Interactive Graphics final project

Artur Back de Luca backdeluca.1900870@studenti.uniroma1.it

Abstract

This report describes the implementation details of the final project of the Interactive Graphics course at the Sapienza University of Rome. In addition to the technologies employed for the said task, we disclose the project's objective and structure, as well as some of the design challenges and solutions along with the implementation.

1. Task description

The theme of the project was the decision of the students, as long if fit a given set of requirements:

- Use at least one hierarchical model
- Use lights and textures
- Enable some kind of user interactions
- Implement custom animations exploiting the hierarchical structure of the models.

This project may be implemented using standard WebGL or more advanced libraries, such as ThreeJS or Babylon, if approved.

2. Proposed solution

My solution consisted of implementing a version in the well-known genre of games, known as the *Endless running*. In this type of game, the user controls the position of a moving character so to avoid incoming obstacles. This way, the character continues moving, and thus increasing the score. The game generally only stops when the character collides with one of the said obstacles, thus explaining the *endless* nature of its name. This genre received wide acclaim over the past decade, especially in the mobile ecosystem. Some of its most successful franchises are: *Subway surfers*, *Jetpack joyride* and *Temple Run*.

Figure 1: Subway surfers (*left*), Jetpack joyride (*middle*), and Temple Run (*right*)

My variation of this game is called *Snowman slide* and, in the same spirit, consists of a snowman sliding down a snow-covered mountain and avoiding obstacles.

Figure 2: Front cover and game-play of Snowman slide

As is the aforementioned example, the Snowman slide runs continuously, only stopping once the main character, also here referred to as hero, hits one of the obstacles. Furthermore, to avoid user disengagement throughout the game-play, the game's speed improves over time, so to keep the game entertaining and challenging.

3. Key features and design choices

In the following subsections, we describe some of the worthy details of the implementation of Snowman slide.

Frameworks

The entire application was solely built around ThreeJS in its release `r120`. As we later see in detail, all the transitions and physical properties were completely accomplished by JavaScript, without relying on external libraries, such as TweenJS or Physijs.

World

Unlike the mentioned examples, this game takes place in a steep hill, so, to achieve this elevation effect, the implementation revolved around a curved surface. Essentially, the camera and the hero remained still along the z-axis, while the sphere beneath them rotated, thus giving the effect of running down a hill. Therefore, the speed of the game is controlled by the angular speed along the x-axis. To prevent game repetition or excessive computation, the obstacles appear and disappear based on their location on the screen. Once these disappear, they are fed a pool to be repurposed later on during the game.

Figure 3: Overview of the game scenario

Furthermore, a fog effect and an evolving sky-box complete the scenario of the game. The cube has roughly the same dimensions as the ground sphere and can show different backgrounds depending on the provided path. For deployment, however, this option is set fixed, but it is easily adjustable by using the `createPathStrings` function in the `src/world.js` file.

Main character

The only interactive model used in this application is the hero, a snowman whose objective is to reach the end of this endless descend. It consists of a ready-made model that was modified for this specific task. It is a hierarchical model with a total of 11 sub-elements.

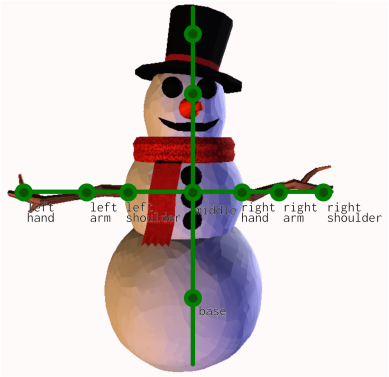


Figure 4: Structure of the hero

Along the y-axis of the model, all parts are recursively dependent, starting by the base of the model. Also, the arms roughly follow the same structure but are anchored in the middle node of the torso. Some details of the model, such as the scarf and the face components were kept fixed in their respective locations since these would not be used. On the other hand, some parts of the given model, extracted from Sketchfab, were generally too rough for the overall aesthetic of the game, so these had to be smoothed out. Additionally, since each element of the model was detached from one another, the devised solution to build

the hierarchical structure based on a recursion method, that not only linked parents and children but also applied the necessary modifications to ensure the correct location of each part of the hero's body.

```
var body =  {
    value: 'base',
    settings: [{name: 'position.y', value: 1.8}, ...], // Settings of base
    children: [
      {value: 'middle',
       children: [
        {value: 'left_shoulder',
         settings: [...],
          children: [
            {value: 'left_arm',
             children: [
              {value: 'left_hand'}
             ]
            }
          ]
        },
        {value: 'right_shoulder', // Sibling of left_shoulder
          children: [...] // Same process goes for right arm
        },
        {value: 'head', children: [
          {value: 'hat'}
        ]},
      ]},
    ]
}
```

Figure 5: Body specifications fed to the recursive building function

To make the animation process much easier, the building method also provided a template of the standard position and rotation of each part of the model. This came to be helpful when animating so as to avoid repetition when specifying the desired target poses. When a particular element of the full pose was not provided, the standard values were used instead.

