Поскольку вращение происходит вокруг центра координат, то для того, чтобы вращать фигуру относительно другого центра координат требуемый центр вращения вместе с фигурой переносится в центр координат, осуществляется поворот и центр вращения вместе с фигурой возвращается обратно.

Методы zoom, move, moveOx, moveOy, moveOz должны возвращать только соответствующую матрицу преобразования. В math3D метод multMatrix назвать multPoint, а метод который перемножает матрицы назвать multMatrix. В math3D напишем несколько методов.

Math3D.js

transform(matrix, point){

const result = this.multPoint(matrix, [point.x, point.y, point.z, 1]);

point.x = result[0];

point.y = result[1];

point.z = result[2];

}

getTransform(…args){

return args.reduce(

(s,t) => this.multMatrix(s,t),

[[1,0,0,0],

[0,1,0,0],

[0,0,1,0],

[0,0,0,1]]

);

}

Surface.js

class Surface{

constructor(points = [], edges = [], polygons = [], center = new Point){

this.points = points;

this.edges = edges;

this.polygons = polygons;

this.center = center;

this.animations = [];

}

dropAnimation(){

this.animations = [];

}

addAnimation(method, value, center){

this.animations.push({method,value,center:center||this.center});

}

doAnimation(math3D){

this.animations.forEach(anim => {

const T1 = math3D.move(-anim.center.x, -anim.center.y, -anim.center.z);

const T2 = math3D[anim.method](anim.value);

const T3 = math3D.move(anim.center.x, anim.center.y,anim.center.z);

const matrix = math3D.getTransform(T1,T2,T3);

this.points.forEach(point => math3D.transform(matrix,point));

math3D.transform(matrix, this.center);

});

}

Graph3D.js

constructor(){

//…

this.scene = this.SolarSystem()

setInterval(() => {

this.scene.forEach(surface=>surface.doAnimation(this.math3D));

this.renderScene();

},50);

}

wheel(event){

const delta = …;

const matrix = this.math3D.zoom(delta);

this.scene.forEach(surface => surface.points.forEach(point => this.math3D.transform(matrix, point)));

}

SolarSystem(){

const Earth = this.surfaces.sphere();

Earth.addAnimation(‘rotateOy’,0.1);

const Moon = this.surfaces.cube();

Moon.addAnimation(‘rotateOx’,0.2);

Moon.addAnimation(‘rotateOx’,0.05);

return[Earth,Moon];

}

Домашка: Реализовать код, запуск и остановку анимаций сделать в отдельный чекбокс

Задание\*:Добавить в анимацию Солнце

Задание\*\*:из программы компоненты вынести функционал в интерфейс

Задание\*\*\*:Сделать одну общую матрицу преобразования на которую умножать все фигуры