Foundations of Computer Graphics

Assignment 6

Instructions:

- Assignments can be submitted in groups of at most three. The purpose of groups is to learn from each other, not to divide work. Each member should participate in solving the problems and have a complete understanding of the solutions submitted.
- Submit your assignments as a zip file (one per group) which includes the Common directory and a separate directory for each of the assignments so that we can run your code by just extracting the zip file and double-clicking the html files.

Problem 1 (20 points)**.**
The goal in this exercise is to create a globe with elevations. To do this, we render a sphere with the image `earth-diffuse.jpg` as the texture wrapped on it and using the image `earth-normal.jpg` as the normal map. For convenience, we will assume that the sphere is centered at the origin.

1. What are the texture coordinates of a point $p = (x, y, z)$ on the surface of the sphere? *Hint:* Use polar coordinates of p. See Lecture 10 slides.

2. What are the unit tangent, bitangent and normal vectors at the point $p = (x, y, z)$?

3. Modify the `Sphere()` function from Problem 1 of Assignment 5 so that the returned object contains three new properties `texCoords`, `tangents` and `bitangents` each of which is an array. The entries `texCoords[i], tangents[i]` and `bitangents[i]` should contain the texture coordinates, tangent and bitangent respectively at the i^{th} vertex.

4. Modify the function `objInit(Obj)` so that it sets up the necessary texture objects and buffers for the texture and normal maps. The image to be used for texture mapping is `Obj.diffuseMap` and the image to be used for normal mapping is `Obj.normalMap`. As a convention, we will use texture unit 0 for the texture associated with `Obj.diffuseMap` and texture unit 1 for the texture associated with `Obj.normalMap`. You may use the function `newTexture` in `Common\Utils\Texture.js` for creating texture objects.

   The function `Obj.draw` should be appropriately set so that texture mapping is used if `Obj.diffuseMap` is defined and normal mapping is used if `Obj.normalMap` is defined. Note that either or both of the properties `diffuseMap` and `normalMap` may be absent in `Obj`. Your code should work in all cases.

5. Display a sphere obtained using the `Sphere` function and using `earth-diffuse.jpg` and `earth-normal.jpg` as the texture and normal maps respectively. Choose the parameters in Phong Lighting appropriately so that the color and elevations on the sphere are clear. Please provide a virtual trackball interface so that the user can rotate the sphere.

   You code should look something like:

```
window.onload = function(){
    ...
    var sphere = Sphere();
    sphere.diffuseMap = "earth-diffuse.jpg";
    sphere.normalMap = "earth-normal.jpg";
    ...
    requestAnimationFrame(render);
};

function render(){
    ...
    sphere.draw();
}

function Sphere(){
    ...
}
```

Problem 2 (10 points)**.**
Use `Models\obj2json.html` to convert `Models\Rabbit\rabbit.obj` to a javascript object. Save the output in a file `rabbit.js` and give a suitable name to the output object (e.g. `var rabbit =` $\{\cdots\}$`;`). Include the file `rabbit.js` in your html file. Display the corresponding model with `Models\Rabbit\rabbitDiffuse.jpg` as the texture map. Use a small shininess constant. Provide a virtual trackball interface so that the user can move the model.

Note that you can use the code from previous exercise. You just need to replace the *sphere* by the *rabbit* and disable Normal Mapping.

Problem 3 (10 points)**.**
Implement a sky box, using a cube along with `cubemap.jpg` as the texture. The camera should be placed at the center of the cube. Since the texture represents scenery at a distance the camera location should always be at the center of the cube but the user should be able to change the gaze direction using the arrow keys. Only diffuse lighting should be used.

Problem 4 (10 points)**.**
Do a non-recursive implementation of the recursive ray tracer in the folder '`Recursive Ray Tracer`'.

Add cylinders and cubes to the scene. For this you need to compute the intersection of a ray with a cylinder or a cube. Use instancing to reduce the computation to the intersection of a ray with a fixed object of each type.