National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”

Coursework

from a discipline

“Visualization of graphical and geometric information”

Performed by:

Bohdan Kalika

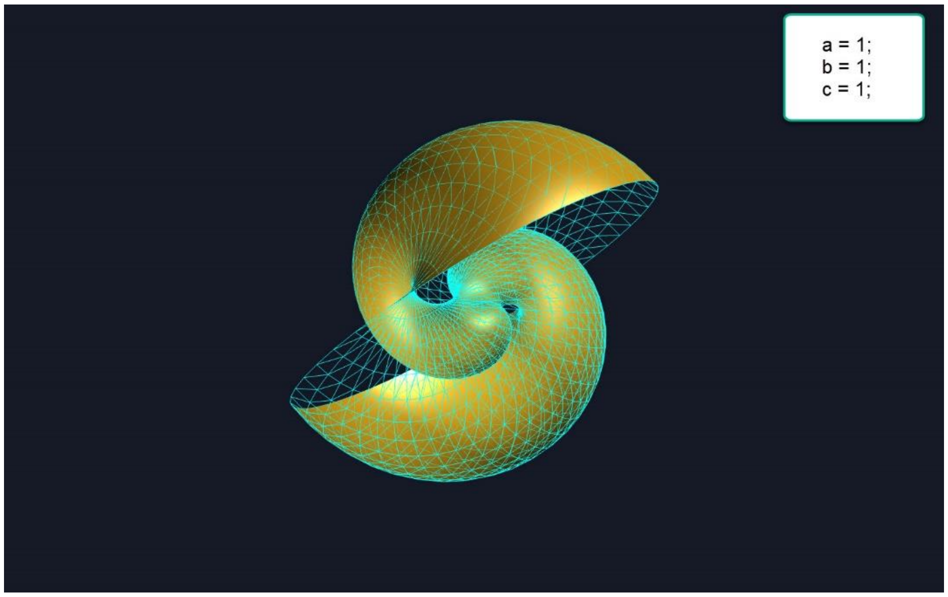
TM-01mp

Reviewed by:

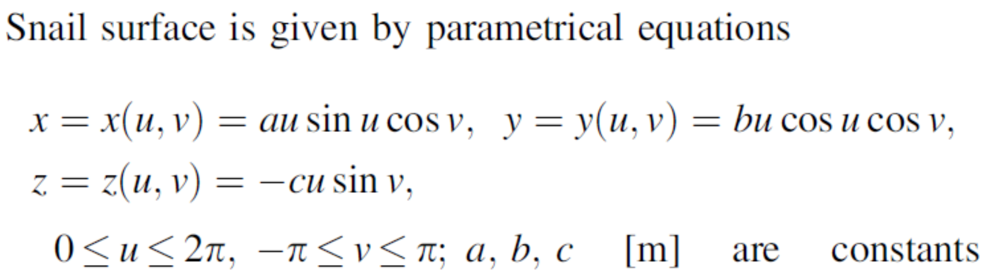
Anatol Demchyshyn

Kyiv – 2020

In this course work I have made tangent and a normal vector to the graphical figure “Snail surface”. The surface looks like:



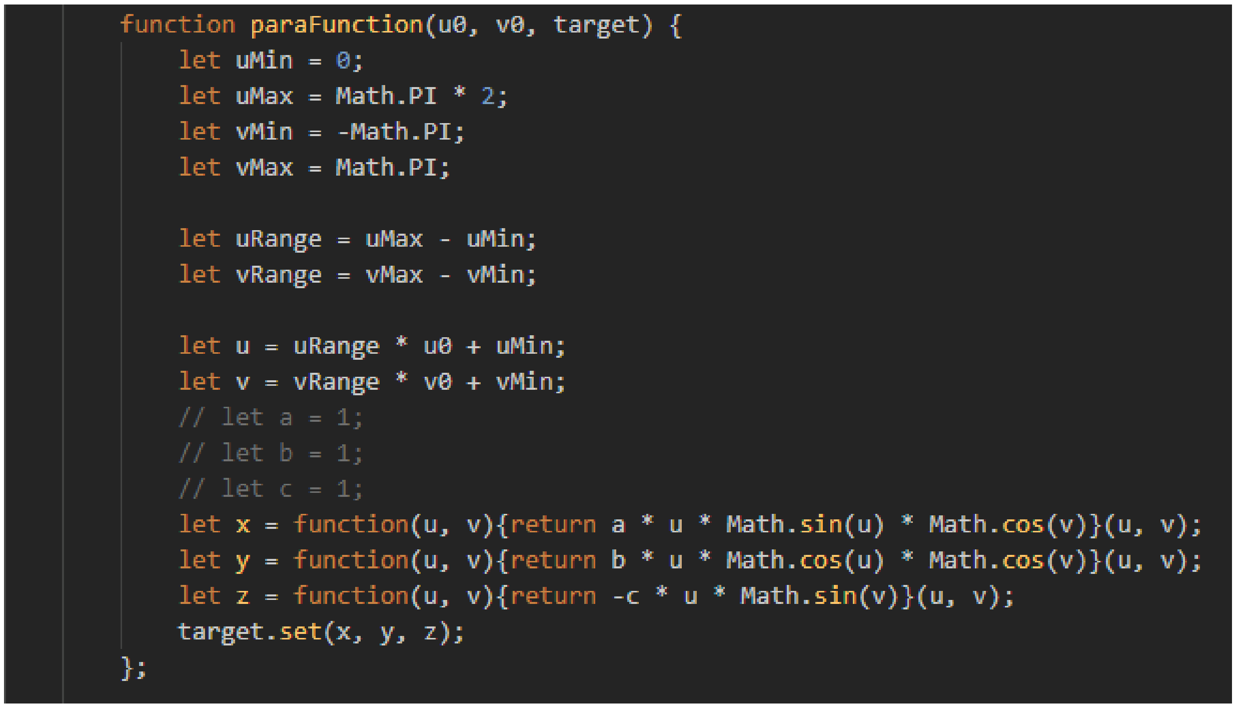
To create this surface, need to use a formula [1]:



Snail surface is given by parametrical equations

$$x = x(u,v) = au \sin u \cos v, \quad y = y(u,v) = bu \cos u \cos v,$$
$$z = z(u,v) = -cu \sin v,$$
$$0 \leq u \leq 2\pi, \ -\pi \leq v \leq \pi; \ a,\ b,\ c \quad [\mathrm{m}] \quad \text{are} \quad \text{constants}$$

For implementing this formula in a programing language, I choose to code it by the JavaScript and for visualize as a surface I used a THREEEJs library [2]:



```
function paraFunction(u0, v0, target) {
    let uMin = 0;
    let uMax = Math.PI * 2;
    let vMin = -Math.PI;
    let vMax = Math.PI;

    let uRange = uMax - uMin;
    let vRange = vMax - vMin;

    let u = uRange * u0 + uMin;
    let v = vRange * v0 + vMin;
    // let a = 1;
    // let b = 1;
    // let c = 1;
    let x = function(u, v){return a * u * Math.sin(u) * Math.cos(v)}(u, v);
    let y = function(u, v){return b * u * Math.cos(u) * Math.cos(v)}(u, v);
    let z = function(u, v){return -c * u * Math.sin(v)}(u, v);
    target.set(x, y, z);
};
```

```
var geometry = new THREE.ParametricGeometry(paraFunction, 50, 50);
var wireframeMaterial = new THREE.MeshBasicMaterial({color: 0xfffff, wireframe: true, transparent: true});
var material = new THREE.MeshPhongMaterial({
    color: 0xdaa520, // 0xcf0e0e;
    specular: 0xbcbcbc,
    transparent: true
});
```

Also, I have added to the scene a light, for better visualization of the surface.

The second part of the requirements was implementing tangent and a normal vector to our surface. For creating it, I created a function “calculateVector”

```
function calculateVector(u0, v0, x, y, z){
    let p0 = pointFunction(u0, v0);
    const du = 0.1;
    let p1 = pointFunction(u0+du, v0);
    let df = [(p1[0]-p0[0])/du, (p1[1]-p0[1])/du, (p1[2]-p0[2])/du];
    const dv = 0.1;
    let p2 = pointFunction(u0, v0+dv);
    let ds = [(p2[0]-p0[0])/dv, (p2[1]-p0[1])/dv, (p2[2]-p0[2])/dv];
    let f = [df[1]*ds[2] - df[2]*ds[1], df[2]*ds[0] - df[0]*ds[2], df[0]*ds[1] - df[1]*ds[0]];

    return [p0, df, ds, f]
}
```

