A5 Project Proposal

Title: **Pokemon Fight! Pikachu VS Snorlax**

Name: Zhenyan Zhu
Student ID: 20817494
User ID: z277zhu

1 Purpose

My final project will be an interactive OpenGL 3D Pokemon battle game. You will control
the Pikachu to defeat the Boss, Snorlax. A typical Pokemon scene with natural grass will be rendered.
This game contains several of animations to simulate the battle and the movement of two Pokemon.
Particle systems emulating lightning and earthquake will also be added to embody the excitement of the battle.

2 Statement

2.1 What it's about

It's a 3D battle game, and it will concentrate on the animations, particle effects and damage judgement. In detail, the Snorlax(AI) will use Body Slam from the sky to the ground, which will cause earthquake effect, and after that, he will be stunned for 5 seconds; the user should control the Pikachu properly to escape the body slam and try to attack the Snorlax when he is stunned.

2.2 What to do

I will first refactor a framework from A3 that preserves the foundations used for this project, including hierarchy transformation, lua utils and so on. Then, I will gradually add features to this framework to build up this game. The main features to add are:

1. a sound system (third-party?) to provide convenient sound utils from loading .wav files
2. a cube environment mapping to map the environment to the background
3. movement animations and attack animations of Pikachu and Snorlax
4. a particle system to provide particle effects in battle
5. user interfaces for the user to control the Pikachu and attack the Snorlax
6. I will learn how to combine a geometry shader into the rendering pipeline to implement the wind effect of grass
7. shadow map and grass models with wind effect

2.3 Why it is interesting and challenging

This project is challenging to me because I am completely new to Computer Graphics and Game Development before this course, and everything in my project, including texture mapping, shadow map, key-frame animations are the things that I did not implement before, which may take me a lot of time to learn and put these features into the game. Besides, construct frames of animations will cost me a lot of time to make it more vivid and realistic.
However, this project will be very interesting to me because Pokemon is my favorite cartoon and video game while Pikachu and Snorlax are my favorite Pokemons. Given this oppurtunity, I can develop my own Pokemon game and make cool animations of these two Pokemons. I can make them do battle with each other and do whatever thing I want, so I will be very accomplished if I can complete this game. At the same time, I can also learn cool Computer Graphics knowledge.
In addition, this game will be very extensible because I can easily add more animations, particle, and even models upon this framework; so in the future, I may develop it as a complete and cool game if possible.

2.4 What I will learn

In this project, I will learn several Computer Graphics features and practice my software engineering skills

1. I will try to write as less code as I can; therefore, I will learn and utilize some design patterns to improve the code's maintainability.
2. I will learn how to do texture mapping for the environment as well as objects in the scene
3. I will learn how to cast shadows in OpenGL
4. I will learn how to implement animations by key-frame animations
5. I will learn how to develop realistic particle effects
6. I will learn how to implement realistic grass with wind effects
7. I will learn how to develop a 3D game

3 Technical Outline

3.1 Game Mechanism

Game mechanisms are the main logics of this game. The user will control Pikachu, and an AI will control the Snorlax. When the game started, the Snorlax will use body slam and try to smash Pikachu (the user), when the Snorlax hit the ground, there is a damage judgement to determine whether Pikachu is hurt. After that, the Snorlax will be stunned for 5 seconds and Pikachu can attack Snorlax. There should still be damage judgement to determine whether the electrons emitted by Pikachu hit Snorlax. After the Snorlax woke up, the AI can choose whether using "body slam" or "Meteorite Fall". Both characters will have an HP, when any of the characters' HP = 0, the game will end.

1. **Snorlax AI:** The AI of Snorlax will have two choice, it can either use "Body Slam" or "Meteorite Fall". The probability of choosing a choice is determining by the Snorlax's HP. The less its HP is, the more probability he will choose to use "Meteorite Fall".
   (a) When it uses "Body Slam", it will try to smash the Pikachu from the sky. To implement this, the Snorlax will continuously translate on its Y axis to show it moving(jumping) up to the sky. This can be implemented by transforming the Snorlax's y position in the appLogic(); then, it will obtain Pikachu's position from the program and translate to Pikachu's x and z position, then slamming down by continuously translating the Y axis. When the Snorlax's Y axis equals to the ground's Y axis, a camera shake will be performed and particle effects of flying dirt will be generated.
   (b) When it uses "Meteorite Fall", Meteorites (sphere or cube with textures) will be summoned randomly position through 5 seconds on the sky(a very large Y coordinate). Then, the meteorites will fall down to the ground with some random velocity. This can be implemented similar to "Body Slam", while in the appLogic(), all meteorites will have some rotation transformation to make them more realistic.
2. **Damage Judgement:** In our game, all the dangerous objects, including Pikachu, Snorlax and meteorites, will have their own positions relative to the World coordinate. Snorlax and meteorites will also have a damage radius for damage judgement. When these objects are transforming, their positions will be transformed respectively.
   (a) To determine whether "Body Slam" hit Pikachu or not. We can simplify it to a problem that determining whether a 2D point inside a circle or not because in our scene , the battle field is flat, so we can discard 'z' axis in damage judgement. Then we just use the equation: $(x-h)^2+(y-k)^2 < r^2$ where x,y are Snorlax's position and h,k are Pikachu's position.

