

# Action-GPT: Leveraging Large-scale Language Models for Improved and Generalized Action Generation

Sai Shashank Kalakonda

sai.shashank@research.iiit.ac.in

Shubh Maheshwari

maheshwarishubh98@gmail.com

Ravi Kiran Sarvadevabhatla

ravi.kiran@iiit.ac.in

Centre for Visual Information Technology  
IIIT Hyderabad, Hyderabad, INDIA 500032

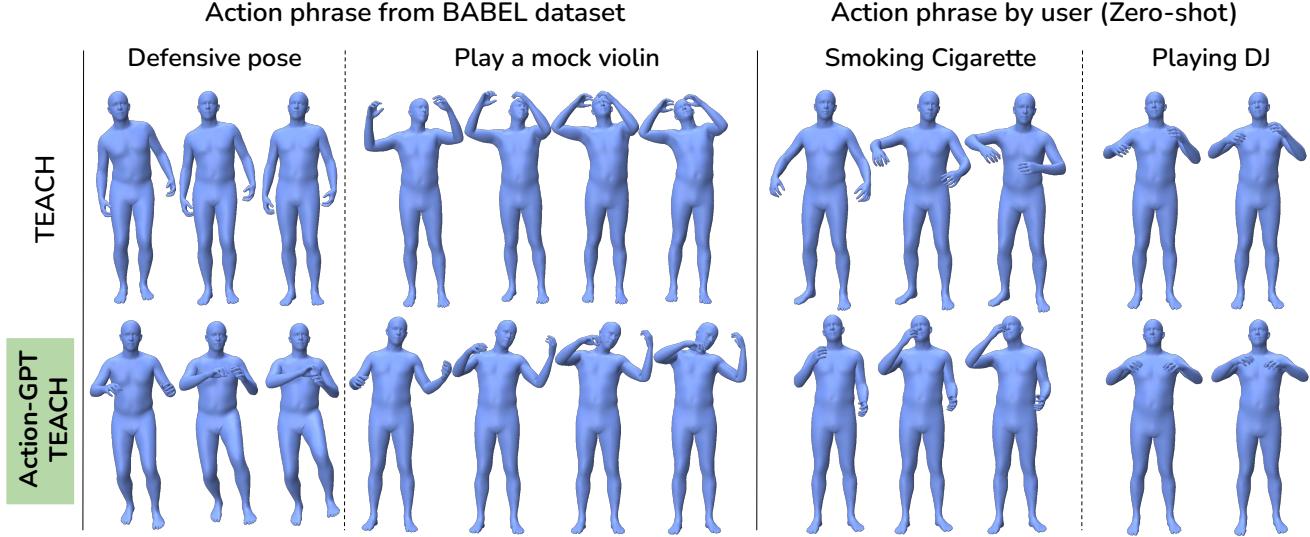


Figure 1. Sample text-conditioned action generations from a state-of-the-art model TEACH [1] (top row) and our large language model-based approach (Action-GPT-TEACH - bottom row). By incorporating large language models, our approach results in noticeably improved generation quality for seen and unseen categories. The conditioning action phrases for seen categories are taken from BABEL [2] whereas unseen action phrases were provided by a user.

## Abstract

We introduce Action-GPT, a plug-and-play framework for incorporating Large Language Models (LLMs) into text-based action generation models. Action phrases in current motion capture datasets contain minimal and to-the-point information. By carefully crafting prompts for LLMs, we generate richer and fine-grained descriptions of the action. We show that utilizing these detailed descriptions instead of the original action phrases leads to better alignment of text and motion spaces. We introduce a generic approach compatible with stochastic (e.g. VAE-based) and deterministic (e.g. MotionCLIP) text-to-motion models. In addition, the approach enables multiple text descriptions to be uti-

lized. Our experiments show (i) noticeable qualitative and quantitative improvement in the quality of synthesized motions, (ii) benefits of utilizing multiple LLM-generated descriptions, (iii) suitability of the prompt function, and (iv) zero-shot generation capabilities of the proposed approach. Code, pretrained models and sample videos will be made available at <https://actiongpt.github.io>.

## 1. Introduction

Human motion generation finds a vast set of applications spanning from entertainment (e.g. game and film industry) to virtual reality and robotics. There have been significant contributions on *category*-conditioned human motion

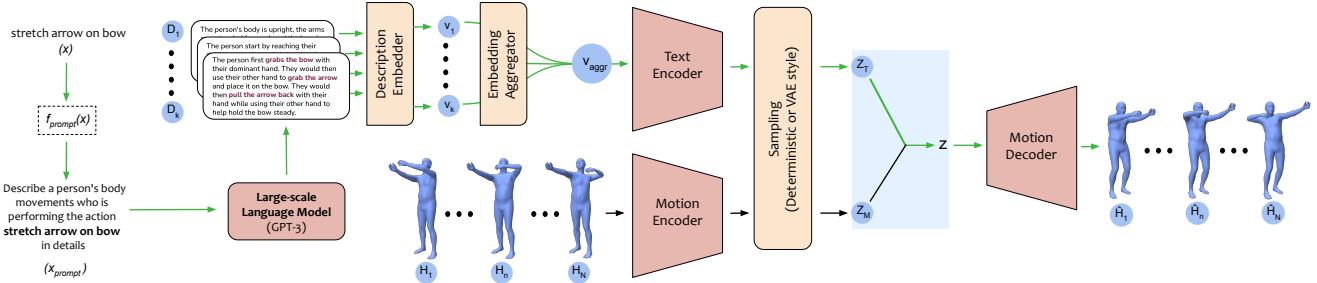


Figure 2. Action-GPT Overview: Given an action phrase ( $x$ ), we first create a suitable prompt using an engineered prompt function  $f_{\text{prompt}}(x)$ . The result ( $x_{\text{prompt}}$ ) is passed to a large-scale language model (GPT-3) to obtain multiple action descriptions ( $D_i$ ) containing fine-grained body movement details. The corresponding deep text representations  $v_i$  are obtained using Description Embedder. The aggregated version of these embeddings  $v_{\text{aggr}}$  is processed by the Text Encoder. During training, the action pose sequence  $H_1, \dots, H_N$  is processed by a Motion Encoder. The encoders are associated with a deterministic sampler (autoencoder) [3] or a VAE style generative model [1,4]. During training (shown with black), the latent text embedding  $Z_T$  and the latent motion embedding  $Z_M$  are aligned. During inference (shown in green), the sampled text embedding is provided to the Motion Decoder, which outputs the generated action sequence  $\hat{H}$ .

generations [5], with some works capable of generation at scale [6,7]. However, the generated samples are restricted to a finite set of action categories. More recent approaches focus on *text*-conditioned motion generation by constraining motion and language representations via a jointly optimized latent space [1,3,4].

