Function Representation of the 3D objects

Aim: The study of the process of the geometric modeling with using the Function Representation of 3D models.

Task: Create 3D models using the geometric modeling language and system IZICAD (https://izicad.com/).

Result: The source code. The report.

Theoretical part:

IZICAD is a simple geometric modeling language for F-rep objects. F-rep stands for function representation. In F-rep, objects are described using a single real continuous function F(x1, x2, x3, ..., xn) >= 0. (Website of project: https://izicad.com/ and https://hyperfun.org)

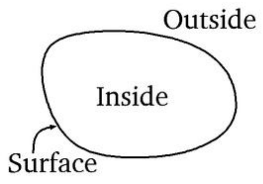
Since F-rep models are more general than traditional skeletal implicit surfaces, convolution surfaces, distance-based models, CSG (Constructive Solid Geometry), sweeps, and voxel models, IZICAD can deal with all these geometric models.

In IZICAD, F-rep objects are described using assignments, conditional selections, and iterations as in traditional programming languages. In addition to arithmetic and relational operators, IZICAD has built-in set-theoretic operators such as union, intersection, subtraction, and so on. Some predefined library primitives and operators are available in IZICAD. Using these functionalities, quite complex objects can be modeled.

F-rep

In F-rep (Function Representation), a geometric object is defined by a single real continuous function F(x1, x2, x3, ..., xn) >= 0. Let's think why F(x1, x2, x3, ..., xn) >= 0 can represent an object.

A geometric object in 3D space is described by: the inside, the surface, and the outside.

