Interactive Graphics

Homework2

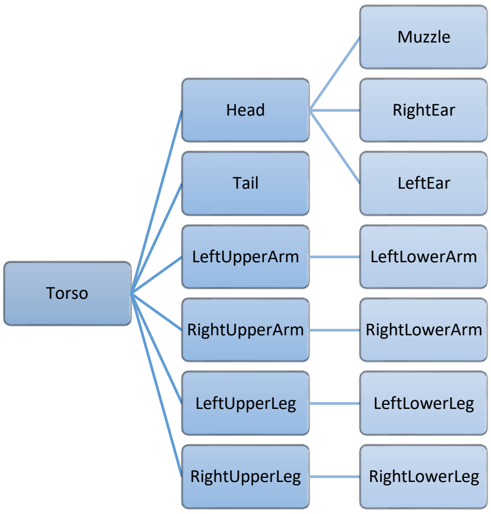
Valentina Sisti

1. Creation of the hierarchical model of a simplified Grizzly bear and the model of a simplified tree

The first request that was asked to implement in the second Homework, was to create a hierarchical model of a simplified Grizzly bear. To achieve such result, I started from the hierarchical model that was already implemented in the template code that we were given to start with. Starting from that, in order to make the model seems like a Grizzly bear, first of all I brought the legs and the arms closer to the torso, by changing the values in the respective translate functions and then I also modified their width and height, in order to have the proportions between all the parts of the body quite similar to that of a Grizzly bear.

Then, it was also asked to add the tail for the bear, that I defined in the same way in which all the other parts that compose the bear are defined, and I placed it in the right position by properly using the translate function. Moreover, to make the representation of the bear more realistic, I also added other details to the body, like the ears and the muzzle.

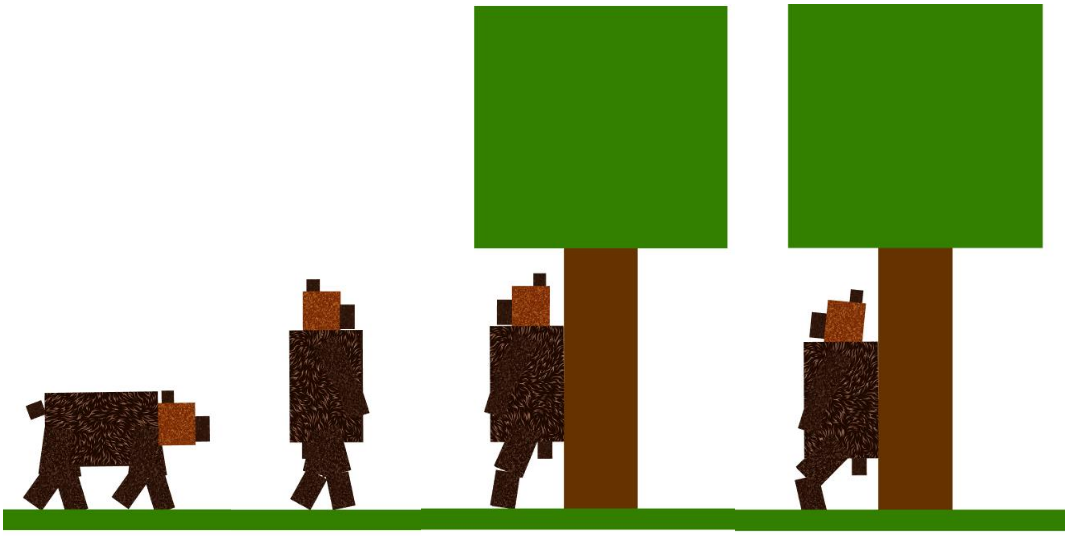
The resulting hierarchical model of the grizzly bear is the following:



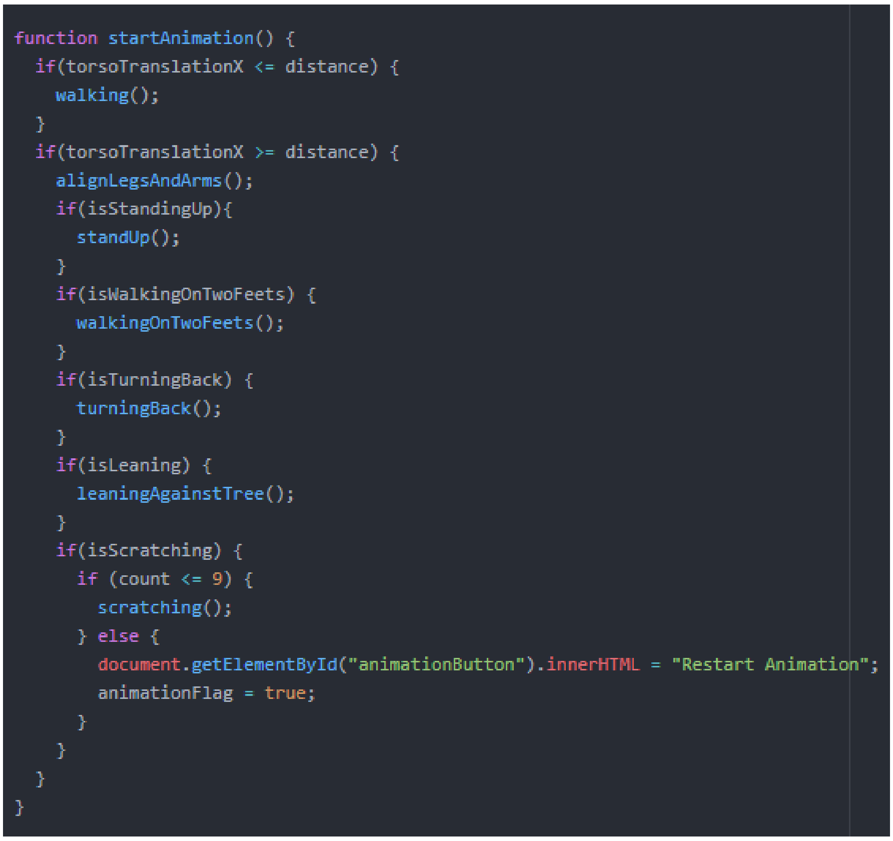
To create the model of the tree, I defined the trunk of the tree and the crown of the tree, I declared their respective drawing functions and their joint angles in the theta vector, and then added them in the hierarchy, following the same approach and implementation used to draw all the other elements in the hierarchy. To make the scene more realistic, I also decided to draw the ground, so that, for the entire motion, it would not seem that both the tree and bear was floating in the air. To have a more satisfying result, I also took care of assign a proper color to the tree and the ground, in particular by assigning a brown color to the trunk of the tree and a green color to the ground and to the crown of the tree.

2. Animation

Another and probably the most important feature that was requested to be implemented in the second homework was to animate the bear in order to make it walk towards the tree, then make it stand up and finally scratch its back against the tree.

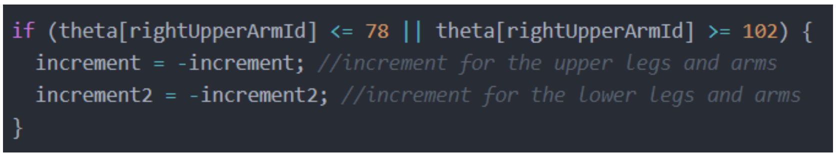


To animate the scene, I subdivided the entire motion into different parts. Here's how the function that handles the entire animation looks like:



```
function startAnimation() {
  if(torsoTranslationX <= distance) {
    walking();
  }
  if(torsoTranslationX >= distance) {
    alignLegsAndArms();
    if(isStandingUp){
      standUp();
    }
    if(isWalkingOnTwoFeets) {
      walkingOnTwoFeets();
    }
    if(isTurningBack) {
      turningBack();
    }
    if(isLeaning) {
      leaningAgainstTree();
    }
    if(isScratching) {
      if (count <= 9) {
        scratching();
      } else {
        document.getElementById("animationButton").innerHTML = "Restart Animation";
        animationFlag = true;
      }
    }
  }
}
```

The first motion, as requested, is the walking mode (on four legs) of the bear. In order to do that, I made the right leg and arm and the left leg and arm alternatively move back and forth by incrementing/decrementing each time their respective joint angles in the theta vector by a certain value, and within a certain range: in fact, let's suppose that at some point the right leg and arm are moving forward and the left leg and arm are moving backward: then, when they will reach, respectively, the upper and lower limit in the range, the motion of the arms and the legs needs to be inverted, and it was possible to do so, with such implementation, simply by inverting the sign of the value used to increment the angles of all the limbs of the bear, that will be positive or negative depending on the fact that the angles of the legs and arms are being incremented or decremented to simulate the walk.



```
if (theta[rightUpperArmId] <= 78 || theta[rightUpperArmId] >= 102) {
  increment = -increment; //increment for the upper legs and arms
  increment2 = -increment2; //increment for the lower legs and arms
}
```