The recent development of large-scale language models (LLMs) [8,9] has triggered a paradigm shift in the field of Natural Language Processing (NLP). These models, pre-trained on enormous amounts of text [10], have demonstrated impressive generalization capabilities for challenging zero-shot setting tasks such as text generation [11]. This exciting advance has also driven progress for various applications in computer vision [12,13], including the related task of pose-based human action recognition [12].

The appeal of LLM models lies in their ability to generate task-relevant text when provided a so-called *prompt* - a small piece of text - as input. Motivated by this observation and the advances mentioned above, we introduce Action-GPT, an approach that utilizes the generative power of LLMs to improve the quality and generalization capabilities of action generative models. In particular, we demonstrate that our plug-and-play approach can be used to advance existing state-of-the-art motion generation architectures in a practical manner.

Our contributions are summarized below:

- To the best of our knowledge, we are the first to incorporate Large Language Models (LLMs) for text-conditioned motion generation.
- We introduce a carefully crafted prompt function that enables the generation of meaningful descriptions for a given action phrase.
- We introduce Action-GPT, a generic plug-and-play framework which is compatible with stochastic (e.g.

VAE-based [1,4]) and deterministic (e.g. Motion-CLIP [3]) text-to-motion models. In addition, our framework enables multiple generated text descriptions to be utilized for action generation.

- Via qualitative and quantitative experiments, we demonstrate (i) noticeable improvement in the quality of synthesized motions, (ii) benefits of utilizing multiple LLM-generated descriptions, (iii) suitability of the prompt function, and (iv) zero-shot generation capabilities of the proposed approach.

Code, pretrained models and sample videos will be made available at <https://actiongpt.github.io>.

## 2. Action-GPT

Our objective is to generate an actor performing the motion conditioned on the given action phrase. The input action phrase is a natural language text that gives a high-level description of the action. It is denoted as a sequence of words  $x = [w_1, w_2, \dots, w_M]$ . The action is represented as a sequence of human poses  $H = \{H_1, \dots, H_n, \dots, H_N\}$  where  $N$  represents the number of timesteps. The human pose  $H_n \in \mathcal{R}^{|\mathcal{J}| \times 6}$ , where  $\mathcal{J}$  is the number of joints, is the parametric SMPL [14] representation which encodes the global trajectory and the parent relative joint rotation using the 6D [15] rotation representation.

Our proposed framework Action-GPT can be incorporated in an autoencoder [3] or a Variational Auto Encoder [1,4] based text-conditioned motion generation model. These motion generation models aim to generate a motion sequence conditioned on the text input by learning a joint latent space between the text and motion modalities. The key components in these models are Text Encoder  $\mathcal{T}_{\text{enc}}$ , Motion Encoder  $\mathcal{M}_{\text{enc}}$  and Motion Decoder  $\mathcal{M}_{\text{dec}}$ . The two text and motion encoders encode the text sequence

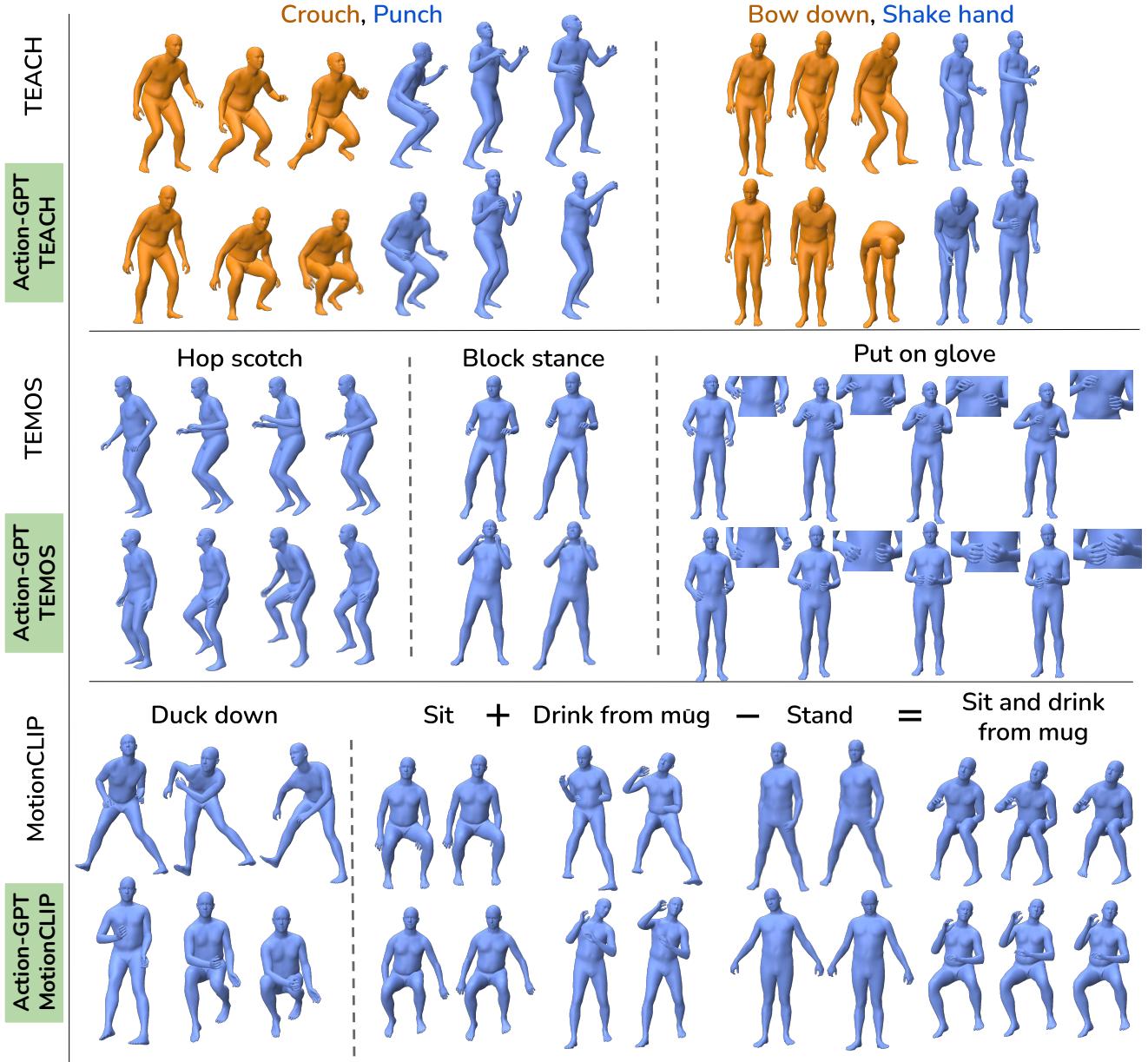


Figure 3. Visual comparison of generated motion sequences across models trained on Action-GPT framework on BABEL [2] dataset. Note that the generations using Action-GPT are well-aligned with the semantic information of action phrases. The example in the bottom right row shows latent space editing. Action-GPT is better able to transfer the *drink from mug* style from standing to sitting pose.