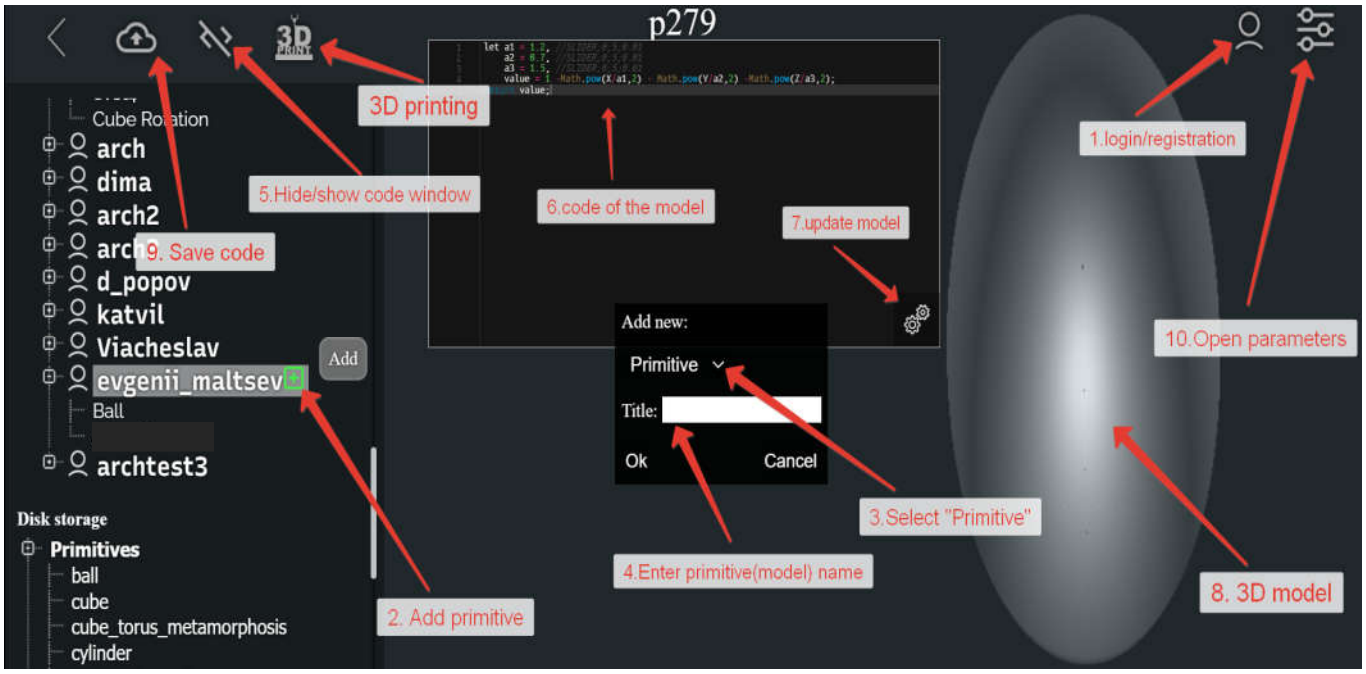


We assume that F(x1, x2, x3, ..., xn) >= 0 is a rule which determines where a given point belongs, that is:

F(x1, x2, x3, ..., xn) > 0 : the inside of an object

F(x1, x2, x3, ..., xn) = 0	:	the surface of an object
F(x1, x2, x3, ..., xn) < 0	:	the outside an object

IZICAD Interface



```
let a1 = 1.2, //SLIDER,[illegible]
    a2 = 0.7, //SLIDER,[illegible]
    a3 = 1.5, //SLIDER,[illegible]
    value = 1 -Math.pow(X/a1,2) - Math.pow(Y/a2,2) -Math.pow(Z/a3,2);
return value;
```

1. You need to create an account by registering on the platform and log in

2. Create a primitive by clicking the "add" button.

3. Select “primitive".

4. Enter the name of the primitive, click ok.

5. Show the “code window”

6. Write the code

7. Update the model

8. Watch the result

9. Save the code

10. Open the Settings

Geometric modeling using IZICAD

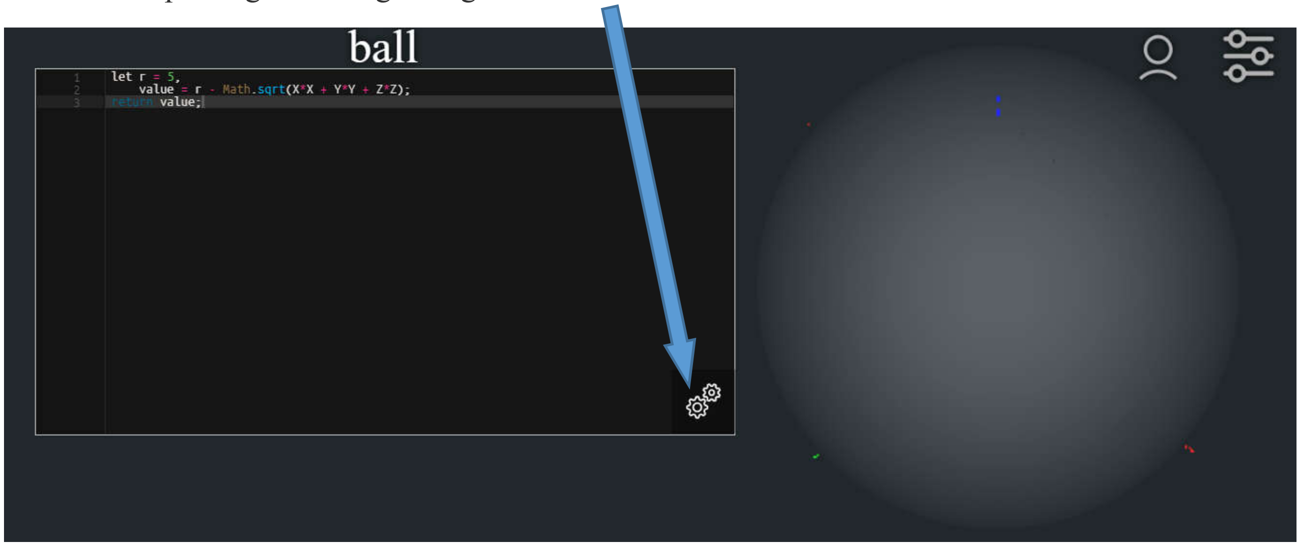
We begin with a simple sphere. The program “ball” describes a solid sphere:

$$5^2 - \left(x^2 + y^2 + z^2\right) \geqslant 0$$

```
// ball

let r = 5,

    value = r - Math.sqrt(X*X + Y*Y + Z*Z);

return value;
```

To see an object, the program has to be interpreted by visualization tools. We use the **IZICAD** that visualizes F-rep objects. The modeling language is javascript.

