Exercise 1

Orgho Anoronyo Neogi

09 December 2016

1. A function of two variables, each of which takes a value in the interval [0,1], that returns a vector that describes a point on a cylinder (but not on the cylinder's circular end caps).

```
function cylinder(u, v) {
    var radius = 1; //radius
    var height = 1; //max height
    var angle = v * 2 * Math.PI; //scaling v to angle
    var x = radius * Math.cos(angle);
    var y = radius * Math.sin(angle);
    var z = u * height; //scaling u to height
    return new THREE.Vector3(x, y, z);
}
```

2. A function of two variables, each of which takes a value in the interval [0,1], that returns a vector that describes a point on a cone (but not on the cone's circular base).

```
function cone(u, v) {
    var radius = 1; //max radius
    var height = 1;
    var angle = v * 2 * Math.PI; //scaling v to angle
    var x = radius * u * height * Math.cos(angle); //scaling u to height
    var y = radius * u * height * Math.sin(angle); //scaling u to height
    var z = height * u; //scaling u to height
    return new THREE.Vector3(x, y, z);
}
```

3. A function of two variables, each of which takes a value in the interval [0,1], that returns a vector that describes a point on a sphere.

```
function sphere(u, v) {
    var radius = 1;
    var angle2 = u * 2 * Math.PI; //scaling u to angle
    var angle1 = v * 2 * Math.PI; //scaling v to angle
    var x = radius * Math.cos(angle2) * Math.sin(angle1);
    var y = radius * Math.sin(angle2) * Math.sin(angle1);
    var z = radius * Math.cos(angle2);
    return new THREE.Vector3(x, y, z);
}
```

4. A mathematical expression that describes the brightness of a small patch of a surface. The expression should contain terms that model ambient lighting, diffuse reflection, and specular reflection. It should show the role of a vector that is perpendicular to the surface, a vector that points to the source of light, and a vector that points to the eye of the viewer in the calculation of these several components of the surface's illumination.

$$I_p = k_a i_a + \sum_{m \in lights} (k_d(\hat{L}_m \cdot \hat{N}) i_{m,d} + k_s(\hat{R}_m \cdot \hat{V})^\alpha i_{m,s})$$

k_s is specular reflection constant.

k_d is diffuse reflection constant.

k_a is ambient reflection constant.

α is "shininess" constant for this material

$lights$ is the set of all light sources

$\hat{L}_m$ is the direction vector from the point to light source specified by m

$\hat{N}$ is the normal vector

$\hat{R}_m$ is the direction of a perfectly reflected ray of light

$\hat{V}$ is the direction towards the viewer

5. A mathematical expression that describes a point on a cubic Bézier curve as a product of vector(s) and matrice(s).

$$\begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = P(t)$$

Where p_0, p_1, p_2 and p_3 are the 4 control points

6. A matrix that describes a perspective transformation.

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Where f stands for far value, n stands for near value, t stands for top, b stands for bottom, r stands for left and l stands for right.

7. An image from a program that you write. This program will produce an image of one of the American manned spacecraft of the 1960s and 1970s. Begin by modelling a vehicle with cylinders and cones, then elaborate. Add more details to the vehicle, a background that might include planets and stars, or add animation.

See this image 8:

8. An excerpt from your program that shows us some key feature of the program. This might be a loop, the definition of a function, a call to a function, an assignment state that contains on its right hand side a key expression, or something else.

```
// Load the JSON files and provide callback functions (modelToScene
var loader = new THREE.JSONLoader();
loader.load("1.json", addModelToScene);
loader.load("2.json", addModelToScene);
```

```
loader.load("4.json", addModelToScene);


// After loading JSON from our file, we add it to the scene
function addModelToScene(geometry, materials) {
    var material = new THREE.MeshFaceMaterial(materials);
    model = new THREE.Mesh(geometry, material);
    //  model.scale.set(5,5,5);
    scene.add(model);
    // camera.lookAt(model.position);
}

function createStars(radius, segments) {
    return new THREE.Mesh(
        new THREE.SphereGeometry(radius, segments, segments),
        new THREE.MeshBasicMaterial({
            map: starsTex,
            side: THREE.BackSide
        })
    );
}
```

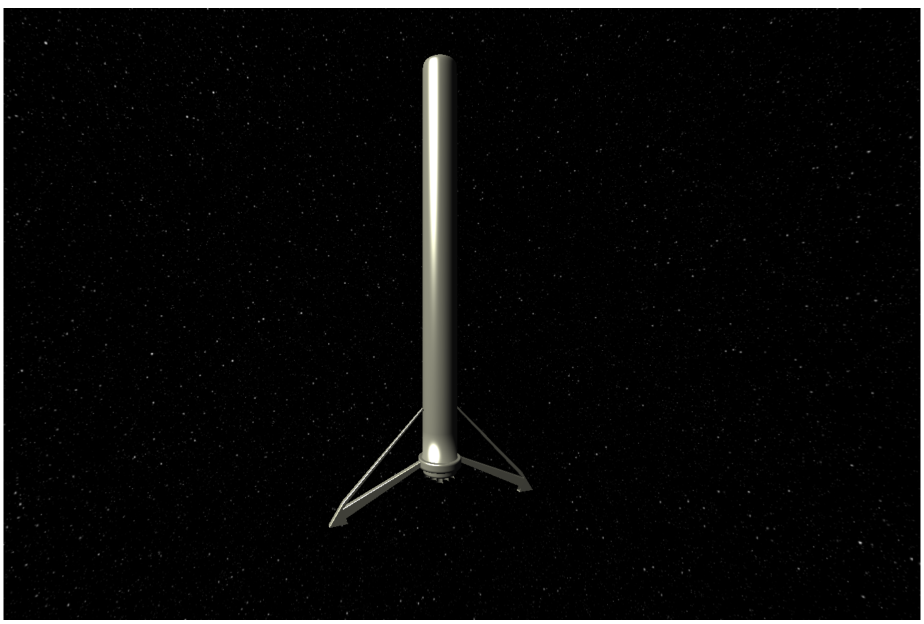

Figure 1: Rocket