and motion sequence to text  $Z_T$  and motion  $Z_M$  latent embeddings of the same dimension, respectively. In the case of autoencoders, the latent embeddings are obtained in a deterministic fashion, whereas in Variational Auto Encoders, the latent embeddings are sampled from the Gaussian distribution  $\mathcal{N}(\mu, \Sigma)$ , where  $(\mu, \Sigma)$  are the outputs of the encoder. The motion decoder, on the other hand, uses the latent embedding  $Z$  as input to generate a sequence of motion poses  $\hat{H} = \{\hat{H}_1, \dots, \hat{H}_n, \dots, \hat{H}_N\}$

Fig. 2 provides an overview of our approach to incor-

porate LLM (GPT-3 in our case) into the text-conditioned motion generation models. In contrast to training directly using the action phrase  $x$  from the dataset, our framework uses carefully crafted GPT-3 generated text descriptions  $D_i$ , which provide low-level details about the movement of individual body parts. The proposed framework consists of three steps (1) Constructing a prompt function  $f_{prompt}$ , (2) Aggregating multiple GPT-3 generated text descriptions  $D_i$ , and finally, (3) utilizing the GPT-3 generated text descriptions  $D_i$  in T2M models.

## 2.1. Prompt strategy

For a given action phrase  $x$ , we generate low-level body movement details using GPT-3 [9]. GPT-3 is an autoregressive transformer model which generates human-like textual descriptions relevant to the small amount of input text provided. However, directly providing the action phrase as input to GPT-3 fails to output text containing the desired detail body movement information and leads to unrealistic motion generations (see Fig. 4). This necessitates the need for a suitable prompt function [10]. After multiple empirical trials, we determine the following prompting function  $f_{prompt}$ : `Describe a person's body movements who is performing the action [x] in detail.` Specifically, adding `Describe a person's` to the prompt restricts the description from generic information to character movement. The phrase `body movements` forces GPT-3 to explain the motion of individual body parts. Lastly, `in detail` forces the descriptions to provide low-level details. Fig. 5 showcases the importance of each component of our prompt function. We provide GPT-3 generated text description ( $D$ ) and corresponding generated action sequence  $\hat{H}$  for the action phrase `act like a dog` along with the observations in the rightmost column.

## 2.2. Aggregating multiple descriptions

Given an action prompt  $x_{prompt}$ , GPT-3 is capable of generating multiple textual descriptions  $D_1, \dots, D_k$  describing the action-specific information. The randomly generated  $k$  descriptions contain common and description-specific text segments, which enhance the overall richness of action description (see Fig. 6). Therefore, we utilize multiple descriptions as part of the text-processing pipeline. The GPT-3 generated  $k$  text descriptions  $D_1, \dots, D_k$  are passed through a Description Embedder  $D_{emb}$  to obtain corresponding description embeddings  $v_1, \dots, v_k$ . These  $k$  description embeddings are aggregated into a single embedding  $v_{aggr}$  using an Embedding Aggregator  $E_{aggr}$ . We consider average operation as our Embedding Aggregator unless stated otherwise.

## 2.3. Utilizing GPT-3 generated text descriptions in T2M models

The text encoder  $T_{enc}$  inputs the aggregated embedding  $v_{aggr}$ , and the outputs are sampled to generate text latent embeddings  $Z_T$ . In a similar fashion, motion encoder  $M_{enc}$  inputs the sequence of motion poses  $H = \{H_1, \dots, H_N\}$ , where  $H_n \in \mathcal{R}^{|J| \times 6}$  and samples motion latent embeddings  $Z_M$ . In a deterministic approach [3], the latent embeddings are generated directly as outputs of the encoder, whereas in a VAE-based approach [1, 4], the encoders generate distribution parameters  $\mu$  and  $\Sigma$ . The la-

tent embeddings are then sampled from the Gaussian distributions  $N(\mu, \Sigma)$ . On the other hand, the generated text and motion embeddings are provided to the motion decoder, which generates the human motion sequence  $\hat{H} = \{\hat{H}_1, \dots, \hat{H}_N\}$ , where  $\hat{H}_i \in \mathcal{R}^{|J| \times 6}$  which is passed through forward kinematics to generate corresponding 3D mesh sequence.

## 3. Experiments

**BABEL** [2] is a large dataset with language labels describing the actions being performed in motion capture sequences. It contains about 43 hours of mocap sequences, comprising over 65k textual labels which belong to over 250 unique action categories. We primarily focus our results on BABEL, considering its vast and diverse set of motion sequences assigned to short text sequences, which contain an average of 3-4 words. The action phrases of the BABEL dataset are to the point and precise about the action information without any additional details about the actor.

### 3.1. Models

We demonstrate our framework on state-of-the-art text conditioned motion generation models – TEMOS [2], MotionCLIP [3] and TEACH [1]. Since TEACH is an extension of TEMOS and uses only pairs of motion data of BABEL, we also demonstrate our results on TEMOS by re-training it on all single action data segments of BABEL. We train these three models as per our framework using their publicly available codes and will call them Action-GPT-[model] further.

**Action-GPT-MotionCLIP:** Similar to MotionCLIP [3], we use CLIP’s pretrained text encoder [16] as our description embedder, but we inputs all the  $k$  text descriptions and generate the aggregated vector representation  $v_{aggr}$  using the embedding aggregator. Note that similar to MotionCLIP, we use CLIP-ViT-B/32 frozen model. We follow the same motion auto-encoder setup as that of MotionCLIP. There is no additional text encoder and sampling process as the constructed  $v_{aggr}$  itself is used as the text embedding  $Z_T$ , and the output of the motion encoder is used as the motion embedding  $Z_M$ .

