Damon Bruccoleri Computer Graphics CISD-0792 Dr. Laszlo Instructor

Assignment 4 Date: 3/10/2015

**[Problem C1 Random Boxes (ctrl-click here to load)](http://damon4.com/Computer%20Graphics%20HW/hw_shell.html?load=C1.js)**

function randomBoxes(nbrBoxes, minSide, maxSide, minHeight, maxHeight){

var base = new THREE.Object3D();

for ( var i=0; i<nbrBoxes; i++){

var boxmat = new THREE.MeshLambertMaterial({ transparent : true });

boxmat.opacity = 0.8;

boxmat.color = new THREE.Color().setHSL(Math.random(),

Math.random()\*0.15+0.8,Math.random()\*0.4+0.3);

var width = Math.random()\*(maxSide-minSide)+minSide;

var depth = Math.random()\*(maxSide-minSide)+minSide;

var height = Math.random()\*(maxHeight-minHeight)+minHeight;

var boxgeom = new THREE.BoxGeometry(width, height, depth);

var mesh = new THREE.Mesh(boxgeom, boxmat);

mesh.position.x = -100 + width/2 + (200-width) \* Math.random();

mesh.position.y = height/2;

// box is to sit on the x-z plane

mesh.position.z = -100 + depth/2 + (200-depth) \* Math.random();

base.add(mesh);

}

return base;

}

function createScene() {

var planegeom = new THREE.PlaneGeometry( 200, 200 );

var planemat = new THREE.MeshBasicMaterial({ color : 'grey'});

var planeMesh = new THREE.Mesh(planegeom, planemat);

planeMesh.rotation.x = -Math.PI/2; // rotate mesh into x-z plane

scene.add(planeMesh);

var boxmesh = randomBoxes(100, 5, 20, 5, 60);

scene.add(boxmesh);

…

[**Problem BA1 Cube Color Animation w/GUI (ctrl-click here to load)**](http://damon4.com/Computer%20Graphics%20HW/hw_shell.html?load=BA1.js)

var mat = new THREE.MeshLambertMaterial();

function Controls() {

this.opacity = 1;

this.rate = 0.2;

}

function initGui() {

gui = new dat.GUI();

controls = new Controls();

gui.add(controls, 'opacity', .1, 1).step(.1);

gui.add(controls, 'rate', 0.01, 2).step(0.01);

}

function ExpandObjectDimension( obj, n){

var newobj = obj.clone();

if (n>1)

newobj.add( ExpandObjectDimension(obj, n-1));

return newobj;

}

// M x N x O matrix of boxes centered in the xyz-plane

function createCube3DMatrix(m, n, o, offset) {

var offset = (offset !== undefined) ? offset : 2.0;

var geom = new THREE.CubeGeometry(1, 1, 1);

mat.color.set("red"); // initial color cause renderer is incremental

mat.transparent = true;

var protoMesh = new THREE.Mesh(geom, mat);

protoMesh.position.set(offset,0,0);

protoMesh = ExpandObjectDimension(protoMesh, m);

protoMesh.position.set(0,offset,0);

protoMesh = ExpandObjectDimension(protoMesh, n);

protoMesh.position.set(0,0, offset);

protoMesh = ExpandObjectDimension(protoMesh, o);

var xMin = -offset \* ((m-1) / 2.0);

var yMin = -offset \* ((n-1) / 2.0);

var zMin = -offset \* ((o-1) / 2.0);

protoMesh.position.set( xMin, yMin, zMin);

scene.add(protoMesh);

}

function render() {

var delta = clock.getDelta();

mat.color.offsetHSL(delta\*controls.rate,0,0);

mat.opacity = controls.opacity;

cameraControls.update(delta);

renderer.render(scene, camera);

}

function createScene() {

createCube3DMatrix(9,9,9);

…

[**Problem BA2 Animated Solar System (ctrl-click here to load)**](http://damon4.com/Computer%20Graphics%20HW/hw_shell.html?load=BA2.js)

function createsatellite( name, size, satcolor, distance, rps\_speed){

// each satellite consists of two objects. An object3D that is used to make

// the satellite revolve, and its child. The child does the translation and

// holds the mesh of the object geometry/material. It also has an optional

// rotation to simulate "day/night" roll of the satellite.

