

Jiang, Junyan et al. (2020). "Transformer VAE: A Hierarchical Model for Structure-Aware and Interpretable Music Representation Learning." In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (page 36).

Jin, Peng et al. (2023). "Act As You Wish: Fine-Grained Control of Motion Diffusion Model with Hierarchical Semantic Graphs." In: *Neural Information Processing Systems (NeurIPS)* (page 19).

Johnson, Justin et al. (2020). "Accelerating 3D Deep Learning with PyTorch3D." In: *SIGGRAPH Asia 2020 Courses* (page 42).

Kalakonda, Sai Shashank, Shubh Maheshwari, and Ravi Kiran Sarvadevabhatla (2023). "Action-GPT: Leveraging Large-scale Language Models for Improved and Generalized Action Generation." In: *International Conference on Multimedia and Expo (ICME)* (page 20).

Kanazawa, Angjoo et al. (2018). "End-to-end Recovery of Human Shape and Pose." In: *Computer Vision and Pattern Recognition (CVPR)* (page 31).

Kanazawa, Angjoo et al. (2019). "Learning 3D Human Dynamics From Video." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 30, 31).

Kania, Kacper, Marek Kowalski, and Tomasz Trzciński (2021). "TrajeVAE: Controllable Human Motion Generation from Trajectories." In: *arXiv:2104.00351* (page 23).

Karras, Tero et al. (2017). "Audio-driven facial animation by joint end-to-end learning of pose and emotion." In: *ACM Transactions on Graphics (TOG)* (page 23).

Karunratanakul, Korrawe et al. (2023). "GMD: Controllable Human Motion Synthesis via Guided Diffusion Models." In: *International Conference on Computer Vision (ICCV)* (page 23).

Kaufmann, Manuel et al. (2020). "Convolutional Autoencoders for Human Motion Infilling." In: *International Conference on 3D Vision (3DV)* (page 22).

Keller, Marilyn et al. (2022). "OSSO: Obtaining Skeletal Shape from Outside." In: *Computer Vision and Pattern Recognition (CVPR)* (page 29).

Keller, Marilyn et al. (2023). "From Skin to Skeleton: Towards Biomechanically Accurate 3D Digital Humans." In: *ACM Transactions on Graphics (TOG)* (page 29).

Kim, Jihoon, Jiseob Kim, and Sungjoon Choi (2023). "FLAME: Free-form Language-based Motion Synthesis & Editing." In: *AAAI Conference on Artificial Intelligence* (page 19).

Kim, Jihoon et al. (2022). "Conditional Motion In-betweening." In: *Pattern Recognition* (page 22).

Kingma, Diederik P and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization." In: *International Conference on Learning Representations (ICLR)* (page 64).

Kingma, Diederik P and Max Welling (2014). "Auto-encoding variational bayes." In: *International Conference on Learning Representations (ICLR)* (pages 14, 21, 39, 61, 62, 169).

Kocabas, Muhammed, Nikos Athanasiou, and Michael J. Black (2020). "VIBE: Video Inference for Human Body Pose and Shape Estimation." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 30, 31, 37, 42, 51).

Kocabas, Muhammed et al. (2021). "PARE: Part Attention Regressor for 3D Human Body Estimation." In: *International Conference on Computer Vision (ICCV)* (page 30).

Kocabas, Muhammed et al. (2024). "PACE: Human and Motion Estimation from in-the-wild Videos." In: *International Conference on 3D Vision (3DV)* (page 30).

Kolotouros, Nikos et al. (2019). "Learning to Reconstruct 3D Human Pose and Shape via Model-fitting in the Loop." In: *International Conference on Computer Vision (ICCV)* (page 31).

Kovar, Lucas, Michael Gleicher, and Frédéric H. Pighin (2002). "Motion graphs." In: *ACM Transactions on Graphics (TOG)* (page 84).

Kulkarni, Nilesh et al. (2023). "NIFTY: Neural Object Interaction Fields for Guided Human Motion Synthesis." In: *arXiv:2307.07511* (pages 16, 24).

