Визуальное преобразование на сцене возможно двумя способами:

1. Изменение самой сцены
2. Изменение положения камеры

Мы будем использовать первый способ. Это даст нам фиксированное положение камеры и угол её обзора. Изменение сцены происходит через изменение каждой точки сцены. Любые преобразования трехмерной сцены производится с помощью матрицы поворота. Есть матрица общего преобразования, мы применим частные случаи масшатбирования, переноса, поворота. Матрицы поворота вокруг осей устроены таким образом, что они поворачивают относительно точки координат. Матрица преобразования имеет размерность 4 на 4. Точка которую мы умножаем на матрицу, тоже должна иметь размерность 4 на 4.

Math3D.js

multMatrix(T,m){

const a = [0,0,0,0];

for (let i=0; I < T.length; i++){

let b = 0;

for (let j=0; j < m.length; j++){

b+=T[j][i]\*m[j];

}

a[i]=b;

}

return a;

}

zoom(point, delta){

const T=[

[delta,0,0,0],

[0,delta,0,0],

[0,0,delta,0],

[0,0,0,1]

];

const array = this.multMatrix(T,[point.x, point.y, point.z, 1]);

point.x=array[0];

point.y=array[1];

point.z=array[2];

}

move(point, dx=0, dy = 0, dz = 0){

const T=[

[1,0,0,0],

[0,1,0,0],

[0,0,1,0],

[dx,dy,dz,1]

];

const array = this.multMatrix(T,[point.x, point.y, point.z, 1]);

point.x=array[0];

point.y=array[1];

point.z=array[2];

}

rotateOx(point, alpha){

const T=[

[1,0,0,0],

[0,Math.cos(alpha),Math.sin(alpha),0],

[0,-Math.sin(alpha),Math.cos(alpha),0],

[0,0,0,1]

];

const array = this.multMatrix(T,[point.x, point.y, point.z, 1]);

point.x=array[0];

point.y=array[1];

point.z=array[2];

}

rotateOy(point, alpha){

const T=[

[Math.cos(alpha),0, -Math.sin(alpha),0],

[0,1,0,0],

[Math.sin(alpha),0,Math.cos(alpha),0],

[0,0,0,1]

];

const array = this.multMatrix(T,[point.x, point.y, point.z, 1]);

point.x=array[0];

point.y=array[1];

point.z=array[2];

}

rotateOz(point, alpha){

const T=[

[Math.cos(alpha), Math.sin(alpha),0,0],

[-Math.sin(alpha),Math.cos(alpha),0,0],

[0,0,Math.cos(alpha),0],

[0,0,0,1]

];

const array = this.multMatrix(T,[point.x, point.y, point.z, 1]);

point.x=array[0];

point.y=array[1];

point.z=array[2];

}

У класса graph пропишем несколько callback’ов(на кручение колесиком и тд)

Graph3D.js

callbacks:{

wheel:(event) => this.wheel(event),

mousemove:(event) => this.mousemove(event),

mouseup:(event) => this.mouseup(event),

mousedown:(event) => this.mousedown(event)

}

//В контекст компоненты Graph3D надо накинуть флажок canMove

this.canMove= false;

mouseup(){

this.canMove=false;

}

mousedown(){

this.canMove=true;

}

wheel(event){

event.preventDefault();

const delta = (event.wheelDelta>0)?-0.3:0.3;

this.scene.points.forEach(point =>

this.math3D.zoom(point,delta));

this.renderScene();

}

mousemove(event){

if(this.canMove){

const gradus = Math.PI/180/4;

this.scene.points.forEach(point => {

this.math3D.rotateOx(point,this.dy – event.offsetY)\*gradus);

this.math3D.rotateOy(point,this.dx – event.offsetX)\*gradus);

});

this.renderScene();

}

this.dx=event.offsetX;

this.dy=event.offsetY;

}

// Домашка: все сделать

// Со звездой: на пкм относительно Oz, на скм относительно Ox и Oy