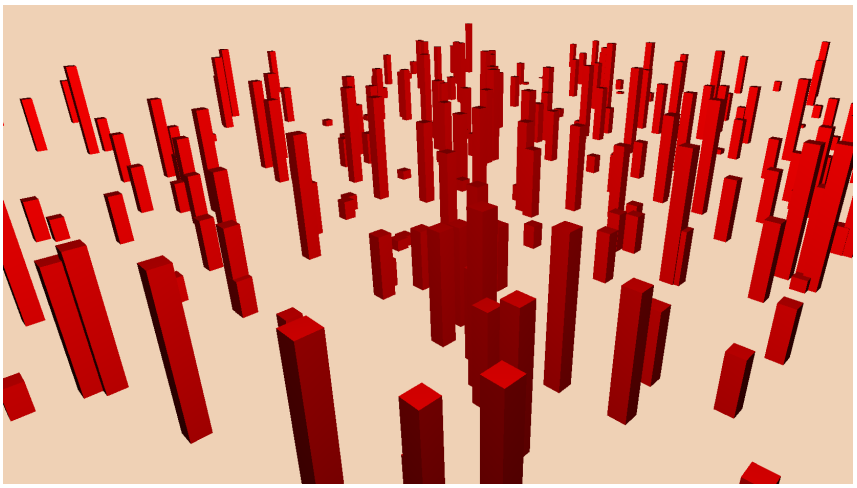
Computer Graphics - "Flying"

CS 4830 — Dr. Mihail

April 1, 2019

1 Introduction

In this assignment you will implement flying in 3D through a city made up of blocks.



1.1 Buildings

The buildings are created using simple box geometries, Lambertian material with random height and positions. The code to create one such item is as follows:

```
var geometry = new THREE.BoxGeometry(width, height, length);
var material = new THREE.MeshLambertMaterial( { color: 0xff0000 } );

var mesh = new THREE.Mesh( geometry, material);

mesh.position.x = 0;
mesh.position.y = 0;
mesh.position.z = 0;
mesh.needsUpdate = true;

scene.add(mesh);
```

1.2 Mousemove

The most important concept you will implement in this assignment is moving through the world by controlling the camera using both the keyboard and the mouse. You have to implement this yourself, i.e., if you use THREE mousemove, you will not receive credit for this assignment. The WASD keys will

control strafe using keys A and D for moving left and right. Moving forward and backward will be done using keys W and S.

In Three.js, this can be implemented using the PerspectiveCamera object's position and up attributes, as well as lookAt methods. To implement mouselook, you have to keep track of 3 vectors:

1. cameraLookAt
2. cameraUp
3. cameraLeft

These vectors can be initialized as follows:

```
var cameraLookAt = new THREE.Vector3(0, 0, -1);
var cameraLeft = new THREE.Vector3(-1, 0, 0);
var cameraUp = new THREE.Vector3().crossVectors(cameraLeft, cameraLookAt);
```

Moving through space involves updating the camera object's position attribute. Pressing A and D makes the camera position move along the positive and negative cameraLeft vector, while pressing W and S makes the camera move along the cameraLookAt vector. This update is done by vector addition. Keep in mind that adding a unit vector to the camera object's position will make a very large change (one unit). If the entire world in a unit square space, one second will move the camera very far, thus scaling the cameraLookAt and cameraLeft vectors when adding them to the camera position vector may be necessary.

Mouse input is handled by a custom function that receives the mouse coordinates (in screen space). These coordinates need to be transformed into two scalars: amount of movement along the x-axis and amount of movement in the y-axis. These can be negative and positive and should be used to update the yaw and pitch angles, where yaw affects the cameraLeft vector and pitch affects the cameraLookAt vector.

Vector3 has a method applyAxisAngle, that takes 2 arguments: a vector around which to rotate, and an angle. Documentation for that method can be found here: `https://threejs.org/docs/#api/math/Vector3.applyAxisAngle`. Updating the camera object's position is done by updating the .position attribute: `https://threejs.org/docs/#api/core/Object3D.position`. The camera's lookAt vector is updated via a method of the camera object called lookAt, that takes one argument, Vector3. This vector is the sum of the camera's position vector and the cameraLookAt unit vector. During the rendering loop, BOTH the camera's lookAt vector and the camera's up vector has to be updated. The camera up vector has to be done before the lookAt function call, and is done by changing an attribute called up.

2 Implementation details

One possible way to implement mousemove is to keep track of two angles: pitch and yaw. The pitch is driven by the mouse y-axis and the yaw is driven by the mouse x-axis. These changes are incremental, i.e., for every mousemove event, you record the new position and save the old position. This will give you the difference, that can be applied (after scaling) to pitch and yaw. The pitch and yaw angles are then applied to the two vectors used to initialize the camera: lookAt and cameraLeft.

3 Due Date

This assignment is submitted via the electronic submission form on the course web page (as a zip file) and is due before midnight on Sunday, April 7th.

4 Grading Rubric

This assignment will receive 0 credit if mousemove is not implemented.

- Keys WASD work as specified 50%
- Moving the mouse changes the three vectors (lookAt, up and left) properly 50%

5 Extra Credit

It is possible to earn up to 100% extra credit by turning this assignment into a game. To receive this credit, the basic functionality has to be implemented.