The following window will be shown, and you can see the resulting rendered object. You can rotate an object with the right mouse button and translate with the right mouse button, or scale with mouse wheel. For updating rendering the "gear icon" should be used.



```
let r = 5,
    value = r - Math.sqrt(X*X + Y*Y + Z*Z);
return value;
```

Let's see the program step by step.

```
line 1: // ball primitive

line 2: let r = 5,

line 3: value = r - Math.sqrt(X*X + Y*Y + Z*Z);

line 4: return value;
```

Line 1: The first line is a comment. The section from – to the newline is assumed to be a comment. You can insert comments anywhere in a program.

Line 2: Declare variable "r"

Line 3: Declare variable "value" and set it the formula.

Math – is mathematical library in javascript.

Note that X,Y,Z are reserved key words in IZICAD, they are point.

Line 4: returns the result of the calculation.

Arithmetic operators in IZICAD	
+	addition
-	subtraction
*	multiplication
/	division
^	power

Set-theoretical operations in IZICAD	
Math.min (A,B)	Intersection (A and B)
Math.max (A,B)	Union (A and B)
Math.min(A,-B)	Subtraction (A \ B)

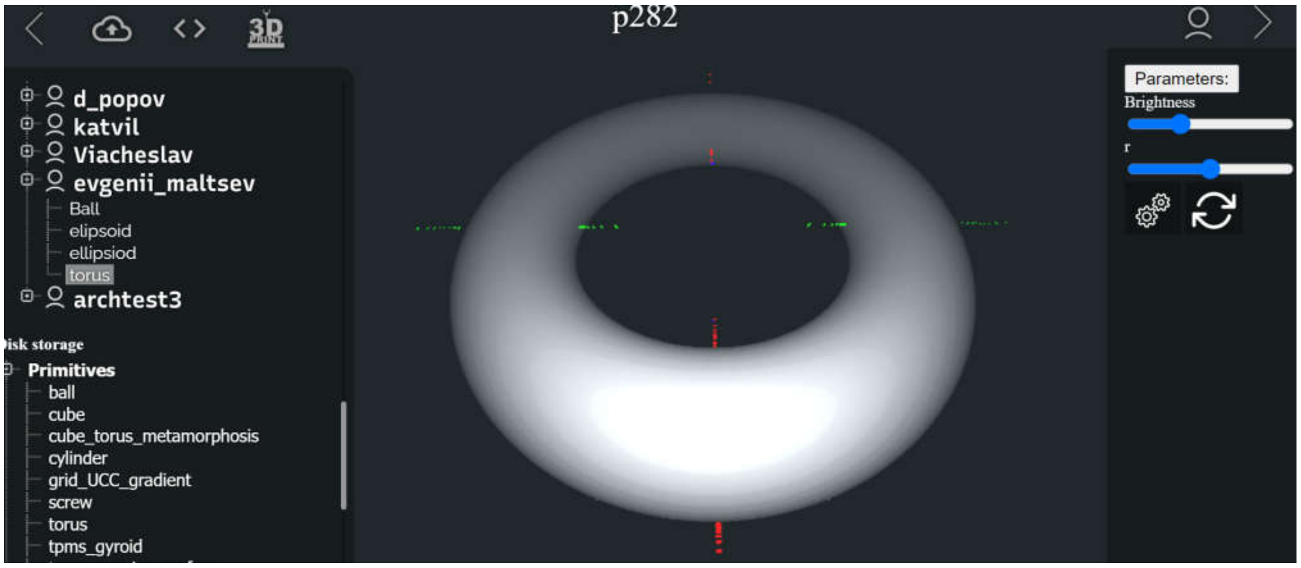
Tasks:

Modelling Implicit Surfaces Using Equations

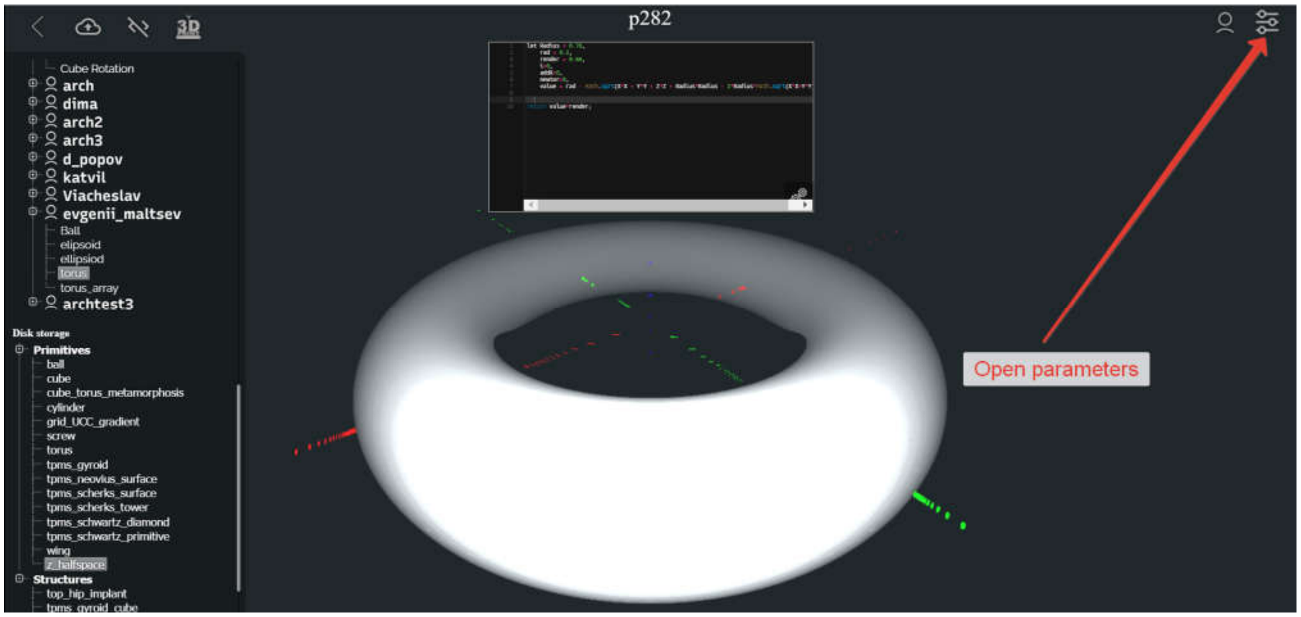
Let's create a torus:

1. Type the following model of an torus primitive:

```
let Radius = 0.76,
            rad = 0.2,
            render = 0.66,
   value = rad - Math.sqrt(X*X + Y*Y + Z*Z + Radius*Radius - 2*Radius*Math.sqrt(X*X+Y*Y));
return value*render;
```