Lab, Bio Motion (n.d.). *BMLhandball Motion Capture Database*. URL: https://www.biomotionlab.ca// (page 30).

Lee, Hsin-Ying et al. (2019). "Dancing to Music." In: *Neural Information Processing Systems (NeurIPS)* (page 23).

Lee, Kyungho, Seyoung Lee, and Jehee Lee (2018). "Interactive character animation by learning multi-objective control." In: *ACM Transactions on Graphics (TOG)* (pages 23, 36).

Lee, Taeryung, Gyeongsik Moon, and Kyoung Mu Lee (2023). "MultiAct: Long-Term 3D Human Motion Generation from Multiple Action Labels." In: *AAAI Conference on Artificial Intelligence* (page 20).

Lei, Jie et al. (2020). "TVR: A large-scale dataset for video-subtitle moment retrieval." In: *European Conference on Computer Vision (ECCV)* (pages 86, 100).

Li, Jiaman et al. (2020). "Learning to Generate Diverse Dance Motions with Transformer." In: *arXiv:2008.08171* (page 23).

Li, Junnan et al. (2022a). "BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation." In: *International Conference on Machine Learning (ICML)* (pages 25, 85, 90).

Li, Peizhuo et al. (2022b). "GANimator: Neural Motion Synthesis from a Single Sequence." In: *ACM Transactions on Graphics (TOG)* (pages 16, 17).

Li, Ruilong et al. (2021). "AI Choreographer: Music Conditioned 3D Dance Generation with AIST++." In: *International Conference on Computer Vision (ICCV)* (pages 22, 23).

Lin, Angela S et al. (2018a). "Generating Animated Videos of Human Activities from Natural Language Descriptions." In: *ViGIL NeurIPS Workshop* (pages 18, 19, 58, 67–71, 162).

Lin, Jing et al. (2023). "Motion-X: A Large-scale 3D Expressive Whole-body Human Motion Dataset." In: *Neural Information Processing Systems (NeurIPS)* (page 32).

Lin, Tsung-Yi et al. (2014). "Microsoft COCO: Common Objects in Context." In: *European Conference on Computer Vision (ECCV)* (page 26).

Lin, X. and M. Amer (2018b). "Human Motion Modeling using DVGANs." In: *arXiv:1804.10652* (page 19).

Lin, Zhaojiang et al. (2020). "Variational Transformers for Diverse Response Generation." In: *arXiv:2003.12738* (page 36).

Ling, Hung Yu et al. (2020). "Character Controllers Using Motion VAEs." In: *ACM Transactions on Graphics (TOG)* (page 23).

Liu, Jun et al. (2019a). "NTU RGB+D 120: A Large-Scale Benchmark for 3D Human Activity Understanding." In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (pages 31, 42).

Liu, Yinhan et al. (2019b). "RoBERTa: A Robustly Optimized BERT Pretraining Approach." In: *arXiv:1907.11692* (page 77).

Liu, Yuejiang et al. (2022). "Towards Robust and Adaptive Motion Forecasting: A Causal Representation Perspective." In: *Computer Vision and Pattern Recognition (CVPR)* (page 21).

Loper, Matthew, Naureen Mahmood, and Michael J. Black (2014). "MoSh: Motion and Shape Capture from Sparse Markers." In: *ACM Transactions on Graphics (TOG)* (page 30).

Loper, Matthew et al. (2015). "SMPL: A Skinned Multi-Person Linear Model." In: *ACM Transactions on Graphics (TOG)* (pages 1, 22, 26, 28–30, 36, 38, 60, 61, 87, 93, 115, 116, 122, 147, 166, 171).

Loshchilov, Ilya and Frank Hutter (2019). "Decoupled Weight Decay Regularization." In: *International Conference on Learning Representations (ICLR)* (pages 64, 91, 151).

Ltd., Eyes JAPAN Co. (n.d.). *Eyes Japan MoCap Dataset*. URL: http://mocapdata.com (page 30).

