

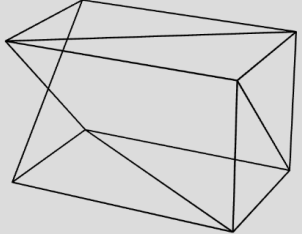
Algorithmic Art – Die Schnecke

1) 3D rectangle mit vertices

JS sketch.js

Rectangle > JS sketch.js > ...
1 let vertices = [];
2
3 function setup() {
4 createCanvas(400, 400, WEBGL);
5
6 vertices[0] = createVector(-80, -50, -50);
7 vertices[1] = createVector(80, -50, -50);
8 vertices[2] = createVector(80, 50, -50);
9 vertices[3] = createVector(-80, 50, -50);
10 vertices[4] = createVector(-80, -50, 50);
11 vertices[5] = createVector(80, -50, 50);
12 vertices[6] = createVector(80, 50, 50);
13 vertices[7] = createVector(-80, 50, 50);
14 }
15
16 function draw() {
17 background(220);
18 fill(200);
19
20 // LINE Mode weil sonst innen auch Dreiecke sind
21 beginShape(LINE_STRIP);
22
23 vertex(vertices[0].x, vertices[0].y, vertices[0].z);
24 vertex(vertices[1].x, vertices[1].y, vertices[1].z);
25 vertex(vertices[2].x, vertices[2].y, vertices[2].z);
26 vertex(vertices[3].x, vertices[3].y, vertices[3].z);
27 vertex(vertices[4].x, vertices[4].y, vertices[4].z);
28 vertex(vertices[5].x, vertices[5].y, vertices[5].z);
29 vertex(vertices[6].x, vertices[6].y, vertices[6].z);
30 vertex(vertices[7].x, vertices[7].y, vertices[7].z);
31 vertex(vertices[0].x, vertices[0].y, vertices[0].z);
32 vertex(vertices[4].x, vertices[4].y, vertices[4].z);
33 vertex(vertices[1].x, vertices[1].y, vertices[1].z);
34 vertex(vertices[5].x, vertices[5].y, vertices[5].z);
35 vertex(vertices[2].x, vertices[2].y, vertices[2].z);
36 vertex(vertices[6].x, vertices[6].y, vertices[6].z);
37 vertex(vertices[3].x, vertices[3].y, vertices[3].z);
38 vertex(vertices[7].x, vertices[7].y, vertices[7].z);
39 }

Simple Browser X
← → ↻ http://127.0.0.1:5500/



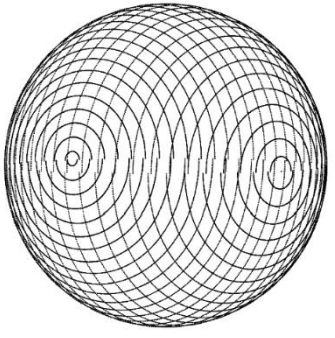
2) Sphere

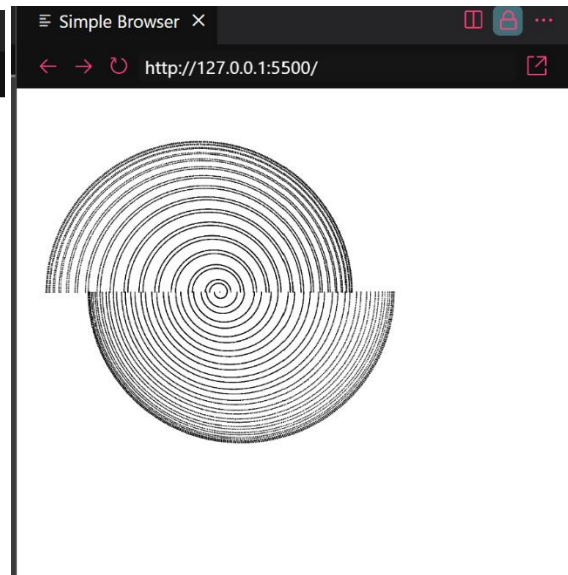
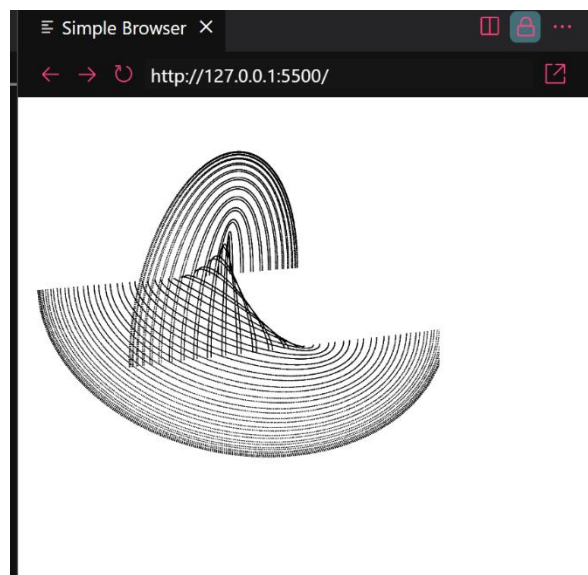
JS sketch.js x