if(name=='sun') // seperate material for sun so it does not have dark side

var satellite = new THREE.Mesh(new THREE.SphereGeometry(size,40,40),

new THREE.MeshBasicMaterial({color:satcolor}));

else

var satellite = new THREE.Mesh(new THREE.SphereGeometry(size,40,40),

new THREE.MeshPhongMaterial({color:satcolor}));

satellite.position.set(distance.x, distance.y, distance.z);

satellite.userData.rps\_speed = [0,0,0]; // simulate day/night here

satellite.userData.rps = new THREE.Euler(0,0,0);

var satelliteParent = new THREE.Object3D(); // each sat has a parent obj to allow

satelliteParent.add(satellite); // translation before rotation

satelliteParent.userData.rps\_speed = rps\_speed; // store animation info in obj

satelliteParent.userData.rps = new THREE.Euler(0,0,0);

satelliteParent.name = name;

return satelliteParent;

}

function createScene() {

const PI = Math.PI;

var dist = new THREE.Vector3( 0,3,0);

var rps = [0,0,PI];

var moon = createsatellite('moon', 0.3, 'LightGray', dist, rps);

dist.set(10,0,0);

rps= [0, PI/16,0];

var earth = createsatellite('earth', 1, 'blue', dist, rps);

earth.children[0].add(moon);

rps = [0, -PI/128, 0];

dist.set(-25.0,0,0);

var pluto = createsatellite('pluto', 0.1,'Gray', dist, rps);

dist.set(0,0,0);

rps = [0,0,0];

sun = createsatellite('sun', 3, 'yellow', dist, rps);

sun.children[0].add(pluto);

sun.children[0].add(earth);

light = new THREE.PointLight(0xFFFFFF, 1.0);

light.position.set(0, 0, 0);

var ambientLight = new THREE.AmbientLight(0x444444);

scene.add(light);

scene.add(ambientLight);

scene.add(sun);

}

function EulerAddMult( E, a, m){ //E passed by ref so E gets updated too

const PI2 = Math.PI\*2;

return E.set((E.x+a[0]\*m)%PI2, (E.y+a[1]\*m)%PI2, (E.z+a[2]\*m)%PI2 );

}

function render() {

var delta = clock.getDelta();

sun.traverse( function ( obj ){

obj.rotation = EulerAddMult( obj.userData.rps, obj.userData.rps\_speed,delta);} );

cameraControls.update(delta);

renderer.render(scene, camera);

}

function animate() {

window.requestAnimationFrame(animate);

render();

}

[**Problem MFWA1 Multiple Wave Animation (ctrl-click here to load)**](http://damon4.com/Computer%20Graphics%20HW/hw_shell.html?load=MFWA1.js)

function heightFunction(d, max,freq) {

return ((d > 0)&&(d< waveLimit)) ? max \* Math.sin(freq\*d) : 0;

}

function colorFunction(ht,max) {

var colorIndex = Math.floor((ht /max) \*(nbrColors/2)+nbrColors/2);

if (colorIndex<0)

colorIndex=0;

if (colorIndex>200)

colorIndex = 200;

return colors[colorIndex];

}

var newpress= false;

function updateSquares(deltat) {

if (newpress){

if(SelectedSquares.indexOf(touchedSquare) == -1){ // don't allow same square twice

touchedSquare.curWave = 0;

touchedSquare.waveRate = minWaveRate+Math.random()\*(maxWaveRate-minWaveRate);

touchedSquare.maxHeight = minHeight + Math.random()\*(maxHeight-minHeight);

touchedSquare.freq = minFreq + Math.random()\*(maxFreq-minFreq);

initializeColors();

SelectedSquares.push(touchedSquare);

}

newpress=false;

}

for ( var s = 0; s < SelectedSquares.length ; s++){

var theSquare = SelectedSquares[s];

var changed = false;

theSquare.curWave += (theSquare.waveRate \* deltat);

for (var i = 0; i < theObjects.length; i++) {

var obj = theObjects[i];

var dist = distance(theSquare , obj);

var delta = theSquare.curWave - dist;

var ht = heightFunction(delta, theSquare.maxHeight, theSquare.freq);

if (ht!=0)

changed=true; // if ht non zero then we changed

if (s==0)

obj.position.z = 0; // first wave processed initializes z

obj.position.z += ht; // waves add

obj.material.color = colorFunction(obj.position.z, theSquare.maxHeight);

}

if(!changed)

SelectedSquares.splice(s--,1);// its not contributing to the waves

}

}

function onDocumentMouseDown(event) {

var mouseVec = new THREE.Vector3(2\*(event.clientX/canvasWidth)-1, 1-2\*(event.clientY/canvasHeight), 0);

var raycaster = projector.pickingRay(mouseVec.clone(), camera);

var intersects = raycaster.intersectObjects(theObjects);

if (!newpress && (intersects.length > 0)) {

// select the closest intersected object

touchedSquare = intersects[0].object;

newpress=true;

}

}

function render() {

var delta = clock.getDelta();

updateSquares(delta);

cameraControls.update(delta);

renderer.render(scene, camera);

}

..