Assignments 1: Skeleton-based Animation

General Guidance

1. **Plagiarism is always strictly forbidden**.
2. These assignments are designed to help you to learn the algorithms and meanwhile practice your programming skills. **We may or may not provide reference resources and everything is up to your design**.
3. Various implementations are accepted as long as the results are satisfactory. You can use your favorite language or frameworks if it is not specified in assignment.
4. **It is always requested to explain your codes to our TA after submission.**
5. It is encouraged to utilize open source projects or third party libraries for **non-core** part (tagged with **[non-core]**) of your homework. **BUT you are requested to also explain the key part of how their codes work**. Usage of misunderstood or mysterious codes leads to a score deduction.
6. You need implement yourself for the **core** part (tagged with **[core]**)
7. Feel free to send emails to TA or post question on Piazza for help.
8. Good luck and have fun.

Introduction

In assignment 1, you are expected to write codes to implement a simple skeleton-based animation system from scratch which is capable to pose and animate a mesh with skeleton and virtual armature. For this assignment, we provide some reference C++ codes including a simple OpenGL framework, and a glTF loader. Two example glTF animated models is also provided to you for testing. You can either choose to use or not to use these reference resources and modify whatever you want. It is all up to your decision.

Reading Materials

The implementation would become easier if you understand the whole process before you start. There are some materials which might be helpful. Have a quick look before you start coding.

1. You can play with an interesting 2D skeleton-based animation demo here.
2. You can watch this video to see how people make a simple skeleton-based animation with modern software.

3. Here is a tutorial from learnopengl.com about the skeletal animation.
4. Here is a course from *ACM SIGGRAPH 2014* talking about skinning.
5. Here is another short tutorial from Andrea Venuta's blog.
6. If you are not familiar with OpenGL, this site would be helpful.
7. If you want to use glTF model we provide, the glTF specification would be the full reference.

Requirements

1. **[non-core]** Mesh Data Structure
2. **[core]** Skeleton System
    i. Data: hierarchy and local transformation
    ii. Algorithm: performing Forward Kinematics
3. **[core]** Skinning System
    i. Data: representation of a pose and the inverse bind matrix
    ii. Data: blending weights at vertices
    iii. Algorithm: calculating the mesh from current pose
    iv. Hint: basic Linear Blending Skinning (LBS) algorithm is acceptable
    v. Hint: you can either use predefined vertex weights in the model or generate a sophisticated weight for each vertex by its distance from bones
4. **[core]** Animation System
    i. Data: keyframe of poses
    ii. Algorithm: calculating pose at current time with interpolation, linear interpolation (Lerp) to the euler angle or quaternion are both acceptable
5. **[non-core]** Visualization System of Mesh and Skeleton

Checkpoints

1. **[Req. 1 and 5]** Visualize the mesh *[10 pts]*
    i. A realtime renderer with lighting is preferred but an offline wireframe third-party renderer is also acceptable
    ii. We have provided a wireframe mesh viewer in the reference codes and you can use it freely
2. **[Req. 2]** Visualize the skeleton of the rest pose and some other poses. *[20 pts]*
    i. Use lines to represent bones and show their hierarchy
    ii. You can visualize the mesh and skeleton separately or together with different color
3. **[Req. 3]** Visualize the blended mesh at the pose you made at Checkpoint 2. *[25 pts]*
4. **[Req. 4]** Animate the skeleton with interpolated poses and visualize it *[25 pts]*
    i. The rotation of the bones should be key-framed and interpolated as quaternion
5. Design the pipeline to combine your works together and generate a series of outputs with your skeletal animation system. *[20 pts]*

   i. You can achieve a full score if three different clips from the 'res/fox.glb' can be animated correctly

Reference Resources and Submission

This semester, we take advantage of the github classroom to send and collect assignments. You can access this link to accept the repository with the reference codes and example models we provide. **Please clone the repository down** to write your homework and **submit with `git commit` and `git push`.** If you are not familiar with git or github, please take a look at the github document.

If you decide to start from scratch and use little reference resources, you can put your work in the `submission` folder. Otherwise, you can modify and add codes in the `reference` folder and commit directly.

Submission Deadline: 2024-4-7 23:59

Late hand-in policy

Each student is allotted a total of five late-day points for the whole semester, which work as follows:

- A student can extend a programming assignment deadline by one day using one point.
- If a student does not have remaining late-day points, late hand-ins will deduce 10% of the total score per day.
- No assignments will be accepted more than five days after the deadline. This is true whether or not the student has late-day points remaining.
- We will strictly follow the rule above for late-hand-in policy unless you have a **VERY STRONG** reason, which should be explained to the course instructor and TAs.