Lucas, Thomas et al. (2022). "PoseGPT: Quantizing human motion for large scale generative modeling." In: *European Conference on Computer Vision (ECCV)* (page 17).

Luo, Calvin (2022). "Understanding Diffusion Models: A Unified Perspective." In: *arXiv:2208.11970* (page 117).

Luo, Huaishao et al. (2022). "CLIP4Clip: An empirical study of CLIP for end to end video clip retrieval and captioning." In: *Neurocomputing* (page 25).

Luo, Zhengyi, S. Alireza Golestaneh, and Kris M. Kitani (2020). "3D Human Motion Estimation via Motion Compression and Refinement." In: *Asian Conference on Computer Vision (ACCV)* (page 30).

Mahmood, Naureen et al. (2019). "AMASS: Archive of Motion Capture as Surface Shapes." In: *International Conference on Computer Vision (ICCV)* (pages 30–32, 37, 66, 91, 123, 145, 149, 152, 162).

Mandery, Christian et al. (2015). "The KIT whole-body human motion database." In: *International Conference on Advanced Robotics (ICAR)* (pages 30, 31, 66).

Mao, Wei, Miaomiao Liu, and Mathieu Salzmann (2022). "Weakly-supervised Action Transition Learning for Stochastic Human Motion Prediction." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 20, 149).

Marr, D., H. K. Nishihara, and Sydney Brenner (1978). "Representation and recognition of the spatial organization of three-dimensional shapes." In: *Royal Society of London. Series B. Biological Sciences* (page 28).

Martinez, Julieta, Michael J. Black, and Javier Romero (2017). "On Human Motion Prediction Using Recurrent Neural Networks." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 21, 36, 58).

Mason, Ian, Sebastian Starke, and Taku Komura (2022). "Real-Time Style Modelling of Human Locomotion via Feature-Wise Transformations and Local Motion Phases." In: *ACM Computer Graphics and Interactive Techniques* (page 30).

Miech, Antoine et al. (2020). "End-to-End Learning of Visual Representations from Uncurated Instructional Videos." In: *Computer Vision and Pattern Recognition (CVPR)* (page 25).

Miech, Antoine et al. (2021). "Thinking Fast and Slow: Efficient Text-to-Visual Retrieval with Transformers." In: *Computer Vision and Pattern Recognition (CVPR)* (page 88).

Mikolov, Tomas et al. (2013). "Distributed Representations of Words and Phrases and their Compositionality." In: *Neural Information Processing Systems (NeurIPS)* (page 18).

Mildenhall, Ben et al. (2020). "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis." In: *European Conference on Computer Vision (ECCV)* (page 36).

Moltisanti, Davide et al. (2022). "BRACE: The Breakdancing Competition Dataset for Dance Motion Synthesis." In: *European Conference on Computer Vision (ECCV)* (page 23).

Müller, M. et al. (2007). *Documentation Mocap Database HDM05*. Tech. rep. Universität Bonn (page 30).

O'Rourke, Joseph and Norman I. Badler (1980). "Model-based image analysis of human motion using constraint propagation." In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (pages 16, 28).

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals (2018). "Representation learning with contrastive predictive coding." In: *arXiv:1807.03748* (pages 25, 85, 90, 95, 97).

OpenAI (2022). *ChatGPT: Conversational AI in OpenAI's GPT*. URL: https://openai.com/blog/chatgpt (page 13).

Ormoneit, Dirk et al. (2005). "Representing cyclic human motion using functional analysis." In: *Image and Vision Computing* (page 16).

Osman, Ahmed A A et al. (2022). "SUPR: A Sparse Unified Part-Based Human Body Model." In: *European Conference on Computer Vision (ECCV)* (pages 28, 29).

Osman, Ahmed A. A., Timo Bolkart, and Michael J. Black (2020). "STAR: Sparse Trained Articulated Human Body Regressor." In: *European Conference on Computer Vision (ECCV)* (pages 28, 29, 38).

