A·P·I·I·T

ASIA PACIFIC INSTITUTE
OF INFORMATION TECHNOLOGY

INDIVIDUAL ASSIGNMENT

RESEARCH METHODS FOR COMPUTING AND TECHNOLOGY

Student Name	Cheng Yi Heng
TP Number	TP058994
Intake Code	APU2F2305CGD
Module Code	CT09832RMCT
Lecturer Name	Assoc. Prof. Ts. Dr. Tan Chin Ike
Hand Out Date	7th November 2023
Hand In Date	19th February 2024

Exploring Deep Learning Approaches for Real-Time Interactive Character Animation

Yi Heng Cheng

voxell.technologies@gmail.com

1. Literature Review

1.1. Introduction

This review is divided into several sections. The first section will discuss about the evolution of animation. The following section will be a deep dive into the research that has been done to create the technologies needed to make interactive character animation possible. Finally, we look into the recent advancements of deep learning that could possibly propel the interactive character animation standards.

1.2. Evolution of Animation

Animation in its simplest form is a sequence of actions that when played in a sequential manner, produces an illusion of movement. In the beginning, all animations are offline or pre-recorded in some form before displaying it to its audience. As oppose to many real-time animations we see today, especially in games, offline animations are not interactable, and thus only fits the purpose of the film industry.

Animation production started off with hand-drawn animations. From the 1940s to the 1980s, hand-drawn animations was the main mode of output in the animation industry (Lamotte, 2022). An animation is produced frame-by-frame, requiring prodigious quanitites of labor for the construction of a 24 frames per second film (Baecker, 1969). It was slow and ineffective, but it was the only choice given the state of the technology at that time.

The emergence of computer-assisted animation started gaining popularity during the 1970s (Lamotte, 2022). Computer graphics systems strive to create a better experience to replace the drawing and painting process, widely known as the "Ink and Paint" process at that time. TicTacToon was a method that proposes a paperless 2D animation production line (Fekete et al., 1995). Motion capture was also introduced using potentiometers to track the movement of the human body (Gleicher, 1999; Sturman, 1994) .

Starting around 2000s, purely computer generated images (CGI) has started to become possible. Computer graphics systems had evolved to be able to render 3D objects. A tremendous improvement in CGI can be seen from the film *Tron: Legacy* that was released in December 2010 and in production since 2009. There were also 3D released games with 3D interactive animations like *Halo: Combat Evolved*, *Gears of War*, and *Half-Life*.

1.3. Interactive Character Animation

Interactive character animation is made up of multiple underlying technologies. It is a subset of animation where characters are typically animated using a rig which deforms a mesh made up of triangles that is rendered onto the screen in real-time. The end goal is to create a system that is capable of providing visual feedback of character movements for users in real-time applications.

1.3.1. Mesh Skinning

Skinning is the process of performing mesh deformation according to a function of skeletal poses (Rumman & Fratarcangeli, 2016). In character animation, it is important to adopt a skinning method that is high in fidelity and performance. In this section, we will explore the various methods of skinning that had been developed over the years as well as their pros and cons.

Linear Blend Skinning (LBS) is a commonly used method in character animation where each vertex of the character mesh is influenced by a weighted sum of the transformations of nearby bones (Lander, 1998). It is being used in *AAA* game engines like Unity3D and Unreal Engine. LBS is known for its fast and simple algorithm that maps advantageously to the graphics hardware.

Spherical Blend Skinning (SBS) is another form of skinning method that employs spherical interpolation to smoothly blend between bone transformations (Kavan & Žára, 2005). SBS aims to solve the "lost of volume" artifact that LBS brings despite its efficient algorithm.

To solve the computational and memory overhead that SBS brings, Kavan et al. (2007) propose Dual Quaternion Blending (DQB). DQB uses dual quaternions to represent both translation and rotation, allowing for more accurate and natural deformations of the character mesh. Unlike SBS, it does not require additional memory to cache rotation centers. The DQB method is also extremely efficient. However, DBQ comes with a limitation, it only supports rigid transformation and is not suitable for scaling or shearing effects.

1.3.2. Inverse Kinematics

Inverse kinematics (IK) is widely used in video games and robotics to create realistic poses within a defined constraint. In short, the ultimate goal of IK is to determine an appropriate joint configuration that allow the end effectors to reach a target position (Aristidou et al., 2018).

