CG_10_Character Animation Report

PB21010479 Caoliwen Wang

May 19, 2024

1 Problem Statement

In this task, we are implementing character animation. Although there is a lot of research in this area, our task is to achieve the simplest results. To simulate a character, we first need to study the changes in the skeleton and then address the skin. The skin is related to certain points on the skeleton and is handled using a method similar to interpolation. For the skeleton, we only need to consider the transformation of coordinates.

Skeleton animation is the foundation of character animation, primarily involving the transformation of the skeleton's coordinates. The skeleton typically consists of a series of joints connected by bones, forming a hierarchical structure. Each joint has a position and a rotation, and through these transformations, various character movements can be realized. We only need to focus on the transformations of the joints, specifically the changes in their positions and rotations over time.

Based on the changes in the skeleton, we then need to study the skin. Skin deformation applies the skeleton's animations to the character's mesh, allowing the character's appearance to change with the movement of the skeleton. Each point on the skin is influenced by certain points on the skeleton, and we use interpolation to handle the deformation of these points. Each skin point is affected by multiple joints, and the final deformation position is calculated based on weights.

By transforming the skeleton's coordinates and interpolating the skin points, we can achieve basic character animation. Although this method is simple, it can produce relatively natural character movements and provides a foundation for more complex animations in the future.

2 Propose Algorithms

2.1 Skeletal Animation

Skeleton animation consists of two main components:

1. Skeleton: A low-degree-of-freedom hinge system that acts as the driver.

2. Skinning: The surface mesh bound to the skeleton. The movement of the mesh vertices is influenced by multiple skeleton joints, and their final spatial coordinates are calculated through interpolation and other methods.

Due to its wide applicability and efficiency, skeletal animation is widely used in games and animations.

We can observe that the movement of parent joints will drive the movement of child joints, forming a tree-like structure similar to the diagram below:

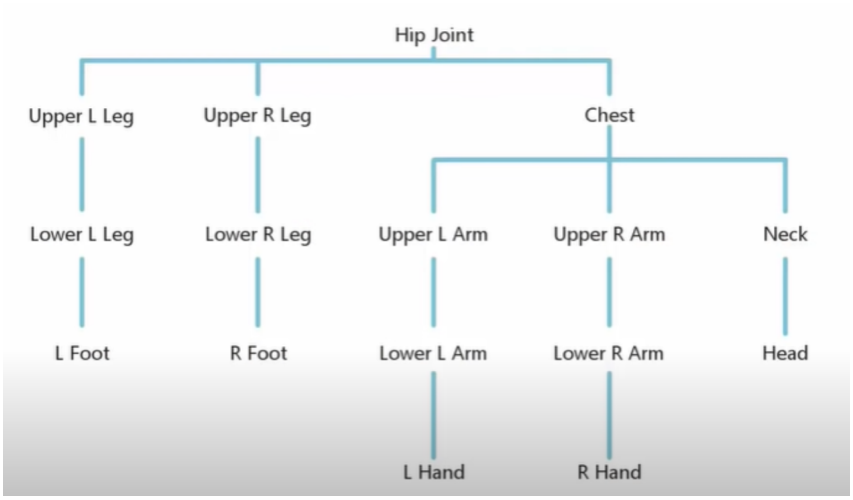

Figure 1: Illustration of Skeleton

Next, let's introduce some important transformation matrices:

1. **localTransform**: The coordinate transformation of the joint in the local coordinate system (4x4 matrix).

2. **worldTransform** (or global transform): The coordinate transformation of the joint in the world coordinate system (4x4 matrix). For the root joint, in our assignment, we can directly set its *worldTransform* = *localTransform*.

3. **bindTransform**: The position and orientation of the joint relative to the mesh vertices in the bind pose (e.g., T-pose). This is typically defined during the creation of the 3D model and remains constant throughout the animation's lifecycle. Its role will be seen in the skinning section below.

