Interactive Graphics - Homework 2

Engineering on Computer Science

Daniele Cantisani 1707633

June 2020

1 Create the grizzly and the tree

The bear construction was done by modifying all the components that were on the default file following a hierarchical structure.
The grizzly is composed by a Torso (father of the figure) that has 6 children as **head, leftupperarm, rightupperarm, leftupperleg, rightupperleg and lastly the tail**. All the legs(or arms) have a son representing the lower part of the leg(arm). These are **leftlowerarm, rightlowerarm, leftlowerleg, rightlowerleg**. Although a bear has no distinction like a man in terms of arms and legs, the names have been left so for ease of code recognition.
Each component is a cube, which is shaped in height, width and depth to vaguely resemble the figure of a grizzly. From this idea comes the choice to have a very large torso like that of an adult bear from which the other components branch out.
Each component has a recognition index. To give the initial position to all components I modified the theta vector. Each index of the vector represented the angle of rotation of the i-th component with which it was associated. Then a cycle was performed on all the components to initialize the nodes (initNodes(index$_i$)): here a switch()−case recognizes all the components and modifies the position m (ModelViewMatrix) through two operations that are performed for each component, such as translate(x,y,z) and rotate(theta[index$_i$]).
To avoid compromising the m-position matrix (ModelViewMatrix) we use the traverse function. With this function we can save the m-matrix instance before using it to initialize the child values (and then it is completely modified) and then retrieve it later when we exit the recursive node of the child component; in this way we can use it again for a possible sibling of the initial component. This is possible thanks to the function **push(modelViewMatrix) and pop(modelViewMatrix)**. It is the same traverse function that calls the final function that builds the figure as we imagine it and as we have positioned and rotated it. In fact each component has its own function where we scale the object and through the TRIANGLE_FAN method we build it.
While the functions and components for the bear parts were already present in the default file, the ones for building the tree were not there. Following the same scheme I built a trunk and tree canopy using the cube function. The two components are siblings and do not depend on each other.

2 Configure the textures

As for the texture inserted for the components, I followed similar steps as for the first homework. The difference is that I decide to use internal textures, created by code from scratch without having an external image to take inspiration from. Some textures were created following the code of the book *Interactive Computer Graphics 7E* by Angel and Schreiner.
The required goal was to create two different textures for the body and the head of

the grizzly: through a flag sent to the fragment shader it was possible to recognize the different cases and differentiate the two textures. After recognizing the component by flag, proceed by applying the correct texture.
Example with the texture of the head:

$$fColor = vec4(1.0, 1.0, 1.0, 1.0) * (texture(Tex1, fTexCoord));$$

The same reasoning was carried out for two other textures that color the trunk and the tree canopy with two purposely created textures.

3 Animation

A button has been inserted to start the full animation (a flag is activated in the js file). During the execution it is possible to press the same button to stop it. At the end of the animation you can restart it by pressing on "Restart".
The idea of animation of the figure is handled by repeatedly calling the animation() function. At each execution, variables are modified that will later be the parameters of translate() and rotate() functions of a given component. In fact at each animation() execution, the figure of the grizzly will be recreated from scratch through the initnodes() and point(1) passages.
Examples:

$$m = mult(m, translate(translatebodyX, 0, 0));$$

$$m = mult(m, rotate(rotatetorso, vec3(1, 0, 0)));$$

The small changes that are made by the computer at very high speed will appear as a fluid movement on the screen.
I divided this whole part into three subsections: **1)** the grizzly walks towards the tree **2)** the grizzly stands up and turns its back to the trunk **3)** the grizzly rubs its back against the trunk.
Note: all the movements of the grizzly, apparently not inherent to the target required by the homework are added in order to make the passages more realistic.

3.1 The Walk

The main movement of the walk consists of two main parts: the translation of the whole body and the backward and forward movement of the legs. The speed of movement was chosen as a very low value. In fact, a grizzly has a slow pace when walking unless he's attaccking.
As for the movement of the legs, the alternation of movement during the walk is that of movement of the right side (right arm and right leg) and movement of the left side (left arm and left leg) forwards and backwards respectively. Through flag I rotated on the x-axis the four parts of the upper leg. The movement was also transferred to the four lower leg parts being child components. On these have been added another rotation to make the movement more realistic: in fact the lower parts of the leg rotate in the opposite direction of the upper one to give the feeling of a bend in the leg as it really happens when you walk.
Finally, a few seconds after the start of the walk, the head starts to move in the direction of the tree. This is an additional movement to represent a sort of recognition by the animal of the object on its left (the tree).

3.2 Get into Position

Once the figure sits next to the tree, the goal is to get the bear up and rotate it so that its back rests on the tree.
At first the grizzly loads the thrust on his legs for a few moments. Then the grizzly

stands upright (the axis of rotation is the z axis with respect to our perspective). While standing up there is another very important rotation: the legs must remain on the ground being the only remaining support of the grizzly, which is why while the body rotates in one direction the two legs rotate in the opposite direction, to stay attached to the ground and give the feeling of upward thrust.
After getting up, the bear must rotate around the y-axis 90 degrees to give the back to the tree. Since grizzlies have a good balance even when standing, although for a short time, I thought I'd let him fill the little space left to lean against the tree, walking backwards for a few steps. So to make this shift realistic, I moved the legs in the same way as before like he's walking. The body rotates at the same time and is shifted slightly.

3.3 Scratching the back

After reaching the ideal position, the grizzly starts rubbing against the tree. As inspiration for this movement, I took my cue from a video on YouTube of the Orphaned WildLife Center.
I decided to move the bear alternatively up and down, moving the legs accordingly, simulating an unloading and loading of the weight on the legs and the bending of the knees of the back legs.
First I added a movement that translate the body of the grizzly alternately up and down. Then to give the idea of leg flexing, it was necessary to implement an up and down leg movement alternating with the body movement. In fact, when the figure goes up, the legs go down and are longer (the bear is standing at the highest point) while when the figure goes down, the legs go up (the bear is bending on his knees).
At the same time a new rotation at the lower legs alternately makes it appear that the knees bend forward and backward.
At the same time to make the scene more realistic, the head moves right and left trying to simulate the video on YouTube. Another movement added was the movement of the front arms. Before he starts scratching his back against the tree, the grizzly put the two parts of the arms parallel to the body. This movement was also noticed and added after watching a real video of a bear scratching his back against a tree.

To observe the scene from all possible perspectives, a slider has been added to change the theta angle of the perspective. It was added also a button that changes the perspective from the initial theta to another predefined one.