(b) To determine whether "Meteorite Fall" hit Pikachu or not, for each meteorite hit the ground, we can just use method for "Body Slam" to determine the damage by substituting Snorlax's position to meteorite's position.

(c) To determine whether "Discharge" (Pikachu's attack) hit Snorlax or not. We can utilize the technique we learnt in A4 about determining the intersection between the ray and the sphere because "Discharge" is simply emitting a ray from Pikachu and the Snorlax's body is just a sphere.

3. **Camera Shake:** When meteorites hit the ground or the Snorlax slam the ground, there will be a camera shake happening to enhance the excitement of the game. To define a camera shake, we will first define a duration for the shake and preserve the original view matrix. Then at each frame, we draw a random vec3, and all its coordinate are from (-1, 1). After that, we normalize the vec3 and perform a translation of such vec3 to the view transformation matrix. After the shake, the view transformation matrix will be reassigned by its original view matrix.

3.2 Animations

1. Animations will be implemented by key-frame technique.

2. All animations will be pre-computed and stored in the form of {vector(KeyData), float timestamps}, and each KeyData is represented as {float timestamp, map(SceneNode*, mat4) }. In detail, an animation is the collection of KeyData(normally 3 or 4 KeyData) and the time length of this animation; a KeyData is a collection of transformations corresponding to SceneNodes at a specific timestamp.

3. When we need to perform an animation, in appLogic(), we will transform the object based on the interpolation of two KeyData. In detail, we will calculate the scaleFactor to multiply to the transformation matrix and then apply to the object while $scaleFactor = \frac{(currentTime - currentKey.timestamp)}{(nextKey.timestamp - currentKey.timestamp)}$

4. When the Pikachu is idle(user does not control it), it will stand up with 2 legs for 5 seconds and then sit down. If the user types "WASD" to control the Pikachu when it's idle, there will be animations illustrating it's transforming from a stand pose to a running pose with four legs. When the user release all the "WASD" keys, there will also be animations showing it's transforming from a running pose to a stand pose.

3.3 Particle Effects

There are two particle effects I want to implement. The first one is: when the Snorlax uses "Body Slam", it will jump to the sky and slam the ground; then some dirt(particles) will be shaken to the sky and fall onto the ground afterwards. Secondly, when Pikachu uses "Discharge", there will be electrons emitted from Pikachu as a beam to the target position.

1. Each particle has its own position, velocity, color and lifeTime. Each particle has the shape of a square, containing two triangles.

2. When updating the dirt particle effects, the gravity should be considered to make the particles more natural. While the color of electrons particles should be updated to make the effect more like a lightning.

3.4 Shadow Map

1. The point lights in A3 will be modified to directional light for implementing shadow mapping.

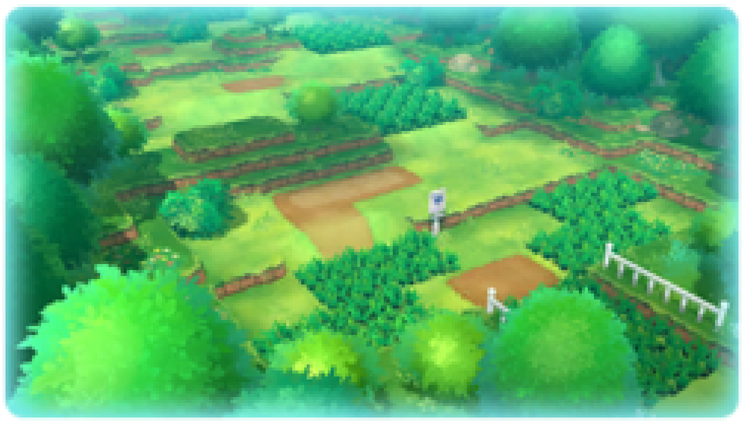
2. A framebuffer per light will be created to generate depth map for each light. The depth map is rendered from the light's perspective for testing shadows. (No fragment shaders are required for the passes of generating depth map)

3. The main fragment shader will take the shadow map as input, and it will also have a 'shadow' value that is either 1 when the fragment is in shadow or 0 when not in shadow.