For a child joint, its *worldTransform* is the composite of the parent joint's *worldTransform* and the child joint's *localTransform*:

$$\text{worldTransform}_{\text{child}} = \text{worldTransform}_{\text{parent}} \times \text{localTransform}_{\text{child}}$$

This hierarchical transformation ensures that changes in a parent joint's position and orientation will correctly propagate to its child joints, maintaining the integrity of the skeletal structure during animation.

2.2 Skinning Animation

In skeletal animation, skinning refers to the process of deforming a character's mesh based on the transformations of its skeleton. This ensures that the surface mesh moves naturally with the underlying bone structure.

One of the most common techniques for skinning is Linear Blend Skinning (LBS), also known as smooth skinning. In LBS, each vertex of the mesh is influenced by multiple joints, and its final position is calculated based on the weighted transformations of these joints.

1. **Vertex Weights**: Each vertex v_i on the mesh is associated with one or more joints, with each association having a corresponding weight w_{ij}. These weights determine the influence each joint has on the vertex. The sum of weights for each vertex is typically normalized to 1.

2. **Vertex Transformation**: The final position v_i' of a vertex v_i in the animated mesh is calculated as the weighted sum of the transformed positions influenced by each joint. Mathematically, this is expressed as:

$$v_i' = \sum_j w_{ij} \cdot (\text{worldTransform}_{J_j} \cdot \text{bindTransform}_{J_j}^{-1} \cdot v_i)$$

   Here, $\text{bindTransform}_{J_j}$ is the bind pose transformation of joint J_j, and $\text{worldTransform}_{J_j}$ is the current world transformation of joint J_j.

3 Programming

3.1 Node programming

In this assignment, the node used in this instance is very simple. We adopted the connection method as shown in Figure 2.

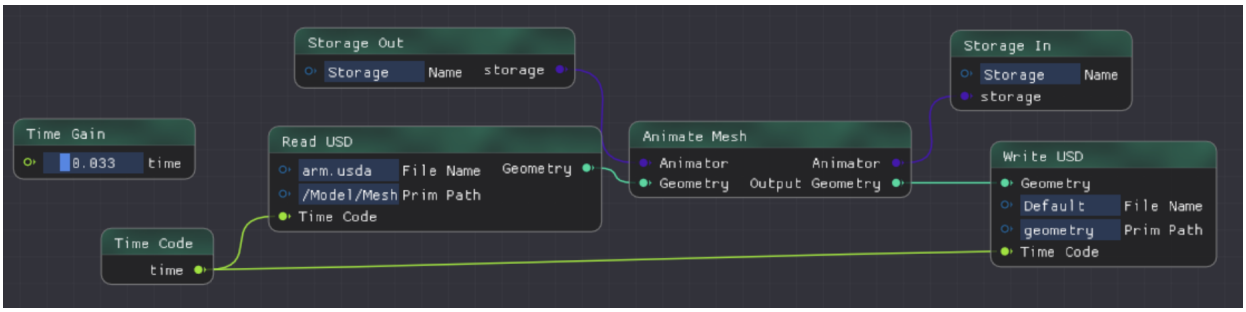

Figure 2: Geometry Nodes

4 Experimental Results

I think the results of this algorithm are fantastic.

We first tested the simple arm.usda file and obtained animation. You can find the video results in the ”outcomes” folder.

Then, we tested the animation of the complex belly_dance_girl.usda file. You can find the video results in the ”outcomes” folder.

We can observe that the animation effect of this animation is very fast, compared to all previous simulations.

I actually tried many other animations as well, but encountered some unresolved issues. Regarding the second link provided, which involves animations from miHoYo's Genshin Impact, PMX files were provided. These files can be opened using Blender, but after converting them to USD format and successfully importing them, the skeleton data cannot be detected. This issue has been perplexing me, and I have yet to find a solution. Additionally, in Blender, I can only see the character but not the animation, so I suspect there may be problems with the data conversion and handling.

So, unfortunately, in the end, I could only provide two essential animations in the outcomes, along with the data for hutao.usda.

5 Summary

In this task, we have implemented simple animation production, which is actually quite fascinating and closely related to today's topics such as gaming and anime. We can also observe that the effectiveness of this algorithm is good, as it only employs the most basic linear operations.