One use case of IK is to perform animation retargeting to map movements between characters with different proportions (Molla et al., 2017). In the context of interactive applications like games, IK can also be used to perform secondary motions on top of an already playing animation (Ruuskanen, 2018). For example, turning the head towards an interest point, or moving the hand towards a target position.

At its current state, there is a total of 4 main categories towards IK:

1. **Analytical**

   Analytical IK solvers aim to determine all potential solutions based on mechanism lengths, initial posture, and rotation constraints. They often rely on assumptions to compute a single solution.

2. **Numerical**

   Numerical methods often require a set of iterations to achieve a satisfactory approximation by minimizing a predefined cost function.

3. **Data-Driven**

   Data-driven methods relies on large accurate animation databases. Most data-driven methods employs some kind of machine learning algorithms to learn from the dataset.

4. **Hybrid**

   The hybrid method is simply a way of combining 2 or more different IK methods into a single solution.

1.3.3. Physics Based Character Animation

Physics based character animation offers a completely new solution for developers to prioritize physics accuracy over animation precision. It forces characters to obey the laws of physics like preventing collisions between collidable objects and interacting with external forces such as gravity, pressure, etc (Ye Hu, 2016).

Authoring physics based character animation can be extremely hard. This is due to the unpredictability of the physical world. For example, a character might accidentally get hit by a physical object during runtime, resulting in unexpected movements or behaviors that can disrupt the intended animation sequence.

A major limitation of physics based animation is the inability to precisely control the artistic intent for achieving specific visual effect. Additionally, ensuring computational efficiency while simulating complex physical interactions adds another layer of challenge to the authoring process.

1.3.4. Animation System

Multiple animation clips are normally used in interactive environments to create a variety of dynamic motions. In a conference talk by Holden (2018), he mentioned that Assassin's Creed Origins had around 15,000 animations in the game. These animations are needed to be handled by an animation system to systematically select the correct animation clips to sample depending on the current scenario.

Game engines like Unity3D uses a hierarchical state machine (HSM) graph, shown in Figure 1. It controls the sampling of animation clips and the transition between them.

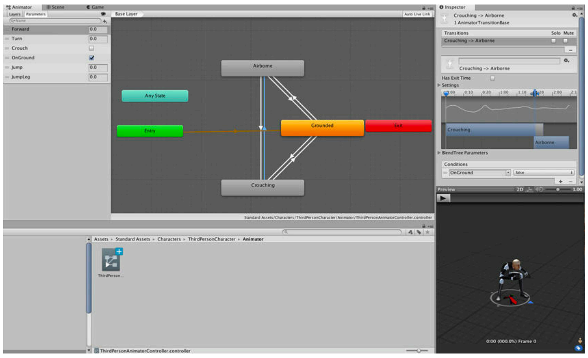

Figure 1: Unity's Mechanim

This allows developers to divide complex systems into smaller isolated modules (Berg, 2023). During runtime, the animation system will traverse the state machine graph and subsequently transition to the animation clip it reaches.

In some cases, animators would also like to mix and match different animation clips together. For example, an in between animation of walking and jogging to produce a slow jog. This can be achieved using a method called blend trees (Berg, 2023).

Another method known as motion blending is also used to apply motion trajectories onto the rig, based on a weighted sum of multiple animation clips (Ménardais et al., 2004).

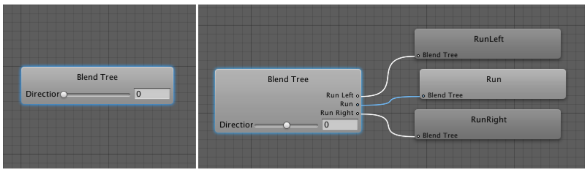

Figure 2: Blend tree.

This can create interesting motion dynamics like a walking animation clip towards the lower body part and a punching animation towards the upper body part.

1.4. Deep Learning in Animation

1.4.1. A Brief History of Deep Learning

1.4.2. Using Deep Learning to Drive Character Animation

The key idea of using neural networks is to attempt to generalize the problem and solve the scalability issue of many traditional animation systems.

2. Methodology

2.1. Target user

2.2. Sampling method

2.3. Data collection method

3. Conclusion

Real-time interactive character animation contributes largely into the immersion of interactive applications such as games. Constructing such system that can react realistically and naturally to dynamic environments is extremely challenging without incorporating machine learning components. Thanks to the adoption of neural networks and the abundance of accelerated computing of GPUs in recent years, the animation industry has been able to benefit from it by harnessing the enormous learning capability of neural networks to revolutionize the interactive application industry.