Paszke, Adam et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In: *Neural Information Processing Systems (NeurIPS)* (page 42).

Pavlakos, Georgios et al. (2019). "Expressive Body Capture: 3D Hands, Face, and Body From a Single Image." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 28, 29, 32, 38, 42).

Pavllo, Dario, David Grangier, and Michael Auli (2018). "QuaterNet: A Quaternion-based Recurrent Model for Human Motion." In: *British Machine Vision Conference (BMVC)* (pages 21, 23).

Pavllo, Dario et al. (2019). "Modeling Human Motion with Quaternion-Based Neural Networks." In: *International Journal of Computer Vision (IJCV)* (page 21).

Petrovich, Mathis, Michael J. Black, and Gül Varol (2021). "Action-Conditioned 3D Human Motion Synthesis with Transformer VAE." In: *International Conference on Computer Vision (ICCV)* (pages 9, 13, 16, 17, 60, 112, 144, 145, 148, 150, 169).

Petrovich, Mathis, Michael J. Black, and Gül Varol (2022). "TEMOS: Generating diverse human motions from textual descriptions." In: *European Conference on Computer Vision (ECCV)* (pages 9, 19, 67, 113, 114, 125, 144, 145, 147–150, 153, 161–163, 168–171, 173).

Petrovich, Mathis, Michael J. Black, and Gül Varol (2023). "TMR: Text-to-Motion Retrieval Using Contrastive 3D Human Motion Synthesis." In:

International Conference on Computer Vision (ICCV) (pages 9, 25, 96, 124).

Petrovich, Mathis et al. (2024). "Multi-Track Timeline Control for Text-Driven 3D Human Motion Generation." In: *CVPR Workshop on Human Motion Generation* (pages 9, 124).

Plappert, Matthias, Christian Mandery, and Tamim Asfour (2016). "The KIT Motion-Language Dataset." In: *Big Data* (pages 31, 32, 65, 66, 69, 85, 90, 93, 99, 145, 152, 157, 162).

Plappert, Matthias, Christian Mandery, and Tamim Asfour (2018). "Learning a bidirectional mapping between human whole-body motion and natural language using deep recurrent neural networks." In: *Robotics Auton. Syst.* (page 18).

Punnakkal, Abhinanda R. et al. (2021). "BABEL: Bodies, Action and Behavior with English Labels." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 31, 32, 85, 86, 99, 145, 152, 162, 163, 172).

Qian, Yijun et al. (2023). "Breaking The Limits of Text-conditioned 3D Motion Synthesis with Elaborative Descriptions." In: *International Conference on Computer Vision (ICCV)* (page 20).

Radford, Alec et al. (2021). "Learning Transferable Visual Models From Natural Language Supervision." In: *International Conference on Machine Learning (ICML)* (pages 5, 18, 25, 83, 85, 88, 90, 124).

Radouane, Karim et al. (2023). "Guided Attention for Interpretable Motion Captioning." In: *arXiv:2310.07324* (page 136).

Radouane, Karim et al. (2024). "Motion2language, unsupervised learning of synchronized semantic motion segmentation." In: *Neural Computing and Applications* (page 136).

Ramesh, Aditya et al. (2021). "Zero-Shot Text-to-Image Generation." In: *International Conference on Machine Learning (ICML)* (page 24).

Ramesh, Aditya et al. (2022). "Hierarchical Text-Conditional Image Generation with CLIP Latents." In: *arXiv:2204.06125* (pages 13, 145, 158).

Regneri, Michaela et al. (2013). "Grounding action descriptions in videos." In: *Transactions of the Association for Computational Linguistics (TACL)* (pages 86, 100).

Rempe, Davis et al. (2021). "HuMoR: 3D Human Motion Model for Robust Pose Estimation." In: *International Conference on Computer Vision (ICCV)* (pages 25, 112).

Rempe, Davis et al. (2023). "Trace and Pace: Controllable Pedestrian Animation via Guided Trajectory Diffusion." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 21, 23).