Sphere > JS sketch.js > ...
1 var radius = 110;
2 var breitenwinkel = 0.01;
3 var längenwinkel = 0.1;
4
5 // Canvas wird nicht richtig angezeigt. KA warum.
6 function setup() {
7 createCanvas(300, 300, WEBGL);
8 strokeWeight(0.9);
9 }
10
11 function draw() {
12 background(300);
13
14 beginShape(POINTS);
15
16 // iteriert durch breitenwinkel, Längenwinkel (in radians) bis
17 for(var breite = 0; breite <= PI; breite += breitenwinkel){
18 for(var länge = 0; länge <= TWO_PI; länge += längenwinkel){
19
20 // berechnet jede sphärische Koordinate (Breitenwinkel, Längenwinkel, Radius) als Pu
21 // konvertiert es in die kartesischen Koordinaten (x, y, z) über Trigo
22
23 var x = radius * cos(breite) * sin(länge);
24 var y = radius * sin(breite) * sin(länge);
25 var z = radius * cos(breite); //wenn hier Länge, wirds zur normalen Kugel
26 // rendert jeden Punkt als Vertex (Ecke)
27 vertex(x, y, z);
28 }
29 }
30 }

Simple Browser x

http://127.0.0.1:5500/



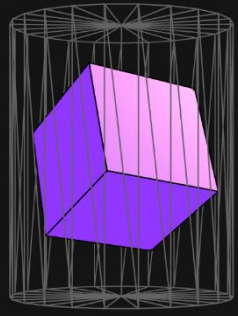


3) Cube in a cylinder

JS sketch.js

Cube in Cylinder > JS sketch.js > ...
1 function setup() {
2 createCanvas(400, 400, WEBGL);
3 }
4
5 function draw() {
6 background(20);
7
8 // Licht funktioniert nicht ganz. Geht mit bei der Drehung.
9 ambientLight(20); // Ambient light to brighten the scene
10 pointLight(255, 255, 255, width/2, -height/2, 0);
11
12 // Rot, Grün, Blau, Alpha
13 emissiveMaterial(125, 35, 240, 0.639);
14
15 push();
16 translate(0, -30, 0);
17 let axis = [1, 1, 0]; // Define the axis of rotation
18 rotate(QUARTER_PI, axis); // Rotate the box by 45 degrees around the axis
19 box(100, 100, 100);
20 pop();
21
22 push();
23 translate(0, -30, -0); // Move the cylinder down
24 rotateX(PI); // Rotate the cylinder to make it stand upright
25 noFill(); // make the cylinder see-through
26 stroke(100);
27 cylinder(90, 220);
28 pop();
29
30 orbitControl();
31 }
32

Simple Browser X
← → ↺ http://127.0.0.1:5500/



4) Donut (Supertoroid)

```
JS sketch.js x
Donut > JS sketch.js > draw
1 let r1 = 90; // Hauptradius
2 let r2 = 80; // Kleiner Radius
3 let angle = 0; // Rotation angle
4
5 function setup() {
6   createCanvas(400, 400, WEBGL);
7   angleMode(DEGREES);
8 }
9
10 function draw() {
11
12   // https://en.wikipedia.org/wiki/Supertoroid -> values here for superto
13   let t = 2.5;
14   let s = 1.92;
15
16   background(20);
17   orbitControl();
18
19   //noFill();
20   stroke('pink');
21   // Der Donut ist aus vielen kleinen Kreisen gemacht, die aneinander gereiht sind.
22   beginShape(TRIANGLE_STRIP);
23   for(let theta=0; theta<360; theta+=6) { // Breite der Schleife -> Glätte des Donuts
24     for (let phi=0; phi<360; phi+=40) { // Höhe der Schleife -> Dichte des Donuts
25       let a = cos(theta); // Wo der punkt links oder rechts ist. Wie bei einem Stück flach
26       let b = sin(theta); // '' -> theta + angle = Rotation<-
27       let c = cos(phi); // Wo der punkt oben oder unten ist
28       let d = sin(phi); // ''
29
30       let an = xn(a, t); // Streckung/Stauchung je nach Zahl bei t
31       let bn = xn(b, t);
32       let cn = xn(c, s); // Streckung/Stauchung je nach Zahl bei s
33       let dn = xn(d, s);
34
35       // Berechnung der x, y, z Koordinaten auf der Oberfläche des Shapes auf Basis der Su
36       let x = (r1 + r2*cn)*an;
37       let y = (r1 + r2*cn)*bn;
38       let z = r2*dn;
```