4. References

Aristidou, A., Lasenby, J., Chrysanthou, Y., & Shamir, A. (2018). Inverse kinematics techniques in computer graphics: A survey. *Computer Graphics Forum*, *6*, 35–58.

Baecker, R. M. (1969). Picture-driven animation. *Proceedings of the May 14-16, 1969, Spring Joint Computer Conference*, 273–288.

Berg, J. (2023). Animation Graph. *Lulea University of Technology*.

Büttner, M., & Clavet, S. (2015). Motion matching-the road to next gen animation. *Proc. Of Nucl. Ai*, *2015*, 2.

Clavet, S. (2016). Motion matching and the road to next-gen animation. *Proc. Of GDC*, *4*, 9.

Fekete, J.-D., Bizouarn, É., Cournarie, É., Galas, T., & Taillefer, F. (1995). TicTacToon: A paperless system for professional 2D animation. *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, 79–90.

Gleicher, M. (1999). Animation from observation: Motion capture and motion editing. *ACM SIGGRAPH Computer Graphics*, *4*, 51–54.

Harvey, F. G., Yurick, M., Nowrouzezahrai, D., & Pal, C. (2020). Robust motion in-betweening. *ACM Transactions on Graphics (TOG)*, *4*, 60–1.

Holden, D. (2018). Character control with neural networks and machine learning. *Proc. Of GDC 2018*, 2.

Holden, D., Kanoun, O., Perepichka, M., & Popa, T. (2020). Learned motion matching. *ACM Transactions on Graphics (TOG)*, *4*, 53–1.

Kavan, L., & Žára, J. (2005). Spherical blend skinning: a real-time deformation of articulated models. *Proceedings of the 2005 Symposium on Interactive 3d Graphics and Games*, 9–16.

Kavan, L., Collins, S., Žára, J., & O'Sullivan, C. (2007). Skinning with dual quaternions. *Proceedings of the 2007 Symposium on Interactive 3d Graphics and Games*, 39–46.

Lamotte, C. (2022). Discovering Animation Manuals: Their Place and Role in the History of Animation. *Animation*, *1*, 127–143.

Lander, J. (1998). Skin them bones: Game programming for the web generation. *Game Developer Magazine*, *1*, 10–18.

Lasseter, J. (1998). Principles of traditional animation applied to 3D computer animation. *Seminal Graphics: Pioneering Efforts That Shaped the Field*, 263–272.

Menolotto, M., Komaris, D.-S., Tedesco, S., O'Flynn, B., & Walsh, M. (2020). Motion capture technology in industrial applications: A systematic review. *Sensors*, *19*, 5687.

Molla, E., Debarba, H. G., & Boulic, R. (2017). Egocentric mapping of body surface constraints. *IEEE Transactions on Visualization and Computer Graphics*, *7*, 2089–2102.

Ménardais, S., Multon, F., Kulpa, R., & Arnaldi, B. (2004). Motion blending for real-time animation while accounting for the environment. *Proceedings Computer Graphics International, 2004.*, 156–159.

Peng, X. B., Abbeel, P., Levine, S., & Panne, M. Van de. (2018). Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, *4*, 1–14.

Peng, X. B., Guo, Y., Halper, L., Levine, S., & Fidler, S. (2022). Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions on Graphics (TOG)*, *4*, 1–17.

Rose III, C. F., Sloan, P.-P. J., & Cohen, M. F. (2001). Artist-directed inverse-kinematics using radial basis function interpolation. *Computer Graphics Forum*, *3*, 239–250.

Rumman, N. A., & Fratarcangeli, M. (2016). State of the art in skinning techniques for articulated deformable characters. *International Conference on Computer Graphics Theory and Applications*, 200–212.

Ruuskanen, A. (2018). *Inverse Kinematics in Game Character Animation*.

Starke, S., Mason, I., & Komura, T. (2022). DeepPhase: Periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics (TOG)*, *4*, 1–13.

Starke, S., Zhao, Y., Komura, T., & Zaman, K. (2020). Local motion phases for learning multi-contact character movements. *ACM Transactions on Graphics (TOG)*, *4*, 54–1.

Sturman, D. J. (1994). A brief history of motion capture for computer character animation. *Siggraph94, Course9*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

Ye Hu, L. (2016). Physics-based character animation. *Universitat PolitèCnica De Catalunya*.