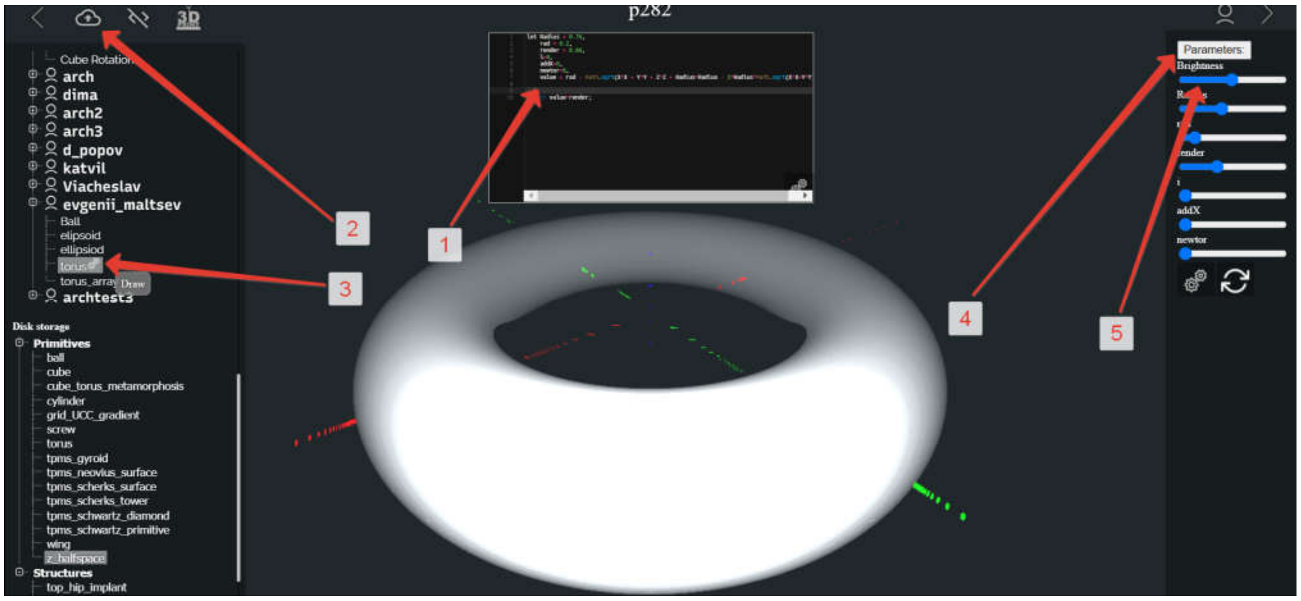


You can change parameter:



```
let Radius = 0.75,
    rad = 0.2,
    render = 0.04,
    t=0,
    addX=0,
    newton=0,
    value = rad - Math.sqrt(X*X + Y*Y + Z*Z + Radius*Radius - 2*Radius*Math.sqrt(X*X+Y*Y [illegible]

return value+render;
```

For updating parameters of model, you need to save your model (2), and push “draw” (3), and parameters will be added and updated automatic (4). You will able to control the Brightness (5).



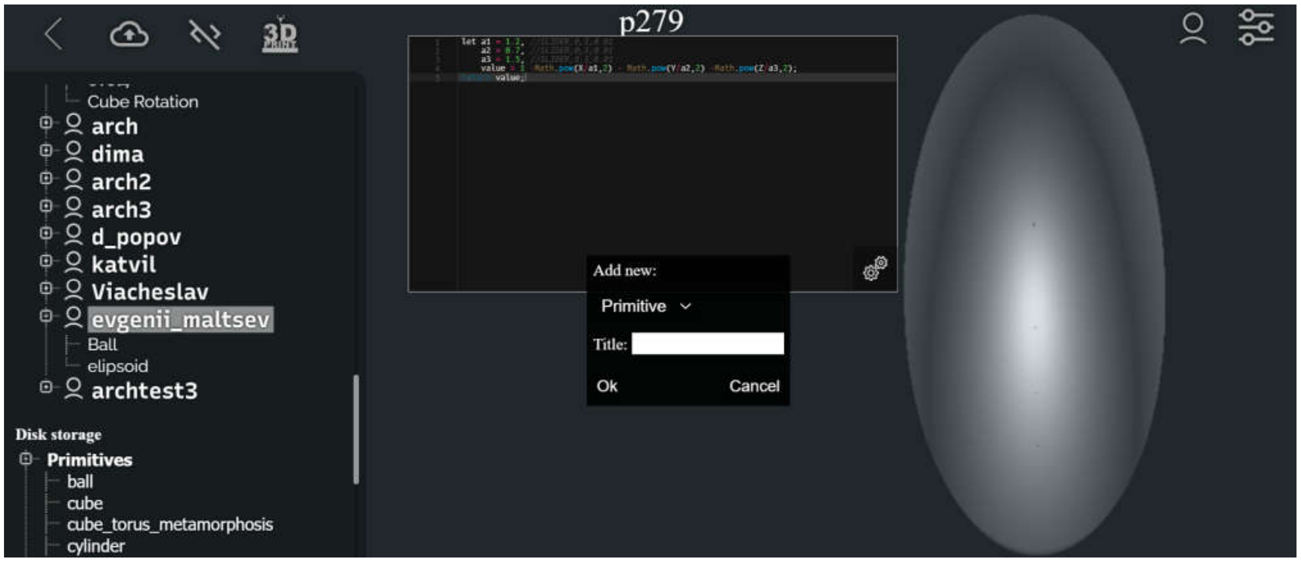
```
let Radius = 0.75,
    rad = 0.2,
    render = 0.06,
    t=0,
    addX=0,
    newtor=0,
    value = rad - Math.sqrt(X*X + Y*Y + Z*Z + Radius*Radius - 2*Radius*Math.sqrt(X*X+Y*Y

    value=render;
```