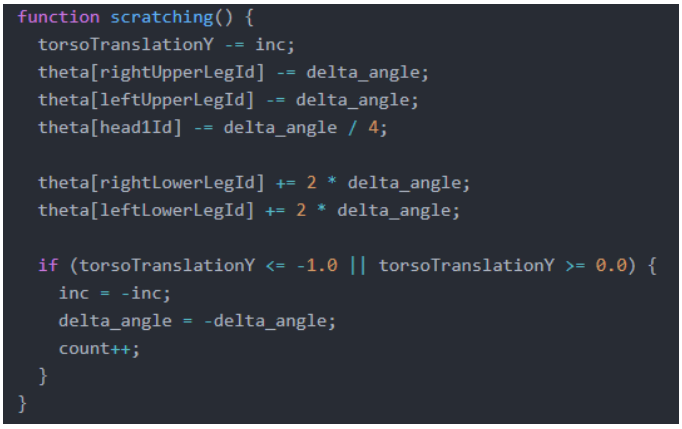
When the bear reaches the tree, he needs to stand up, but before doing that, since the legs and the arms would possibly not be in a straight position, before calling the standup() function, it will be called the alignLegsAndArms() function, that, as the name says, takes care of aligning the legs and the arms to a straight position, as when the bear is simply standing on four legs, without having the legs and the arms rotated as when the walking motion stopped because the bear has reached the defined distance with the translation of the torso.

Once that the legs and the arms have been realigned, then it will be called the standUp function, which makes the bear standing up by modifying the angle of rotation of the torso. An important point to which I had to pay attention to during the standing up motion was that of keeping the legs (in particular the upper legs) fixed with respect to the rotation of the torso, and has been possible to do so by incrementing their angles in the theta vector by the same quantity as that of which the angle of the torso has been decremented, otherwise they would have rotated together with the torso.

Since when the bear has stood up he needs to get closer again to the tree in order to perform the scratching, to make it move closer enough to the tree I had to implement also a second walking motion, but this time performed only on two legs, by using the same approach that I used with the walking motion of the legs and the arms, but this time applied only to the legs and not to the arms.

After the walking on two legs motion, the bear will finally be closer enough to the tree to perform the scratching of its back. But, in order to do that, he first needs to turn back, and lean its back against the tree, so for this purpose I defined two functions, one to make the bear turning back (by performing a rotation of the torso following the same approach that I used in the previous functions) and one to make the bear leaning its back against the tree and inclining the legs in a position from which the scratching motion will start from, and this is done by moving the body of the bear using one of the legs as a pivot for this specific movement.

Once that the bear is leaned against the tree, the scratching motion starts and it is performed by making the torso moving up and down and the upper and lower legs rotating in opposite directions, in order to simulate a natural movement with the legs that bends when the bear gets up and down to scratch its back. Here's how the function for the scratching movement is defined:



```
function scratching() {
  torsoTranslationY -= inc;
  theta[rightUpperLegId] -= delta_angle;
  theta[leftUpperLegId] -= delta_angle;
  theta[head1Id] -= delta_angle / 4;

  theta[rightLowerLegId] += 2 * delta_angle;
  theta[leftLowerLegId] += 2 * delta_angle;

  if (torsoTranslationY <= -1.0 || torsoTranslationY >= 0.0) {
    inc = -inc;
    delta_angle = -delta_angle;
    count++;
  }
}
```

The approach used to invert the motion from up to down and down to up for the torso is exactly the same that I used to handle the walking motion, that is when the torsoTranslationY reaches either the upper bound or the lower bound in the range, then the motion of the torso, the upper legs and the lower legs will be inverted.

3. Button to start the animation

As requested, I also added a button that has the purpose of starting the animation and it’s responsive with respect to the status of the animation:

At the beginning, when the html page is loaded, the animation is not activated, so, by pressing the button, the animation will start.

When the animation is already started, at a second press on the button, the animation will stop and will resume if the button will be pressed once again. This can be done at any time for the entire duration of the animation.

At the end of the animation, the press on the button will make the animation start again from the beginning.

4. Textures

Another request that had to be implemented, was the application of a texture to the entire body of the bear, with the condition that the head had a separate texture. The texture that I chose for the body is an image with the representation of a deep brown hair in cartoon style, in line with the simple representation of the whole scene. For the head, in the other hand, the image that I chose is once again that of a bear hair (to not make the head look too different from the rest of the body) but this time of a lighter brown, but in the same cartoon style of the texture used for the body. To decide what texture to apply and when to apply it, I used the gl.activeTexture() function, that permits to specify what texture unit to make active.