Real-time Animation Generation and Control on Rigged Models via Large Language Models

Han Huang
Rensselaer Polytechnic Institute & Microsoft

Fernanda De La Torre
MIT & Microsoft

Cathy Mengying Fang
MIT Media Lab & Microsoft

Andrzej Banburski-Fahey
Microsoft

Judith Amores
Microsoft

Jaron Lanier
Microsoft

Abstract

We introduce a novel method for real-time animation control and generation on rigged models using natural language input. First, we embed a large language model (LLM) in Unity to output structured texts that can be parsed into diverse and realistic animations. Second, we illustrate LLM's potential to enable flexible state transition between existing animations. We showcase the robustness of our approach through qualitative results on various rigged models and motions.

1 Introduction

While recent advances in deep learning have revolutionized many aspects of computer graphics [5, 6, 7, 13, 3, 4, 14, 2, 9], few works have explored direct actuation of rigged 3D models. In this context, we present an innovative framework that leverages large language models (LLM) to enable real-time animation generation and control.

First, we generate novel animations on a given model using only natural language descriptions. Our method outputs structured strings encoding positional and rotational time series for each joint, which are parsed to produce animations on the rigged object. We showcase the generated animations on hierarchically distinct models with a variety of motions to underscore the robustness of our approach.

Second, we integrate a LLM with Unity [18] to program animation transition on humanoid characters via the generation and execution of appropriate Unity C# scripts. Our approach is characterized by its flexibility, allowing for the seamless integration of pre-existing animations with custom game logic.

2 Methodology and Results

Animation Generation Formally, we abstract a rigged 3D model as a tree $T = (V, E)$ encoding its joint hierarchy. An animation on the model is then a set of motion time series (p_i^v, q_i^v) associated with each joint $v \in V$, where $p_i^v \in \mathbb{R}^3, q_i^v \in \mathbb{R}^4$ are the joint position and rotation (as a quaternion) at time $t_i \in [0, t_{end}]$, respectively. To generate animations on a rigged object, we utilize a LLM to output structured texts containing appropriate joint hierarchies and motions according to the user prompt. We show an example in Figure 2 and remark that numerical values are also treated as text tokens during generation. To overcome token size limits, we compress the animation strings to only output motion values at *keyframes* and truncate all floating-point numbers to a single significant figure.

We illustrate novel animations generated for a whale, a pig, and a raccoon in Figure 1. Here, each model comes with an existing animation, which we use for in-context learning[1]. Overall, the generated motions are visually realistic and can be produced within a matter of seconds. Our framework also exhibits a high degree of semantic comprehension on joint hierarchies by producing

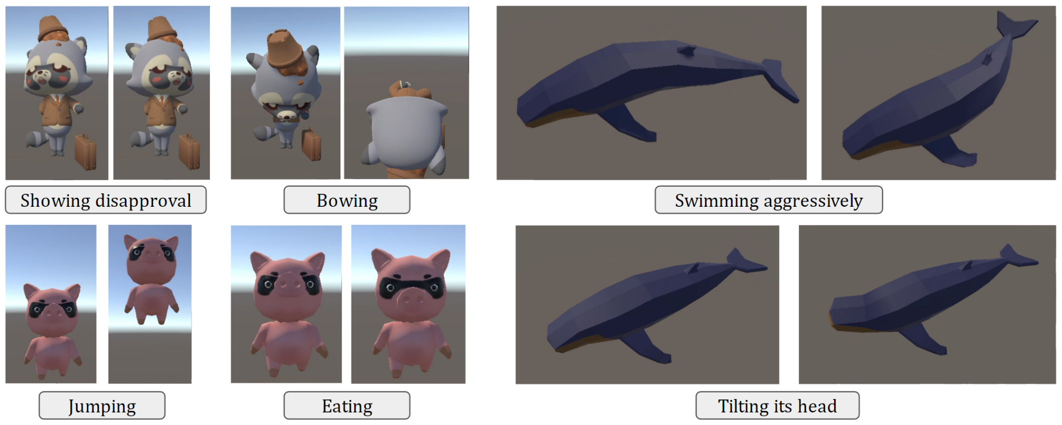
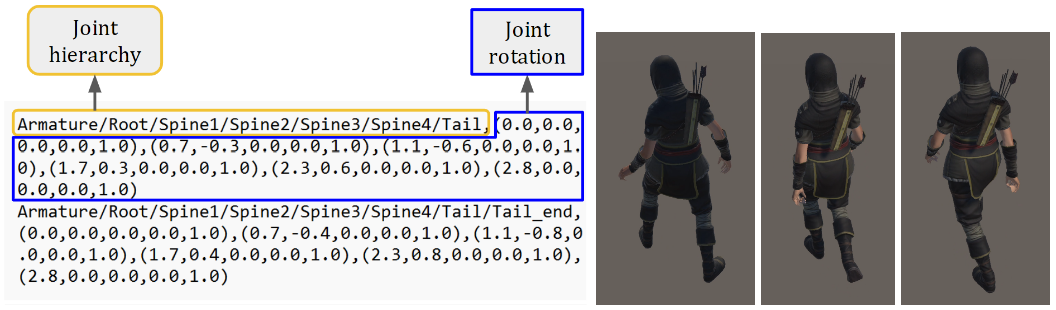