**Action-GPT-TEMOS:** Instead of providing action phrase text, we pass multiple textual descriptions extracted from GPT-3 to DistilBERT [17] to obtain description embedding  $v_i \in R^{n_i \times e}$ , where  $n_i$  is the number of words in description  $D_i$  and  $e$  is the DistilBERT embedding dimension, thus  $v_{aggr} \in R^{G \times e}$  where  $G = \max(n_i)$ . Note that similar to TEMOS, we use pre-trained DistilBERT and freeze its weights during training. The text encoder, motion encoder, and motion decoder used is the same as that of TEMOS. The sampled text embedding  $Z_T$  and motion embedding  $Z_M$  are both  $\in R^d$  where  $d$  is the dimension of latent space.

Action phrase( $x$ )	GPT-3( $x$ ) output	Generated sequence	GPT-3( $x_{\text{prompt}}$ ) output	Generated sequence
Duck down	Duck down is an action typically used to avoid something coming towards you such as a ball or someone trying to hit you.		The person's body movements who is performing the action duck down would be to <b>lower their body down closer to the ground</b> , while bending their knees.	
Javelin throw	The javelin throw is an athletic event where the goal is to throw a javelin as far as possible. The javelin is a spear-like object that is thrown with the arm and hand.		The person stands with their <b>feet shoulder-width apart</b> and <b>toes pointing forward</b> . They grip the <b>javelin with their dominant hand</b> in the middle of the shaft and their other hand near the point. They bend their <b>elbow</b> and raise their <b>arm back behind their head</b> . As they throw, they <b>extend their arm</b> , release the <b>javelin</b> , and <b>follow through with their arm and body</b> .	

Figure 4. This figure highlights the importance of the prompt function. Observe that directly feeding the action phrase text ( $x$ ) to GPT-3 results in poor-quality generations. In contrast, the fine-grained body movement details in the prompt-based text enable higher fidelity generations (last column). Note that the coloured text descriptions correspond to different body movement details.

Prompt Function ( $f_{\text{prompt}}$ )	GPT-3 generated text description (D)	Generated Action Sequence ( $\hat{H}$ )	Observations
Describe the action [ $x$ ]	The action of acting like a dog would involve someone imitating the behavior of a dog. This could involve barking, wagging their tail, playing fetch, or any other number of dog-like behaviors.		Generated text description contains generic information of act like a dog without any details of body movements, resulting in an unrealistic action generation.
Describe a person performing the action [ $x$ ]	A person who is acting like a dog <b>might crawl on the ground</b> , bark, and wag their tail.		Generated text description contains high-level body movements information, resulting in an action generation which just captures minimal high-level details.
Describe a person's body movements who is performing the action [ $x$ ]	The person would be down <b>on all fours</b> , with their <b>hands and feet touching the ground</b> . Their <b>back would be arched</b> and their <b>head would be held high</b> . They would be <b>moving their legs and arms in a coordinated way</b> , similar to how a dog would move		Generated text description contains required low-level body movements information, resulting in a realistic action generation, but suffers in capturing fine-grained details.
Describe a person's body movements who is performing the action [ $x$ ] in detail	The person would be <b>on all fours</b> with their <b>palm and feet flat on the ground</b> . Their <b>legs would alternate moving forward</b> and their <b>hips would sway from side to side</b> . Their <b>arms would move in rhythm with their legs</b> . Their <b>head would be up</b> and their <b>gaze would be forward</b> .		Generated text description contains required low-level fine-grained body movements information, resulting in a realistic action generation. (Additional details, legs would alternate moving forward enabled in better generation)

Figure 5. The table showcases the descriptions generated by GPT-3 (D), generated action sequences ( $\hat{H}$ ) for the action phrase ( $x = \text{act like a dog}$  using different prompt strategies along with the observations (right most column). Notice that our prompt function (bottom row) generates the highest amount of required body movement descriptions, generating the most realistic action sequence. Note that the coloured text descriptions correspond to the body movement details.

**Action-GPT-TEACH:** Since TEACH [1] is an extension of TEMOS, the process of generating description embeddings  $v_i$  is the same as that of in Action-GPT-TEMOS. The text encoder, motion encoder, and motion decoder are used the same as that of TEACH. As TEACH is trained on the pairs of action data, the training iteration consists of two forward passes where an action phrase and its corresponding motion sequence are provided as input in each pass. In addition, a set of the last few frames of generated motion in the first pass are also provided as input in the second pass. In both passes, our framework uses the generated description embeddings corresponding to the input action phrases.

Detailed diagrams illustrating the differences between original architectures and their LLM-based variants along

with the additional details regarding training and testing can be found in the appendix 5.

### 3.2. Implementation details

We access GPT-3 via OpenAI API Beta Access program. Unless stated otherwise, we use the largest GPT-3 model available, `davinci-002`. The Action-GPT prompt strategy consumes a maximum of 140 tokens together for prompt and generation. We use the completions API endpoint with the parameters temperature and top-p set to 0.5 and 1, ensuring we have well-defined diverse descriptions. All the other parameters are set to default. We conduct all our experiments on cluster machines with Intel Xeon E5 2640 v4 and Nvidia GeForce GTX Ti 12GB GPUs with