1. In the text editor type the following model of an ellipsoid:

```
let   value = 1-Math.pow(X/1.2,2) - Math.pow(Y/0.7,2) -Math.pow(Z/1.5,2)

return value;
```

2. Type the model of an ellipsoid with parameters:

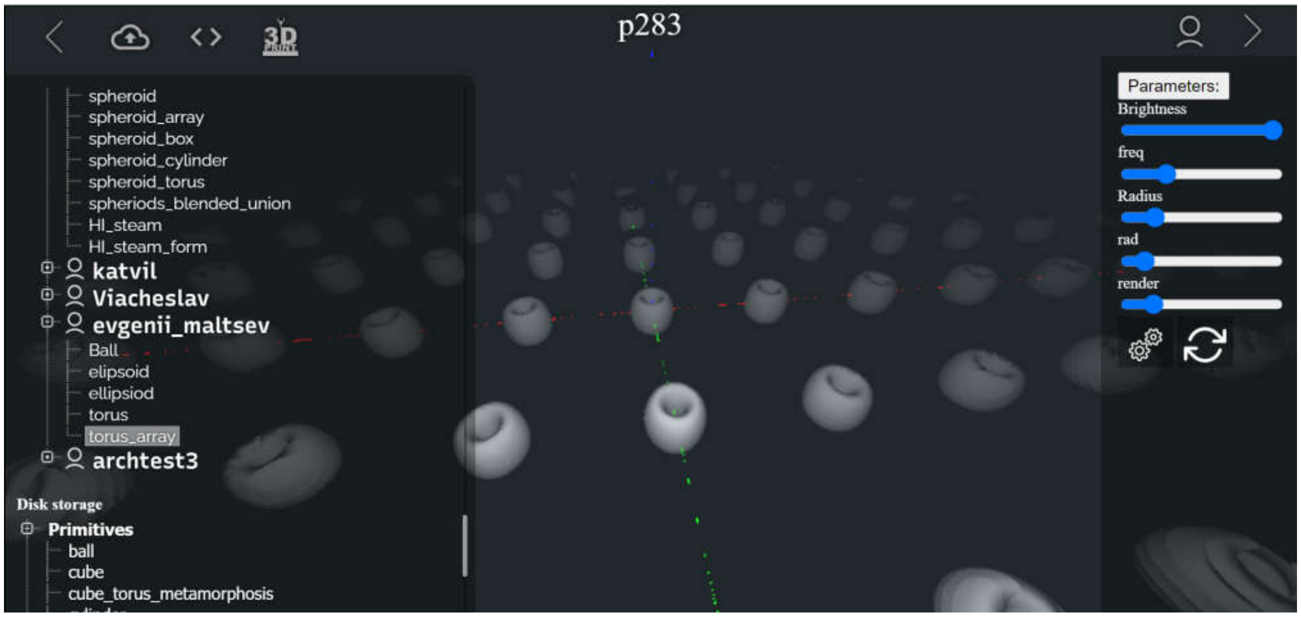
```
// for sliders with parameters, spaces between variables and values are important!

let a1 = 1.2, //SLIDER;0;5;0.01

    a2 = 0.7, //SLIDER;0;5;0.01

    a3 = 1.5, //SLIDER;0;5;0.01

    value = 1 -Math.pow(X/a1,2) - Math.pow(Y/a2,2) -Math.pow(Z/a3,2);

return value;
```