Rezende, Danilo Jimenez and Shakir Mohamed (2015). "Variational inference with normalizing flows." In: *International Conference on Machine Learning (ICML)* (page 16).

Richard, Alexander et al. (2021). "MeshTalk: 3D Face Animation From Speech Using Cross-Modality Disentanglement." In: *International Conference on Computer Vision (ICCV)* (page 23).

Rombach, Robin et al. (2022). "High-Resolution Image Synthesis With Latent Diffusion Models." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 13, 15, 24).

Romero, Javier, Dimitrios Tzionas, and Michael J. Black (2017). "Embodied Hands: Modeling and Capturing Hands and Bodies Together." In: *ACM Transactions on Graphics (TOG)* (pages 28, 29, 38).

Rose, Charles F. et al. (1996). "Efficient Generation of Motion Transitions using Spacetime Constraints." In: *SIGGRAPH* (page 22).

Ruiz, Alejandro Hernandez, Juergen Gall, and F. Moreno-Noguer (2019). "Human motion prediction via spatio-temporal inpainting." In: *International Conference on Computer Vision (ICCV)* (page 22).

Saharia, Chitwan et al. (2022). "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding." In: *arXiv:2205.11487* (pages 13, 145, 158).

Salzmann, Tim, Marco Pavone, and Markus Ryll (2022). "Motron: Multimodal Probabilistic Human Motion Forecasting." In: *Computer Vision and Pattern Recognition (CVPR)* (page 21).

Sanh, Victor et al. (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." In: *Energy Efficient Machine Learning and Cognitive Computing NeurIPS Workshop* (pages 18, 62, 63, 73, 74, 76, 77, 88, 145, 148, 171).

Saunders, Ben, Necati Cihan Camgoz, and Richard Bowden (2020). "Progressive Transformers for End-to-End Sign Language Production." In: *European Conference on Computer Vision (ECCV)* (page 19).

Saunders, Ben, Necati Cihan Camgoz, and Richard Bowden (2021). "Mixed SIGNals: Sign Language Production via a Mixture of Motion Primitives." In: *International Conference on Computer Vision (ICCV)* (page 19).

Schuhmann, Christoph et al. (2021). "LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs." In: *Data Centric AI NeurIPS Workshop* (pages 85, 109).

Shafir, Yonatan et al. (2023). "Human motion diffusion as a generative prior." In: *arXiv:2303.01418* (page 20).

Shahroudy, Amir et al. (2016). "NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 31, 42).

Shoemake, Ken (1985). "Animating Rotation with Quaternion Curves." In: *SIGGRAPH* (pages 149, 151, 162).

Sidenbladh, H., M. J. Black, and L. Sigal (2002). "Implicit probabilistic models of human motion for synthesis and tracking." In: *European Conference on Computer Vision (ECCV)* (pages 24, 84).

Sigal, L., A. Balan, and M. J. Black (2010). "HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion." In: *International Journal of Computer Vision (IJCV)* (page 30).

Sohl-Dickstein, Jascha et al. (2015). "Deep Unsupervised Learning using Nonequilibrium Thermodynamics." In: *International Conference on Machine Learning (ICML)* (pages 15, 19, 21).

Sohn, Kihyuk, Honglak Lee, and Xinchen Yan (2015). "Learning Structured Output Representation using Deep Conditional Generative Models." In: *Neural Information Processing Systems (NeurIPS)* (pages 14, 15, 39).

Soldan, Mattia et al. (2021). "VLG-Net: Video-language graph matching network for video grounding." In: *International Conference on Computer Vision (ICCV)* (page 101).

Soldan, Mattia et al. (2022). "MAD: A Scalable Dataset for Language Grounding in Videos From Movie Audio Descriptions." In: *Computer Vision and Pattern Recognition (CVPR)* (page 101).

Song, Kaitao et al. (2020). "MPNet: Masked and Permuted Pre-Training for Language Understanding." In: *Neural Information Processing Systems (NeurIPS)* (pages 85, 90).