## Action phrase : Stomach ache

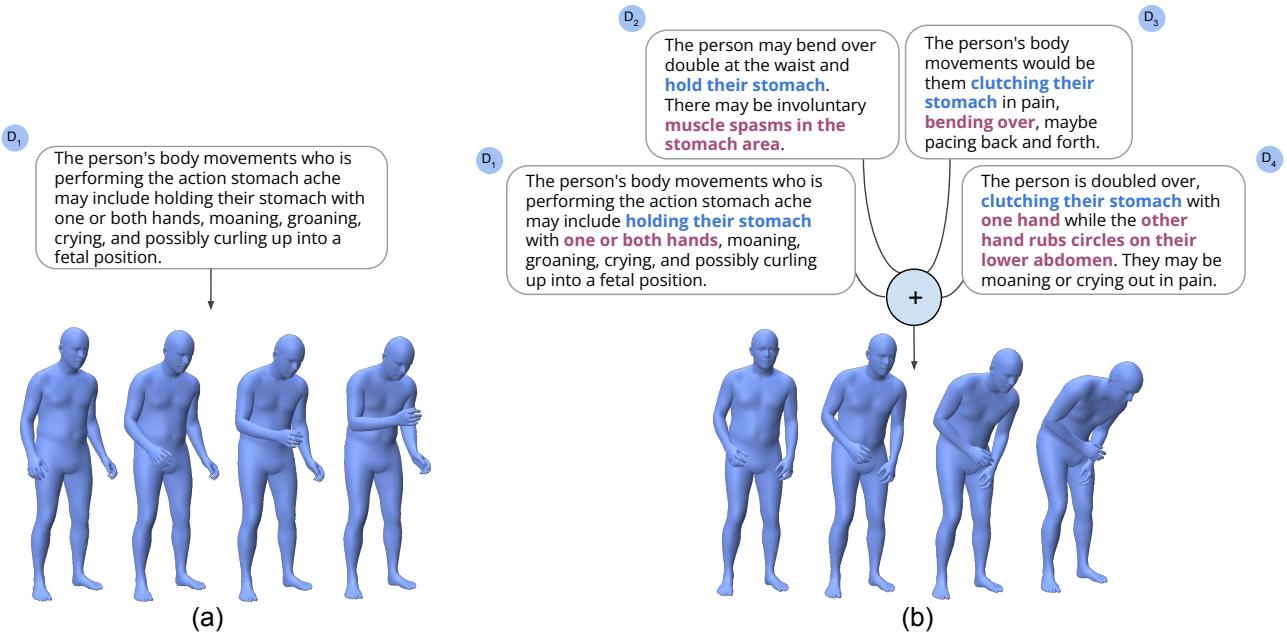


Figure 6. This figure highlights the importance of using multiple GPT-3 generated descriptions ( $D_1, \dots, D_k$ ),  $k = 4$  for each action phrase in Action-GPT framework for TEACH [1]. Notice the visibly improved generation quality when multiple prompted descriptions are used (right column). Body movement text common across descriptions is highlighted in blue. Movements unique to each description are highlighted in pink.

Ubuntu 16.04 OS.

### 3.3. Quantitative analysis

We follow the metrics employed in TEACH [1] for quantitative evaluation, 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 correspond to the joint position in the local coordinate system (with respect to the root), whereas mean global corresponds to the joint position in the global coordinate system. The APE and AVE for a joint are the averages of the L2 distances between the generated and ground truth joint positions and variances, respectively – refer to appendix 8 for details. Tab. 1 summarizes the results of using our framework in comparison with the default setup for each model. Incorporating detailed descriptions using GPT-3 shows an improvement over all the APE (except for MotionCLIP) and AVE metrics. The metrics of root joints for MotionCLIP are empty since it generates only local pose without any locomotion.

### 3.4. Qualitative analysis

In Fig. 3, we provide qualitative comparisons of the model generations. We observe that the generations from our framework are more realistic and well-aligned with the semantic information of the action phrases compared to the default approach. The generations are able to capture the

low-level fine-grained details of the action suggested by the original text phrase input. Please refer to appendix for video examples of zero-shot generations, comparisons with baselines, and examples showing diversity in generated sequences.

### 3.5. Ablations

We perform an ablation study to understand the underlying effects of the Action-GPT framework. All of the ablation experiments are carried out on the Action-GPT-TEACH model unless stated otherwise, as it is capable of handling a series of action phrases as input.

**Number of GPT-3 Text Sequences:** We analyzed the influence of number of generated descriptions in Action-GPT-TEACH framework by varying  $k$  in 1, 2, 4 and 8. We observed that for all the values of  $k$ , Action-GPT-TEACH performs better than the default TEACH and the best results are obtained for  $k = 4$  (see Tab. 2). Increasing the value of  $k$  up to a certain value improves performance. However, aggregating too many descriptions can lead to the injection of excessive noise, which dilutes the presence of text related to body movement details.

**Language Model Capacity:** Open AI provides GPT-3 in different model capacities, `davinci` being the largest. We analyzed the influence of `curie`, the second largest GPT-3 model, on the motion sequence generations of the Action-GPT-TEACH ( $k = 4$ ) framework. Results show that having

Model	Method	Average Positional Error ↓				Average Variance Error ↓			
		root joint	global traj	mean local	mean global	root joint	global traj	mean local	mean global
MotionCLIP	Default	-	-	0.556	0.541	-	-	0.056	0.02
	Action-GPT	-	-	0.590	0.571	-	-	<b>0.042</b>	<b>0.019</b>
TEMOS	Default	0.597	0.574	0.162	0.644	0.113	0.112	0.010	0.122
	Action-GPT	<b>0.561</b>	<b>0.540</b>	<b>0.151</b>	<b>0.605</b>	<b>0.101</b>	<b>0.100</b>	<b>0.010</b>	<b>0.109</b>
TEACH	Default	0.674	0.654	0.159	0.717	0.222	0.220	0.014	0.234
	Action-GPT	<b>0.606</b>	<b>0.586</b>	<b>0.159</b>	<b>0.650</b>	<b>0.204</b>	<b>0.202</b>	<b>0.014</b>	<b>0.216</b>

Table 1. Quantitative evaluation on the BABEL test set

Architectural Component	Ablation Details	Average Positional Error ↓				Average Variance Error ↓			
		root joint	global traj	mean local	mean global	root joint	global traj	mean local	mean global
number of generated descriptions ( $K$ )	$k = 1$	0.655	0.635	0.159	0.698	0.216	0.214	0.015	0.228
	$k = 2$	0.637	0.617	0.158	0.680	0.211	0.209	0.015	0.223
	$k = 8$	0.632	0.613	<b>0.157</b>	0.674	0.212	0.210	0.015	0.224
GPT-3 Capacity	curie	0.642	0.622	0.159	0.680	0.216	0.214	0.015	0.228
<b>Ours</b> ( $k = 4$ )	davinci	<b>0.606</b>	<b>0.586</b>	0.158	<b>0.650</b>	<b>0.204</b>	<b>0.202</b>	<b>0.014</b>	<b>0.216</b>

Table 2. Performance scores for ablative variants.

a larger model capacity helps in generating more realistic motion sequences, as the generated text descriptions provide much relevant and detailed information as required.

## 4. Discussion and Conclusion

The key to good quality and generalizable text-conditioned action generation models lies in improving the alignment between the text and motion representations. Through our Action-GPT framework, we show that such alignment can be achieved efficiently by employing Large Language Models whose operation is guided by a judiciously crafted prompt function. Sentences in GPT-generated descriptions contain procedural text which corresponds to sub-actions. During training, the regularity and frequency of such procedural text likely enable better alignment with corresponding motion sequence counterparts. We also hypothesize that the diversity of procedural sentences in the descriptions enables better compositionality for unseen (zero-shot) generation settings.