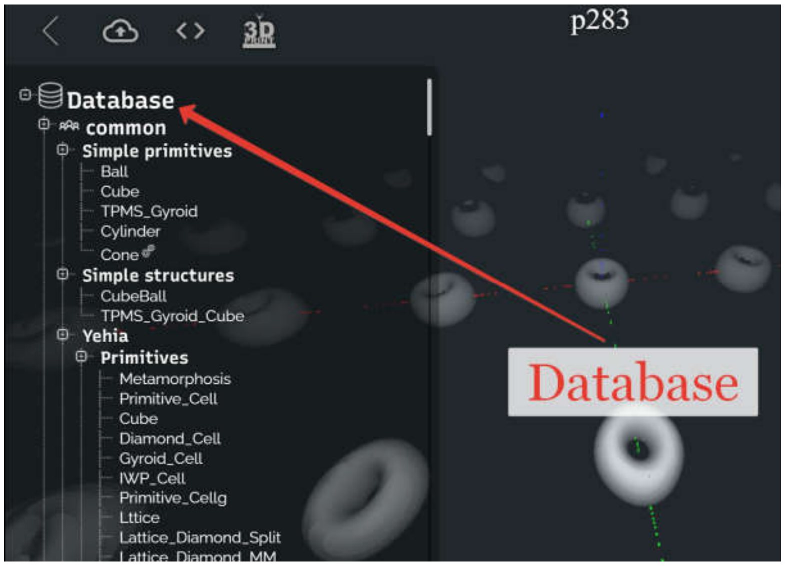
```
let a1 = 1.2, //[illegible]
    a2 = 0.7, //[illegible]
    a3 = 1.5, //[illegible]
    value = 1 -Math.pow(X/a1,2) - Math.pow(Y/a2,2) -Math.pow(Z/a3,2);
return value;
```

Let’s introduce the space mapping into the torus model by addition sinus space mapping for coordinates X and Y.

```
let freq = 1.77,

   new_X = Math.sin(freq*X), // SPACE MAPPING

   new_Y = Math.sin(freq*Y), // SPACE MAPPING

   Radius = 0.34,

            rad = 0.2,

            render = 0.66,


value = rad - Math.sqrt(new_X*new_X + new_Y*new_Y + Z*Z + Radius*Radius - 2*Radius*Math.sqrt(new_X*new_X+new_Y*new_Y));
```



Many examples of primitives and operations you can find in the Database or on the web-site https://hyperfun.org :

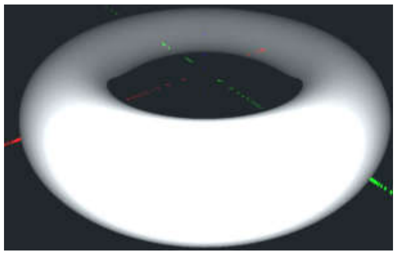


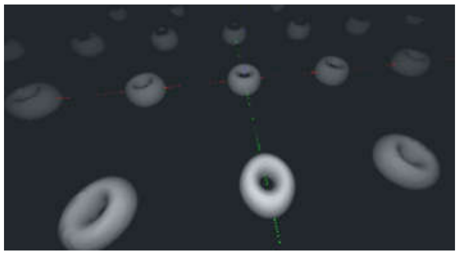
Individual tasks:

Create unit-cell and construct array of this elements using space mapping

Example:

Unit-cell: Array:





Variant 1:

Unit-cell: block (rectangle)

Variant 2:

Unit-cell: prizm (triangle)

Variant 3:

Unit-cell: Sphere

Variant 4:

Unit-cell: Ellipsoid

Variant 5:

Unit-cell: Torus

Variant 6:

Unit-cell: Cone