Figure 1: One-shot animation generation on rigged models. Text bubbles contain the prompts used.



```
Armature/Root/Spine1/Spine2/Spine3/Spine4/Tail,(0.0,0.0,
0.0,0.0,1.0),(0.7,-0.3,0.0,0.0,1.0),(1.1,-0.6,0.0,0.0,1.
0),(1.7,0.3,0.0,0.0,1.0),(2.3,0.6,0.0,0.0,1.0),(2.8,0.0,
0.0,0.0,1.0)
Armature/Root/Spine1/Spine2/Spine3/Spine4/Tail/Tail_end,
(0.0,0.0,0.0,0.0,1.0),(0.7,-0.4,0.0,0.0,1.0),(1.1,-0.8,0
.0,0.0,1.0),(1.7,0.4,0.0,0.0,1.0),(2.3,0.8,0.0,0.0,1.0),
(2.8,0.0,0.0,0.0,1.0)
```

Figure 2: Left: animation string generated with the prompt: "make the whale flapping its tail" on a whale model. Each vector is a quarternion followed by a time stamp: (q_0, q_1, q_2, q_3, t); Right: Animation control on a character with existing animations. Prompt: "Make the 'archer' move around randomly and pauses once in a while with correct animations."

motions on the appropriate bones. For instance, when prompted with "tilting its head" for a whale, our model translates this into motion within the head joint while keeping the main body still. Remarkably, the animated models are rich in structural diversity, which underscores the robustness of our framework in accommodating distinct anatomies and motion patterns. Our result suggests the LLM has successfully leveraged certain physical priors acquired during its training process.

Animation Control Next, we showcase LLM's ability to perform animation control, which seeks to integrate existing animation clips into the game logic. For example, a player-controlled character should enter the jumping animation when the "space" key is pressed. In this work, we follow [10, 11, 19] to generate Unity C# scripts with GPT-4 [8], then execute the composed scripts with the Roslyn run-time compiler to enable animation state transitions [17]. Figure 2 shows an archer model downloaded from Mixamo [15] programmed to transition from being idle to walking per user prompt. We remark that the generated control flow is flexible and can be retargeted to any humanoid characters via Unity's Avatar system [18].

3 Discussion

We encourage the reader to inspect videos for the generated animations on our Github repository. In addition, we provide more details on the results shown in Figures 1 and 2 in the supplementary material.

To the best of our knowledge, our approach marks the pioneering method for actuating 3D models based on natural language descriptions. By embedding our framework in Unity, its output keyframes can be readily adjusted in the Unity editor window, allowing human animators to enhance and refine the results as needed. Hence, a potential application of our model is to produce initial drafts for digital artists similar to frameworks such as Midjourney and Stable Diffusion[16, 12].

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li. Nerf: Neural radiance field in 3d vision, a comprehensive review. *arXiv preprint arXiv:2210.00379*, 2022.

[3] L. Höllein, A. Cao, A. Owens, J. Johnson, and M. Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. *arXiv preprint arXiv:2303.11989*, 2023.

[4] Y. Hong, H. Zhen, P. Chen, S. Zheng, Y. Du, Z. Chen, and C. Gan. 3d-llm: Injecting the 3d world into large language models. *arXiv preprint arXiv:2307.12981*, 2023.

[5] H. Jun and A. Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[6] G. Li, H. Zheng, C. Wang, C. Li, C. Zheng, and D. Tao. 3ddesigner: Towards photorealistic 3d object generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2211.14108*, 2022.

[7] C.-H. Lin, J. Gao, L. Tang, T. Takikawa, X. Zeng, X. Huang, K. Kreis, S. Fidler, M.-Y. Liu, and T.-Y. Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023.

[8] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[9] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[10] J. Roberts, A. Banburski-Fahey, and J. Lanier. Steps towards prompt-based creation of virtual worlds. *arXiv preprint arXiv:2211.05875*, 2022.

[11] J. Roberts, A. Banburski-Fahey, and J. Lanier. Surreal vr pong: Llm approach to game design. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, December 2022.

[12] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[13] U. Singer, S. Sheynin, A. Polyak, O. Ashual, I. Makarov, F. Kokkinos, N. Goyal, A. Vedaldi, D. Parikh, J. Johnson, et al. Text-to-4d dynamic scene generation. *arXiv preprint arXiv:2301.11280*, 2023.

[14] M. Sra, S. Garrido-Jurado, and P. Maes. Oasis: Procedurally generated social virtual spaces from 3d scanned real spaces. *IEEE transactions on visualization and computer graphics*, 24(12):3174–3187, 2017.

[15] Adobe. Mixamo: Animate 3d characters for games, film, and more. *Adobe*, 2008.

[16] Midjourney. Midjourney. *Midjourney*, 2022.

[17] Trivial Interactive. Roslyn c# - runtime c# compiler. *Unity Asset Store*, 2019.

[18] Unity Technologies. Unity game engine. *Unity*, 2005.

[19] F. D. L. Torre, C. M. Fang, H. Huang, A. Banburski-Fahey, J. A. Fernandez, and J. Lanier. Llmr: Real-time prompting of interactive worlds using large language models, 2023.