And added it, to the script for rendering on the web page:

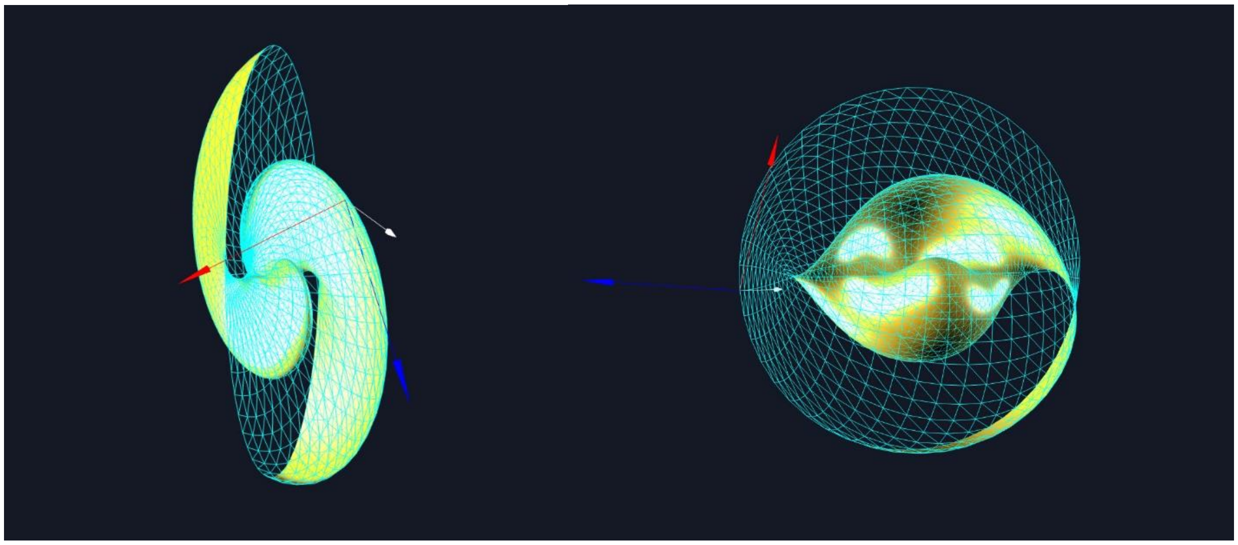
```
function render(){
    requestAnimationFrame(render);

    let vectors = calculateVector(pointU0, pointV0);
    const u0vector = new THREE.Vector3(vectors[1][0], vectors[1][1], vectors[1][2]);
    u0vector.normalize();
    const v0vector = new THREE.Vector3(vectors[2][0], vectors[2][1], vectors[2][2]);
    v0vector.normalize();
    const nvector = new THREE.Vector3(vectors[3][0], vectors[3][1], vectors[3][2]);
    nvector.normalize();

    const origin = new THREE.Vector3(vectors[0][0], vectors[0][1], vectors[0][2]);
    const uVector = new THREE.ArrowHelper(u0vector, origin, 5, "#006400");
    const vVector = new THREE.ArrowHelper(v0vector, origin, 5, "#ff0000");
    const normalVector = new THREE.ArrowHelper(nvector, origin, 5, "#ffffff");
    groupSecond.clear();
    groupSecond.add(uVector);
    groupSecond.add(vVector);
    groupSecond.add(normalVector);
    renderer.render(scene, camera);
}
render();
```

Also, I added a class ArrorHelper from THREEJs library for representing vectors near surface and viewed on the page.

After realization, it looks like this:



To the page, I have added a possibility to move a point vector. We can move it by keyboard keys.
A – move point to the left side,
D – right,
W – above
And S – down.

Additionally, I added a possibility to see the surface from different side and zoom in-out it. For realization it, I used a class OrbitControls from the library THREEJs.
To view all code implementation of the requirement, I will attach file to my mail.

References:

1. Encyclopedia of Analytical Surfaces - S.N. Krivoshapko, V.N. Ivanov. p.280.
2. https://threejs.org/docs/