3.5 Toon Shader

1. Toon shading's basic effect is to have large areas of constant color with sharp transitions . To do this, while computing the cosine angle between the light direction and the normal, we can directly quantize the cosine term. For instance, if $0.75 < cos(\theta) < 1$, we map it to 0.75; if $0.5 < cos(\theta) < 0.75$, we map it to 0.5.

3.6 Modelling & Scene

1. I will add another Lua method called add_child_deepcopy and corresponding copy constructor in SceneNode/GeometryNode/JointNode to achieve deepcopy a node with its related Geometric fields and Joints. This will be very convenient for me to construct models with repeated hierarchy structures.
2. The Pikachu puppet will be upgraded to support movement with four legs, standing up with two legs and attacking.
3. The Snorlax will have 6 degree of freedom to support jumping up and body slam.
4. The scene will be rendered similar to Kanto Route 1(a Scene in Pokemon's world), there will be grasses, fences, grassland and muddy road.
5. 

3.7 Realistic Grass with Wind Effect

1. We will represent one face if the grass by a Quad with grass texture mapped onto. One grass instance will have three faces which we will arrange crosswise
2. To display a more realistic rendering, we will randomly rotate each grass into a random value on the Y axis and give it a random size
3. We will take a flow texture that has both red and green values to encode the direction of the wind, and the blue part encodes the force of the wind. We will also construct a geometry shader to calculate the wind rotation matrix and apply that only to the top corners of the grass Quads such that the base of grass will not move and it will give a natural feel to wind.

3.8 Sound System

1. The third-party library, irrKlang, is used to implement our sound system. It will be maintained in a Singleton and initialized in the start of the game and play the default background music. When the Snorlax has HP lower than one third, another more exciting music will be played.
2. All characters' main motion, such as attack and running, will play a 3D sound based on the their 3D position. The mapping from method name to sound file will be configurable and loaded when the sound engine is initialized.

3.9 Environment Mapping and Texture Mapping

1. The .obj loader will be upgraded to support UV coordinates when loading vertices.
2. Then, a texture loading and cacheing system will be implemented. The image files will be loaded routines provided by lodepng.h and preserved in a Singleton.
3. I will use Cube Mapping to implement Environment Mapping. In detail, I will load 6 pictures by to represent 6 faces of the environment.
4. Then, I will bind the ith decoded image to GL_TEXTURE_CUBE_MAP_POSITIVE_X+i by glTexImage2D routine and setup parameters by glTexParameteri().

4 Bibliography

Learn OpenGL - Graphics Programming, de Vries, Joey, 2020.
I use the web version at `https://learnopengl.com/` here are some specific articles:

1. `https://learnopengl.com/Guest-Articles/2020/Skeletal-Animation` for key-frame animation
2. `https://learnopengl.com/Advanced-OpenGL/Cubemaps` for cube maps and texture mapping
3. `https://learnopengl.com/In-Practice/2D-Game/Particles` for particle systems
4. `https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping` for shadow mapping
5. `https://learnopengl.com/In-Practice/2D-Game/Audio` for 3D audio

GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics, Hubert Nguyen, 2007.
I use the web version at `https://developer.nvidia.com/gpugems/gpugems/contributors`, and I learnt netural grass effect under the article `https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-7-rendering-countless-blades-waving-grass`

Objectives:

Full UserID: z277zhu **Student ID:20817494**

___ 1: There is an environment mapping utilizing cubeMap to render a Pokemon skybox and a texture system to load and cache textures.

___ 2: There is a 3D sound system that can both generate the background music and provide sound effect in the battle

___ 3: There is a series of different animations utilizing transformation when the Pikachu and Snorlax are running, standing and attacking;

___ 4: There are particles simulating lightning and flying dirt when the two Pokemon attack

___ 5: Toon shader is added, and there will be a button to switching from Toon shader or Phong shader.

___ 6: Shadow mapping is added, and there will be a button to turn on/off the shadow mapping.

___ 7: An artistic Pokemon scene will be rendered

___ 8: realistic grass with wind effect is modeled by texture mapping and geometry shader

___ 9: Game mechanisms including damage judgement, Snorlax AI and camera shake are well-established and robust

___ 10: The Pikachu model in A3 will be upgraded and a Snorlax model will be created to support different motions.