Simple example of Three.js usage

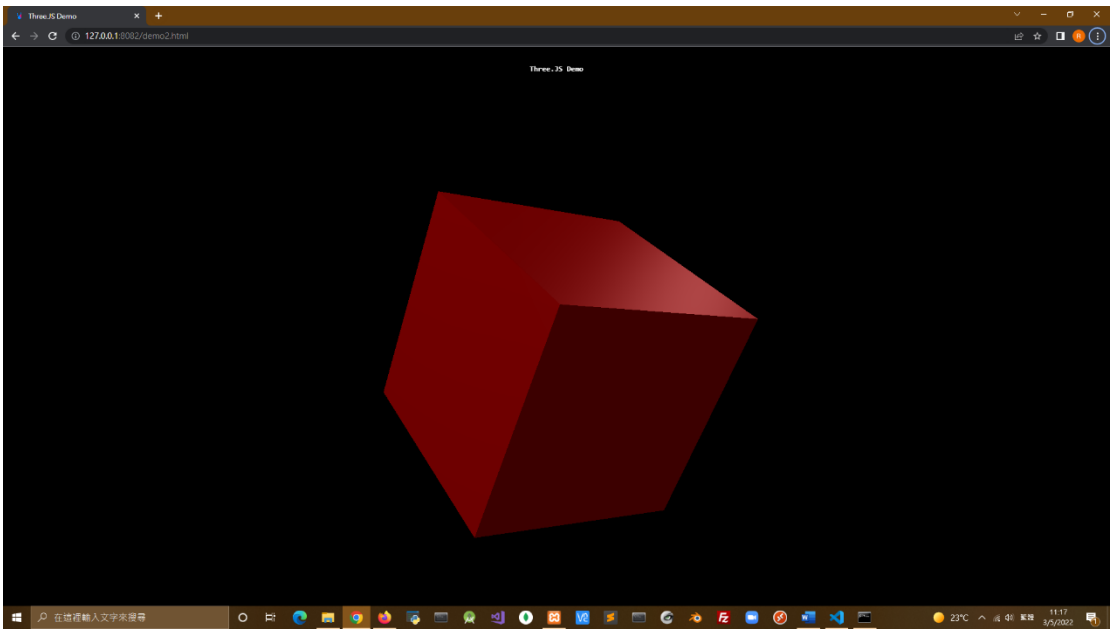

Fig.1 – A simple example of displaying a box using Three.js library

A simple example of using Three.js is depicted in Fig.3. In its minimal form, there is a scene with a camera, lighting and a 3D object (e.g. a box).

To construct a web page like this, we first download the Three.js library from https://threejs.org/. Create a folder and expand the library files inside. Then construct a simple html file called index.html, place it in the same folder and import the Three.js script like this in the body section of the file :

```
<script src="js/three.js"></script>
<script src="js/OrbitControls.js"></script>
```

where OrbitControls.js is the JavaScript file for camera control.

In order to display a 3D model, we need a renderer. The renderer is derived from WebGL, the standard for low-level 3D graphics API. Define the renderer and append to the web document like this :

```
// Renderer
renderer = new THREE.WebGLRenderer();
renderer.setSize(window.innerWidth, window.innerHeight);
renderer.setClearColor( 0x000000 );   // Background color
document.body.appendChild(renderer.domElement);
```

The following lines define the scene, the camera and the 3D object. Finally we add the box to the scene using scene.add() to display it :

```
    // Define a scene
    scene = new THREE.Scene();

    // Define a camera
    camera = new THREE.PerspectiveCamera(45, window.innerWidth /
window.innerHeight, 0.01, 10000);
    camera.position.set(100, 100, 100);

    // Construct a box
    var geometry = new THREE.BoxGeometry( 30, 30, 30 );
    var material = new THREE.MeshPhongMaterial( { color: 0x3c0000 });
    box = new THREE.Mesh( geometry, material );

    scene.add( box );
```

In order to control the camera to move around the object, we have to define an Orbit Control like this :

```
    // Define the camera control
    controls = new THREE.OrbitControls(camera, renderer.domElement);
    controls.enabled = true;
```

Add some fancy effect to let the box to rotate by itself :

```
var update = function() {

          box.rotation.x += 0.005;
          box.rotation.y += 0.005;
          box.rotation.z += 0.005;

        };
```

Define the renderer function to render the scene with the camera :

```
        var render = function() {

            renderer.render(scene, camera);

        };
```

Define an animate function to articulate all the above :

```
var animate = function() {

  requestAnimationFrame(animate);
  update();
  controls.update();

  render();
};
```

In order that the 3D object can be seen, we should add some lighting e.g., ambient light :

```
// Ambient light
var ambientLight = new THREE.AmbientLight(0xffffff, 1.0);
scene.add(ambientLight);
```

All Three.js programs comprise the above essential elements and if you want to know more about the use of different parameters in the constructors, see the Three.js documentation at :
https://threejs.org/docs/index.html#manual/en/introduction/Creating-a-scene

About how to run a web site locally, see :
https://threejs.org/docs/#manual/en/introduction/How-to-run-things-locally