Obstacles

The obstacles used in this game have to major variations: rocks and trees. The trees are further divided into two randomly chosen types: a normal and a snow-covered pine tree. These models are ready-made but suffered a small modification, having a wooden texture at the trunk. The rock used was custom-built, having texture by itself simulating a rough surface.

As previously mentioned, these models are only rendered in a small portion of the world sphere. To ensure variability in the game-play, these have minor random modifications, either in their size or orientations. The models are rendered at a regular interval and can also have slight modifications with respect to the number of obstacles per interval, all of this being dictated by random values drawn every update iteration.

Animations

Besides the sphere rotation previously mentioned, the game also counts with character and world animations. The two other off-character animations involve snow: one simulating snowfall and the other simulating a snow cloud when the hero hits an obstacle. These are achieved by moving small particles. In the case of the falling snow, they position in the y-axis is decreased by a gravity factor iteration. Once it reaches values below zero, they are reset to the top of the sky. Additionally, those particles that make the snow-cloud are only added to the scene when the character collides. When this happens, these are then quickly dispersed simulating an explosion effect.

Figure 6: Hero poses and animations

The mechanics of the game allow for three types of movement, two sideways (left and right) and one upward (jump). For all these cases there's a specific animation. To achieve the desired effect, when the user issues a certain game command, it chains the sequence of poses to be interpolated through a certain number of frame updates. Each frame update fills a small gap between the current and the target pose. When these are through, the only animation that remains is the standard vertical flicker of the contact between the hero and the surface, produced by the addition and subtraction of random noise and a damping factor here called gravity.

4. Considerations

Taking into account the technical details previously described, the author believes that the project meets the agreed requirements. The overall game experience is satisfying, enables mobile interaction and it was well-received among friends and fellow colleagues.

Despite the adversities found along the implementation, the project has been a great exercise of autonomy and analytical thinking as well as an opportunity to revisit and put to practice some of the concepts learned during the course.

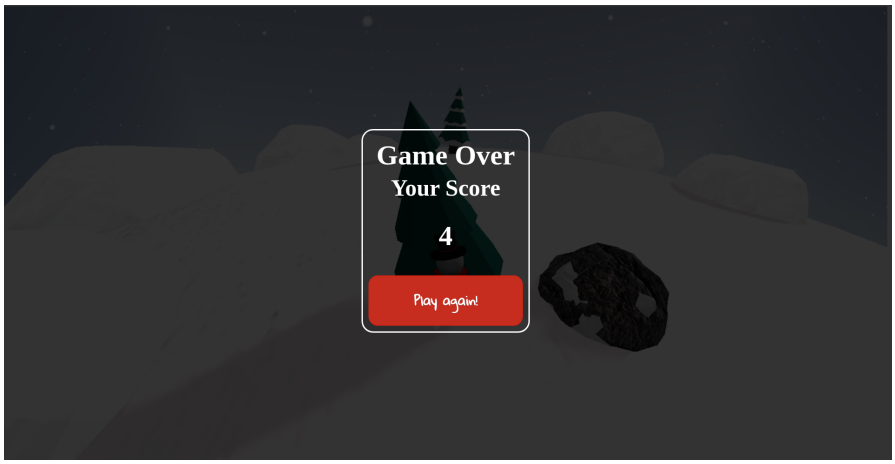
5. Appendix

User manual

1. Begin by pressing start in the initial page

2. Move the character by pressing ← (left), ↑ (up) and → (right) keys to avoid incoming obstacles

3. If you an obstacle, you lose the game. If you wish to play again, click the button in the game overlay.



References of models used throughout the game

Hero: https://bit.ly/2RYJG3U (Sketchfab)
Pine tree: https://bit.ly/3n03geC (Google Poly)
Snowy pine tree: https://bit.ly/3kIok7i (Google Poly)

All the other models not here mentioned were custom made.