Song, Ziyang et al. (2023). "ActFormer: A GAN Transformer Framework towards General Action-Conditioned 3D Human Motion Generation." In: *International Conference on Computer Vision (ICCV)* (page 145).

Starke, Sebastian et al. (2019). "Neural State Machine for Character-Scene Interactions." In: *ACM Transactions on Graphics (TOG)* (pages 16, 17, 23, 35).

Starke, Sebastian et al. (2020). "Local Motion Phases for Learning Multi-Contact Character Movements." In: *ACM Transactions on Graphics (TOG)* (page 24).

Sun, Jiangxin et al. (2022a). "You Never Stop Dancing: Non-freezing Dance Generation via Bank-constrained Manifold Projection." In: *Neural Information Processing Systems (NeurIPS)* (page 23).

Sun, Jiankai et al. (2022b). "LocATe: End-to-end Localization of Actions in 3D with Transformers." In: *arXiv:2203.10719* (page 100).

Taheri, Omid et al. (2022). "GOAL: Generating 4D Whole-Body Motion for Hand-Object Grasping." In: *Computer Vision and Pattern Recognition (CVPR)* (page 24).

Tang, Taoran, Jia Jia, and Hanyang Mao (2018). "Dance with Melody: An LSTM-Autoencoder Approach to Music-Oriented Dance Synthesis." In: *ACM International Conference on Multimedia (ACMMM)* (page 23).

Tang, Xiangjun et al. (2022). "Real-time controllable motion transition for characters." In: *ACM Transactions on Graphics (TOG)* (page 22).

Terlemez, Ömer et al. (2014). "Master Motor Map (MMM) — Framework and toolkit for capturing, representing, and reproducing human motion on humanoid robots." In: *International Conference on Humanoid Robots* (pages 26, 32, 65).

Tevet, Guy et al. (2022). "MotionCLIP: Exposing human motion generation to clip space." In: *European Conference on Computer Vision (ECCV)* (pages 18, 20, 145).

Tevet, Guy et al. (2023). "Human Motion Diffusion Model." In: *International Conference on Learning Representations (ICLR)* (pages 13, 16, 18–20, 84, 85, 113–116, 118, 122, 123, 127, 128, 131).

Troje, Nikolaus F. (2002). "Decomposing Biological Motion: A Framework for Analysis and Synthesis of Human Gait Patterns." In: *Journal of Vision* (page 30).

Trumble, Matthew et al. (2017). "Total Capture: 3D Human Pose Estimation Fusing Video and Inertial Sensors." In: *British Machine Vision Conference (BMVC)* (page 30).

Tseng, Jonathan, Rodrigo Castellon, and C Karen Liu (2023). "EDGE: Editable Dance Generation From Music." In: *Computer Vision and Pattern Recognition (CVPR)* (page 20).

Tulyakov, Sergey et al. (2018). "MoCoGAN: Decomposing Motion and Content for Video Generation." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 49, 50).

Turk, Amazon Mechanical (2023). *Amazon Mechanical Turk*. URL: https://www.mturk.com/ (page 126).

University, Simon Fraser and National University of Singapore (n.d.). *SFU Motion Capture Database*. URL: http://mocap.cs.sfu.ca/ (page 30).

Urtasun, Raquel, David J. Fleet, and Neil D. Lawrence (2007). "Modeling Human Locomotion with Topologically Constrained Latent Variable Models." In: *Human Motion – Understanding, Modeling, Capture and Animation* (page 16).

Valle-Pérez, Guillermo et al. (2021). "Transflower: probabilistic autoregressive dance generation with multimodal attention." In: *ACM Transactions on Graphics (TOG)* (page 23).

Varol, Gül et al. (2017). "Learning from Synthetic Humans." In: *Computer Vision and Pattern Recognition (CVPR)* (page 4).

Varol, Gül et al. (2018). "BodyNet: Volumetric Inference of 3D Human Body Shapes." In: *European Conference on Computer Vision (ECCV)* (page 25).