The plug-and-play nature of our approach is practical for adoption within state-of-the-art text-conditioned action generation models. Our experimental results demonstrate the generalization capabilities and action fidelity improvement for multiple adopted models, qualitatively and quantitatively. In addition, we also highlight the role of various prompt function components and the benefit of utilizing multiple prompts for improved generation quality.

## References

- [1] Nikos Athanasiou, Mathis Petrovich, Michael J. Black, and Güll Varol, “TEACH: Temporal Action Compositions for 3D Humans,” in *International Conference on 3D Vision (3DV)*, 2022. [1](#), [2](#), [4](#), [5](#), [6](#), [9](#), [10](#)
- [2] Abhinanda R. Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J. Black, “BABEL: Bodies, action and behavior with english labels,” in *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 722–731. [1](#), [3](#), [4](#), [9](#)
- [3] Guy Tevet, Brian Gordon, Amir Hertz, Amit H Bermano, and Daniel Cohen-Or, “Motionclip: Exposing human motion generation to clip space,” *arXiv preprint arXiv:2203.08063*, 2022. [2](#), [4](#), [9](#), [10](#)
- [4] Mathis Petrovich, Michael J. Black, and Güll Varol, “TEMOS: Generating diverse human motions from textual descriptions,” in *European Conference on Computer Vision (ECCV)*, 2022. [2](#), [4](#), [9](#), [10](#)
- [5] Mathis Petrovich, Michael J. Black, and Güll Varol, “Action-conditioned 3D human motion synthesis with transformer VAE,” in *International Conference on Computer Vision (ICCV)*, 2021. [2](#)
- [6] Shubh Maheshwari, Debutanu Gupta, and Ravi Kiran Sarvadevabhatla, “Mugl: Large scale multi person conditional action generation with locomotion,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2022, pp. 257–265. [2](#)
- [7] Debutanu Gupta, Shubh Maheshwari, Sai Shashank Kalakonda, Manasvi, and Ravi Kiran Sarvadevabhatla, “Dsag: A scalable deep framework for action-conditioned multi-actor full body motion synthesis,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2023. [2](#)
- [8] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro, “Megatron-lm: Training multi-billion parameter

- language models using model parallelism,” 2019, cite arxiv:1909.08053. 2
- [9] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., “Language models are few-shot learners,” *arXiv preprint arXiv:2005.14165*, 2020. 2, 4
- [10] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig, “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing,” 2021. 2, 4
- [11] Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park, “Gpt3mix: Leveraging large-scale language models for text augmentation,” 2021. 2
- [12] Wangmeng Xiang, Chao Li, Yuxuan Zhou, Biao Wang, and Lei Zhang, “Language supervised training for skeleton-based action recognition,” 2022. 2
- [13] Sarah Pratt, Rosanne Liu, and Ali Farhadi, “What does a platypus look like? generating customized prompts for zero-shot image classification,” 2022. 2
- [14] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black, “SMPL: A skinned multi-person linear model,” *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 248:1–248:16, Oct. 2015. 2
- [15] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao, “On the continuity of rotation representations in neural networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2
- [16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, “Learning transferable visual models from natural language supervision,” in *Proceedings of the 38th International Conference on Machine Learning*, Marina Meila and Tong Zhang, Eds. 18–24 Jul 2021, vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, PMLR. 4
- [17] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf, “Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter,” 2019. 4, 9

# Appendix

## 5. Models

We demonstrate our framework on state-of-the-art text conditioned motion generation models – TEMOS [2], MotionCLIP [3] and TEACH [1], all trained on BABEL [2]. We name their LLM (i.e. GPT here) based variants as Action-GPT-[model].

**Action-GPT-MotionCLIP:** In MotionCLIP [3], for a given action phrase  $x$ , the CLIP text embedding of the phrase  $x$  is considered as its corresponding latent text embedding  $Z_T$ , whereas in our Action-GPT framework the aggregated vector embedding  $v_{aggr} \in R^c$ , where  $c$  is the CLIP text embedding dimension, constructed for the action phrase  $x$  is considered as its latent text embedding  $Z_T$ . In detail, we first construct  $x_{prompt}$  using the action phrase  $x$  and prompt function  $f_{prompt}$ . The  $x_{prompt}$  is then input to LLM (i.e. GPT-3) to generate  $k$  textual descriptions  $D_i$ . Using the CLIP text encoder, we then construct  $k$  corresponding CLIP text embeddings  $v_i$ . These  $k$  CLIP text embeddings  $v_i$  are then aggregated into a single embedding  $v_{aggr}$  using an Embedding aggregator (average operation here). The constructed  $v_{aggr}$  is the corresponding latent text embedding  $Z_T$  for the action phrase  $x$ . (see Fig. 7)

We trained this model using the same training configurations as mentioned in MotionCLIP [3] and provided the results on the test split. We generated the metrics for the baseline MotionCLIP [3] using the pre-trained model provided.

**Action-GPT-TEMOS:** Instead of providing the action phrase  $x$ , directly to DistilBERT [17] as that in TEMOS [4], we first construct  $x_{prompt}$  and generate  $k$  textual descriptions  $D_i$  using LLM (i.e. GPT-3). These  $k$  textual descriptions are then input to DistilBERT [17] to obtain  $k$  corresponding description embeddings  $v_i \in R^{n_i \times e}$ , where  $n_i$  is the number of words in description  $D_i$  and  $e$  is the DistilBERT embedding dimension. The  $k$  description embeddings are aggregated to a single embedding  $v_{aggr} \in R^{G \times e}$ , where  $G = \max(n_i)$  using the average operation. This aggregated embedding  $v_{aggr}$  is then used as input to Text Encoder along with the text distribution tokens  $\mu_{token}^T$  and  $\sum_{token}^T$ . The later training and inference process is carried out the same as that of in TEMOS [4]. (see Fig. 8)

We trained this model on a 4 GPU setup with a batch size of 4 on each GPU, keeping rest all the parameters same as provided in TEMOS [4]. We trained the baseline model of TEMOS [4] and its LLM-based variant on BABEL [2] using the single action data segments provided by TEACH [1] and generated the metrics on the corresponding test set.

