Howework 2

Interactive Graphics

Ibis Prevedello

June 2018

1 Introduction

The goal of this homework is to use WebGL with control code written in JavaScript in order to draw an interactive 3D dog in an HTML canvas in a web browser. The source code for this project can be accessed from my GitHub page.

2 Development

A series of improvements were implemented in the base code in order to achieve the following effects:

1. Create a hierarchical model of a (simplified) dog, composed of the body, 4 legs (each one composed of 2 independent components, upper and lower leg), head and tail. All components are cubes, use the cube function present in the file
2. Add a procedural texture to the body of the dog. The texture should be a checkerboard pattern but with a linear decrease of intensity from the front to the back of the body.
3. Add a button that starts an animation of the dog so that, starting from an initial position where it is standing and positioned along the x axis, it walks to the right by moving (alternatively back and forth) the legs and turns the head in the direction of the viewer.

2.1 Hierarchical model

For the implementation of this item, the base model was used and adapted to generate the dog shown in figure 1. The dog is composed of the head, the body, the tail and four legs with upper and lower parts.

The hierarchical model means that the parts are related to each other, starting from the torso (body), the other parts are positioned and oriented in relation to its parent. Having the following relation:

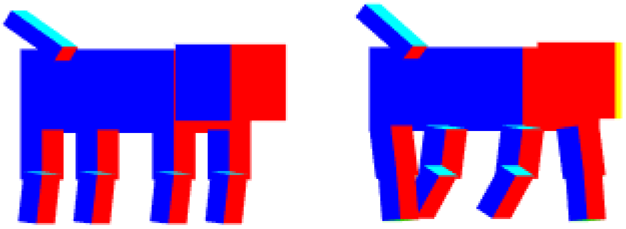

Figure 1: Dog model

1. body
    1. (a) head
    2. (b) tail
    3. (c) left front upper leg
        - i. left front lower leg
    4. (d) right front upper leg
        - i. right front lower leg
    5. (e) left rear upper leg
        - i. left rear lower leg
    6. (f) right rear upper leg
        - i. right rear lower leg

2.2 Procedural texture

In order to add the procedural texture, the *shader* and the *JavaScript* were modified, in the *JavaScript* file goes the code to generate the images and in the *shader* the multiplication matrices between the colors and the texture presented in figure 2.

JavaScript

```
var texSize = 256;
var numChecks = 8;

var image1 = new Uint8Array(4*texSize*texSize);
for ( var i = 0; i < texSize; i++ ) {
    for ( var j = 0; j <texSize; j++ ) {
        var patchx = Math.floor(i/(texSize/numChecks));
        var patchy = Math.floor(j/(texSize/numChecks));
        if(patchx%2 ^ patchy%2) c = 255;
        else c = 0;
        image1[4*i*texSize+4*j] = c;
```

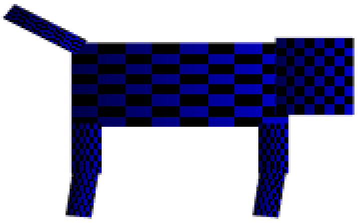

Figure 2: Dog model with texture

```
        image1[4*i*texSize+4*j+1] = c;
        image1[4*i*texSize+4*j+2] = c;
        image1[4*i*texSize+4*j+3] = 255;
    }
}

var image2 = new Uint8Array(4*texSize*texSize);
for ( var i = 0; i < texSize; i++ ) {
  for ( var j = 0; j <texSize; j++ ) {
    c = 200-j/2;
    image2[4*i*texSize+4*j] = c;
    image2[4*i*texSize+4*j+1] = c;
    image2[4*i*texSize+4*j+2] = c;
    image2[4*i*texSize+4*j+3] = 255;
  }
}
```

Shader

```
void main()
{
    gl_FragColor = fColor*(texture2D(Tex0, fTexCoord)*texture2D(
        Tex1, fTexCoord));
}
```

2.3 Animation

For the animation, all the angles movements were implemented in the render function. The head needs to turn in the direction of the viewer, all the upper and lower parts of the legs are moving, and also the tail moving up and down, trying to imitate the movement of a dog.

Also, a button is present in the screen, so the user can start and pause the walk. It is important to notice that when the user pause the walking the dog stops with all feet on the ground and not in the middle of a step.

3 Conclusion

After implement all the features it was possible to see the evolution of the dog as more features were being added, how to work with hierarchical models, which can be really useful when working with more complex models, implement different kinds of textures and more important, how to add animation for a model.