Varol, Gül et al. (2021). "Synthetic Humans for Action Recognition from Unseen Viewpoints." In: *International Journal of Computer Vision (IJCV)* (pages 4, 37, 45).

Vaswani, Ashish et al. (2017). "Attention is All you Need." In: *Neural Information Processing Systems (NeurIPS)* (pages 15, 46, 58, 61, 88).

Villegas, Ruben et al. (2018). "Neural Kinematic Networks for Unsupervised Motion Retargetting." In: *Computer Vision and Pattern Recognition (CVPR)* (page 87).

Wang, Jiashun et al. (2020). "Neural Pose Transfer by Spatially Adaptive Instance Normalization." In: *Computer Vision and Pattern Recognition (CVPR)* (page 87).

Wang, Jingbo et al. (2022a). "Towards diverse and natural scene-aware 3D human motion synthesis." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 20, 24).

Wang, Xi et al. (2022b). "Reconstructing Action-Conditioned Human-Object Interactions Using Commonsense Knowledge Priors." In: *International Conference on 3D Vision (3DV)* (page 163).

Wang, Zan et al. (2022c). "HUMANISE: Language-conditioned Human Motion Generation in 3D Scenes." In: *Neural Information Processing Systems (NeurIPS)* (page 24).

Weng, Lilian (2018). "Flow-based Deep Generative Models." In: *lilianweng.github.io*. URL: https://lilianweng.github.io/posts/2018-10-13-flow-models/ (page 16).

Weng, Lilian (2021). "What are diffusion models?" In: *lilianweng.github.io*. URL: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/ (page 14).

Witkin, Andrew and Michael Kass (1988). "Spacetime Constraints." In: *SIGGRAPH* (page 22).

Xie, Yiming et al. (2024). "OmniControl: Control Any Joint at Any Time for Human Motion Generation." In: *International Conference on Learning Representations (ICLR)* (pages 23, 112).

Xu, Hongyi et al. (2020). "GHUM and GHUML: Generative 3D Human Shape and Articulated Pose Models." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 28, 29, 38).

Xu, Jun et al. (2016). "MSR-VTT: A Large Video Description Dataset for Bridging Video and Language." In: *Computer Vision and Pattern Recognition (CVPR)* (page 19).

Xu, Liang et al. (2023). "ActFormer: A GAN-based Transformer towards General Action-Conditioned 3D Human Motion Generation." In: *International Conference on Computer Vision (ICCV)* (page 13).

Yamada, Tatsuro, Hiroyuki Matsunaga, and Tetsuya Ogata (2018). "Paired Recurrent Autoencoders for Bidirectional Translation Between Robot Actions and Linguistic Descriptions." In: *Robotics and Automation Letters* (page 18).

Yan, Sijie, Yuanjun Xiong, and Dahua Lin (2018). "Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition." In: *AAAI Conference on Artificial Intelligence* (pages 51, 52).

Yan, Sijie et al. (2019). "Convolutional Sequence Generation for Skeleton-Based Action Synthesis." In: *International Conference on Computer Vision (ICCV)* (page 16).

Yang, Jianwei et al. (2022). "Unified Contrastive Learning in Image-Text-Label Space." In: *Computer Vision and Pattern Recognition (CVPR)* (page 18).

Yang, Xu, Hanwang Zhang, and Jianfei Cai (2018). "Shuffle-Then-Assemble: Learning Object-Agnostic Visual Relationship Features." In: *European Conference on Computer Vision (ECCV)* (page 163).

Yang, Zhitao et al. (2023). "SynBody: Synthetic Dataset with Layered Human Models for 3D Human Perception and Modeling." In: *International Conference on Computer Vision (ICCV)* (page 4).

Yu, Jiahui et al. (2022). "CoCa: Contrastive Captioners are Image-Text Foundation Models." In: *Transactions on Machine Learning Research (TMLR)* (pages 25, 85).

Yu, Ruichi et al. (2017). "Visual Relationship Detection with Internal and External Linguistic Knowledge Distillation." In: *International Conference on Computer Vision (ICCV)* (page 163).