**Action-GPT-TEACH:** Since TEACH [1] is an extension of TEMOS [4] the process of generating aggregated description embedding  $v_{aggr}$  is same as that of Action-

GPT-TEMOS. In addition, TEACH can generate an action sequence for a series of action phrases as input  $x = \{x^1, \dots, x^i, \dots, x^s\}$ . So, we compute  $v_{aggr}$   $s$  number of times, once for each phrase  $x^i$ . In detail, for an action phrase  $x^i$  we generate  $k$  text descriptions  $D_1^i, \dots, D_j^i, \dots, D_k^i$ . The  $k$  text descriptions are input to DistilBERT to generate corresponding description embeddings  $v_1^i, \dots, v_j^i, \dots, v_k^i$ , where  $v_j^i \in R^{n_j^i \times e}$ , where  $n_j^i$  is the number of words in description  $D_j^i$  and  $e$  is the DistilBERT embedding dimension. The  $k$  description embeddings  $v_j^i$  are aggregated to a single embedding  $v_{aggr}^i \in R^{G \times e}$ , where  $G = \max(n_j^i)$  using the average operation. The aggregated embedding  $v_{aggr}^i$  is input to Past-conditioned Text Encoder  $T_{enc}$  along with the learnable tokens  $\mu_{token}, \sum_{token}$ , SEP token and  $I_{N_{i-1}-P:N_{i-1}}^{i-1}$ , the motion features generated using Past Encoder, corresponding to the last  $P$  frames of previous generated action sequence  $\hat{H}_{N_{i-1}-P:N_{i-1}}^{i-1}$ . The later training and inference process followed is the same as that of TEACH.(see Fig. 9)

Similar to Action-GPT-TEMOS, we trained this model on a 4 GPU setup with a batch size of 4 on each GPU, keeping rest all the parameters same as provided in TEACH [1]. We generated the metrics for the baseline TEACH [1] using the pre-trained model provided.

## 6. Diverse Generations

Our Action-GPT framework can generate diverse action sequences for a given action phrase  $x$ , utilizing the capability of LLMs to generate diverse text descriptions for a given prompt. We generate multiple text descriptions  $D_i$  for an action phrase  $x$ , and using them as input, multiple action sequences  $\hat{H}_i$  are generated.

## 7. Zero-Shot Generations

Our approach enables the generation of action sequences  $\hat{H}$  for unseen action phrases (zero-shot). The intuition behind this is that our Action-GPT framework uses low-level detailed body movements textual information to align text and motion spaces instead of using the action phrase directly. Hence the action phrase might be unseen, but the low-level body movement details in the generated corresponding text descriptions  $D_i$  will not be completely unseen.

Results corresponding to comparison of the baseline models to their LLM-based variants, diverse and zero-shot generations can be found at <https://actiongpt.github.io>.

## 8. Metrics

We show results using the generative quality metrics followed by [1,4], namely Average Positional Error (APE) and

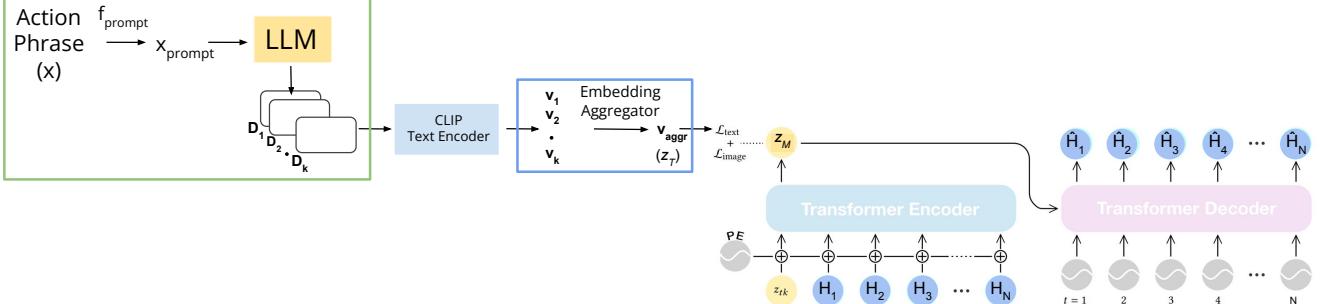


Figure 7. Action-GPT-MotionCLIP Overview: We extend MotionCLIP [3] by incorporating LLM (i.e. GPT-3). The box highlighted in green showcases the generation of  $k$  text descriptions  $D_i$  as the output of LLM on input  $x_{prompt}$ , which is constructed using the prompt function  $f_{prompt}$  and action phrase  $x$ . The box highlighted in blue showcases the aggregation of description embeddings  $v_i$ , outputs of the CLIP text encoder. The aggregated embedding  $v_{agg}$  is considered as the latent text embedding  $Z_T$ , on which the text loss  $\mathcal{L}_{text}$  is computed. All the other components apart from the highlighted boxes represent the original architecture of MotionCLIP [3].

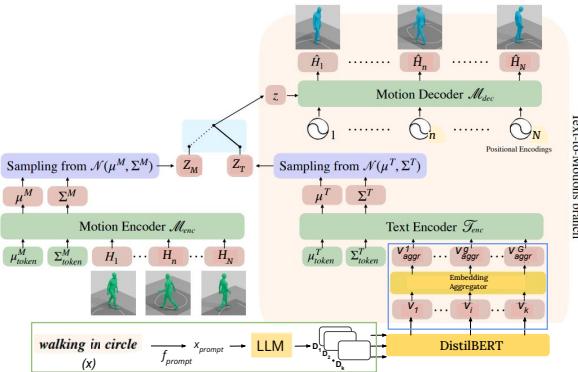


Figure 8. Action-GPT-TEMOS Overview: We extend TEMOS [4] by incorporating LLM (i.e. GPT-3). The box highlighted in green showcases the generation of  $k$  text descriptions  $D_i$  using LLM on input  $x_{prompt}$ , which is constructed using the prompt function  $f_{prompt}$  and action phrase  $x$ . The  $k$  text descriptions  $D_i$  are input to DistilBERT to generate corresponding description embeddings  $v_i$ . The box highlighted in blue showcases the aggregation of description embeddings  $v_i$  to  $v_{agg}$  using an embedding aggregator. All the other components apart from the highlighted boxes represent the original architecture of TEMOS [4].

**Average Variational Error (AVE).** For all the metrics, the smaller the score, the better the generative quality.

**Average Positional Error (APE) :** For a joint  $j$ , APE is calculated as the average of the L2 distances between the generated and ground truth joint positions over the timesteps ( $N$ ) and the number of test samples ( $S$ ).

$$APE[j] = \frac{1}{SN} \sum_{s \in S} \sum_{n \in N} \|\hat{H}_{s,n}[j] - H_{s,n}[j]\|_2$$

**Average Variational Error (AVE) :** For a joint  $j$ , AVE is calculated as the average of the L2 distances between the generated and ground truth variances.

$$AVE[j] = \frac{1}{S} \sum_{s \in S} \|\hat{\sigma}_s[j] - \sigma_s[j]\|_2$$

where  $\sigma[j]$  denotes the variance of the joint  $j$ ,

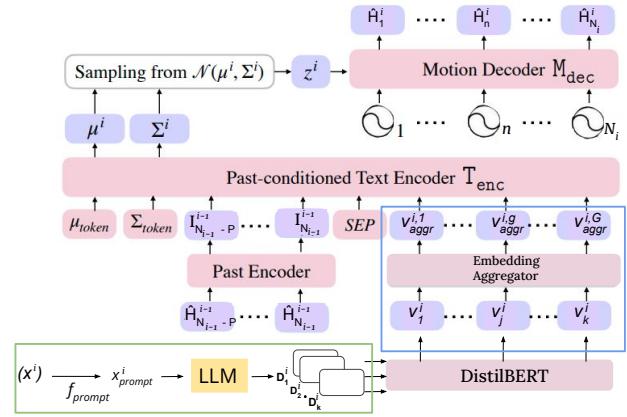


Figure 9. Action-GPT-TEACH Overview: We extend TEACH [1] by incorporating LLM (i.e. GPT-3). As TEACH can generate an action sequence for a series of action phrases, we use  $i^t$  to denote the  $i^{th}$  phrase. The box highlighted in green showcases the  $k$  generated text descriptions  $D_i^t$  using LLM on input  $x_{prompt}^i$ , which is constructed using the prompt function  $f_{prompt}$  and action phrase  $x^i$ . The  $k$  text descriptions  $D_i^t$  are input to DistilBERT to generate corresponding sentence embeddings  $v_j^i$ , which are aggregated into a single embedding  $v_{agg}^{i,1:G}$  using an embedding aggregator. All the other components apart from the highlighted boxes represent the original architecture of TEACH [1].

$$\sigma[j] = \frac{1}{N-1} \sum_{n \in N} (\tilde{H}[j] - H_n[j])^2$$

$\tilde{H}[j]$  is calculated as the mean of the joint  $j$  over  $N$  timesteps.

We calculate four variants of errors for both APE and AVE,

- *root joint* error is calculated on the root joint using all the 3 coordinates  $X, Y$  and  $Z$ .
- *global traj* error is calculated on the root joint using only the 2 coordinates  $X$  and  $Y$ .
- *mean local* error is calculated as the average of all the

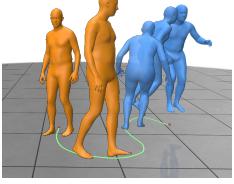
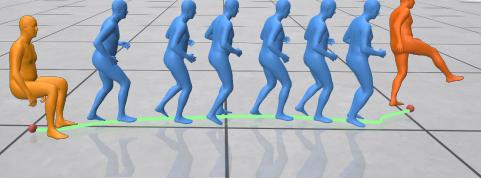
Action phrase( $x$ )	GPT-3 generated text description (GPT-3( $x_{\text{prompt}}$ ))	Generated Action Sequence ( $\mathcal{H}$ )
walk in a circle, hop over obstacle	<p>The person's body is upright, and their feet are moving forward in a rhythmic pattern. Their arms are swinging back and forth, and their head is facing forward. They are turning their body in a circular motion as they walk.</p> <p>The person's body movements would include bending their knees and jumping into the air, clearing the obstacle in front of them.</p>	
sit, run, kick	<p>The person sits down on a chair. The person's back would be straight and their feet would be flat on the ground. They would then place their hands by their sides.</p> <p>The person's body is moving forward at a quick pace. The legs are moving quickly and alternately, propelling the body forward. The arms are also moving quickly, helping to balance the body.</p> <p>The person raises their leg up behind them and then swing their leg forward, making contact with their foot or ankle.</p>	

Figure 10. Actions with locomotion generated using Action-GPT-TEACH. The color of the text (column 2) represents the detailed text generated by GPT-3 and the same color of the mesh represents the pose sequence generated for the sub-action. The green curve shows the trajectory. The red points show the starting and end points of the motion. Action-GPT is able to generate diverse examples involving locomotion such as walk in a circle, run, hop over obstacle and kick.

joint errors in the local coordinate system with respect to the root joint.

- *mean global error* is calculated as the average of all the joint errors in the global coordinate system.

## 9. Locomotion and root movement

Fig. 10 shows the diverse actions **involving locomotion** generated using Action-GPT. Our prompt strategy provides detailed descriptions which include the direction of the locomotion along with the coordination among relevant body parts. We further verify this by quantitatively evaluating using Average Positional Error and Average Variance Error of the global trajectory metric – see Table. 1. We observe that the Action-GPT variants are better aligned with root trajectory than the baseline.

## 10. Current limitations

- *Finger motion:* Current frameworks use SMPL to represent the pose. SMPL does not contain detailed finger joints. Therefore, GPT-generated descriptions for actions requiring detailed finger motion such as rock-paper-scissors, pointing fingers cannot be generated satisfactorily by the current framework.
- *Complex Actions:* Actions containing complex and vigorous body movements such as yoga and dance poses cannot be generated by our current framework.
- *Long duration action sequences:* Due to the limited

duration of training data action sequences (< 10 secs), our method cannot generate long sequences.

Model	Method	Avg. Training Time (secs)	Avg. Inference Time (secs)
MotionCLIP	Default	585 - 598	0.32 - 0.34
	Action-GPT	700 - 712	0.53 - 0.62
TEMOS	Default	225 - 230	0.8 - 0.96
	Action-GPT	255 - 260	1.44 - 1.76
TEACH	Default	234 - 240	1.6 - 2.1
	Action-GPT	315 - 320	3.4 - 3.8

Table 3. Computation costs of baselines and their Action-GPT ( $k = 4$ ) variants.

## 11. Computation cost analysis

There will be no change in the number of parameters of the Action-GPT variants when compared to the baseline models as we are using the frozen pre-trained text embedding models. Although the computation time of the models both during training and inference is higher for the Action-GPT ( $k = 4$ ) variants when compared with the original baselines as shown in Table. 3. For training time, we take the mean of the time taken for hundred epochs. For inference time, we take a batch-size of 16 and average over 10 repetitions. The increase in the computation time of the Action-GPT variants is because of the usage of  $k$  GPT-3 text descriptions, each containing around 128 words, whereas, in the baseline models, a single action phrase is used, which contains around 5 – 8 words.