Yuan, Lu et al. (2021). "Florence: A New Foundation Model for Computer Vision." In: *arXiv:2111.11432* (page 18).

Yuan, Ye and Kris Kitani (2020). "DLow: Diversifying Latent Flows for Diverse Human Motion Prediction." In: *European Conference on Computer Vision (ECCV)* (pages 22, 35).

Yuksekgonul, Mert et al. (2022). "When and why vision-language models behave like bags-of-words, and what to do about it?" In: *International Conference on Learning Representations (ICLR)* (pages 85, 96).

Zanfir, Andrei et al. (2020). "Weakly Supervised 3D Human Pose and Shape Reconstruction with Normalizing Flows." In: *European Conference on Computer Vision (ECCV)* (page 17).

Zhang, Lvmin, Anyi Rao, and Maneesh Agrawala (2023a). "Adding Conditional Control to Text-to-Image Diffusion Models." In: *International Conference on Computer Vision (ICCV)* (pages 23, 114).

Zhang, Mingyuan et al. (2022a). "MotionDiffuse: Text-Driven Human Motion Generation with Diffusion Model." In: *arXiv:2208.15001* (pages 16, 19, 20, 85, 113, 114, 116, 118, 122, 123, 127, 128).

Zhang, Qinsheng et al. (2023b). "DiffCollage: Parallel Generation of Large Content with Diffusion Models." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 20, 114, 115, 120, 121, 126, 128).

Zhang, Tianyi et al. (2020a). "BERTScore: Evaluating Text Generation with BERT." In: *International Conference on Learning Representations (ICLR)* (pages 125, 173).

Zhang, Xiaohan et al. (2022b). "COUCH: Towards Controllable Human-Chair Interactions." In: *European Conference on Computer Vision (ECCV)* (pages 23, 24).

Zhang, Xinyi and Michiel van de Panne (2018). "Data-driven Autocompletion for Keyframe Animation." In: *Conference on Motion, Interaction and Games (MIG)* (page 22).

Zhang, Yan, Michael J. Black, and Siyu Tang (2020b). "Perpetual Motion: Generating Unbounded Human Motion." In: *arXiv:2007.13886* (page 16).

Zhang, Yan, Michael J. Black, and Siyu Tang (2021). "We Are More Than Our Joints: Predicting How 3D Bodies Move." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 22, 35, 60).

Zhang, Yan and Siyu Tang (2022c). "The Wanderings of Odysseus in 3D Scenes." In: *Computer Vision and Pattern Recognition (CVPR)* (page 23).

Zhao, Rui, Hui Su, and Qiang Ji (2020). "Bayesian Adversarial Human Motion Synthesis." In: *CVPR* (pages 16, 37).

Zhong, Chongyang et al. (2022). "Spatio-Temporal Gating-Adjacency GCN for Human Motion Prediction." In: *Computer Vision and Pattern Recognition (CVPR)* (page 21).

Zhou, Yi et al. (2019). "On the Continuity of Rotation Representations in Neural Networks." In: *Computer Vision and Pattern Recognition (CVPR)* (pages 38, 44, 45, 61, 122, 147, 171).

Zhou, Yi et al. (2020). "Generative Tweening: Long-term Inbetweening of 3D Human Motions." In: *arXiv:2005.08891* (page 22).

Zhu, Lingting et al. (2023a). "Taming Diffusion Models for Audio-Driven Co-Speech Gesture Generation." In: *Computer Vision and Pattern Recognition (CVPR)* (page 23).

Zhu, Wentao et al. (2023b). "Human motion generation: A survey." In: *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (page 16).

Zou, Shihao et al. (2020). "3D Human Shape Reconstruction from a Polarization Image." In: *European Conference on Computer Vision (ECCV)* (pages 31, 43).

Zuo, Xinxin et al. (2021). "Sparsefusion: Dynamic human avatar modeling from sparse rgbd images." In: *IEEE Transactions on Multimedia* (page 122).