**4.31 多螺旋體加頭尾**

var Debug = Core.Debug;

var Mesh3D = Core.Mesh3D;

var Path2D = Core.Path2D;

var Plugin = Core.Plugin;

var Tess = Core.Tess;

var Solid = Core.Solid;

params = [

{ "id": "turn\_angle","displayName": "旋轉度數(度)","type": "float","rangeMin": 0,"rangeMax": 1000,"default": 540},

{ "id": "p","displayName": "z軸每360度升起距離","type": "float","rangeMin": 0,"rangeMax": 100,"default": 70 },

{ "id": "dd","displayName": "螺旋個數", "type": "int", "rangeMin": 1, "rangeMax": 5, "default": 2 },

{ "id": "x8\_pos","displayName": "x8 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 20},

{ "id": "z8\_pos","displayName": "z8 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 10 },

{ "id": "x7\_pos","displayName": "x7 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 23},

{ "id": "z7\_pos","displayName": "z7 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 5 },

{ "id": "x6\_pos","displayName": "x6 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 20},

{ "id": "z6\_pos","displayName": "z6 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 0 },

{ "id": "x5\_pos","displayName": "x5 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 20},

{ "id": "z5\_pos","displayName": "z5 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 0 },

{ "id": "x4\_pos","displayName": "x4 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 20},

{ "id": "z4\_pos","displayName": "z4 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 0 },

{ "id": "x3\_pos","displayName": "x3 position","type": "float","rangeMin": 0,"rangeMax": 100,"default":20},

{ "id": "z3\_pos","displayName": "z3 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 0},

{ "id": "x2\_pos","displayName": "x2 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 20},

{ "id": "z2\_pos","displayName": "z2 position","type": "float","rangeMin": 0,"rangeMax": 100,"default": 0 },

{ "id": "x1\_pos","displayName": "x1 position","type": "float","rangeMin": 0,"rangeMax": 100,"default":20},

{ "id": "z1\_pos","displayName": "z1 position","type": "float","rangeMin": 0,"rangeMax": 100,"default":0 },

{ "id": "sides","displayName": "成型步數", "type": "int", "rangeMin": 3, "rangeMax": 360, "default": 360 },

{ "id": "headLength", "displayName": "Head Length", "type": "int", "rangeMin": 3, "rangeMax": 60, "default": 20 },

{ "id": "endLength", "displayName": "End Length", "type": "int", "rangeMin": 3, "rangeMax": 60, "default": 20 },

{ "id": "x\_pos","displayName": "x position","type": "float","rangeMin": -100,"rangeMax": 100,"default": 0},

{ "id": "y\_pos","displayName": "y position","type": "float","rangeMin": -100,"rangeMax": 100,"default": 0 },

{ "id": "z\_pos","displayName": "z position","type": "float","rangeMin": -100,"rangeMax": 100,"default": 0}

];

function process(params) {

var turn\_angle=params.turn\_angle;

var p = params.p;

var dd=params.dd;

var x9\_pos = params.x1\_pos; //第九層取與第一層同

var z9\_pos = params.z1\_pos; //第九層取與第一層同

var x8\_pos = params.x8\_pos;

var z8\_pos = params.z8\_pos;

var x7\_pos = params.x7\_pos;

var z7\_pos = params.z7\_pos;

var x6\_pos = params.x6\_pos;

var z6\_pos = params.z6\_pos;

var x5\_pos = params.x5\_pos;

var z5\_pos = params.z5\_pos;

var x4\_pos = params.x4\_pos;

var z4\_pos = params.z4\_pos;

var x3\_pos = params.x3\_pos;

var z3\_pos = params.z3\_pos;

var x2\_pos = params.x2\_pos;

var z2\_pos = params.z2\_pos;

var x1\_pos = params.x1\_pos;

var z1\_pos = params.z1\_pos;

var headLength=params.headLength;

var endLength=params.endLength;

var x\_pos = params.x\_pos;

var y\_pos = params.y\_pos;

var z\_pos = params.z\_pos;

var tr9 = x9\_pos;

var tr8 = x8\_pos;

var tr7 = x7\_pos;

var tr6 = x6\_pos;

var tr5 = x5\_pos;

var tr4 = x4\_pos;

var tr3 = x3\_pos;

var tr2 = x2\_pos;

var tr1 = x1\_pos;

var sides = params.sides;

var angle = turn\_angle/360\*2\*Math.PI / sides; //每部旋轉角度

var mesh = new Mesh3D();

var side1=[];

var side2=[];

var side3=[];

var side4=[];

var side5=[];

var side6=[];

var side7=[];

var side8=[];

var side9=[];

for (var j = 0 ; j <dd ;j++) {

side1[j]=[];

side2[j]=[];

side3[j]=[];

side4[j]=[];

side5[j]=[];

side6[j]=[];

side7[j]=[];

side8[j]=[];

side9[j]=[];

}

var tz;

var tx1 ;

var ty1 ;

var tx2 ;

var ty2;

var tx3 ;

var ty3 ;

var tx4 ;

var ty4 ;

var tx5 ;

var ty5;

var tx6 ;

var ty6;

var tx7 ;

var ty7 ;

var tx8;

var ty8 ;

var tx9 ;

var ty9 ;

var tx11=[];

var ty11=[];

var theadz=[];

var sideHead=[];

var tx99=[];

var ty99=[];

var tendz=[];

var sideEnd=[];

var anglen=2\*Math.PI/dd;

for (var n=0; n< dd; n++)

{

for (var i =0; i < sides; i++) //設定各點座標

{

tz=i\*angle/2/Math.PI\*p; //z軸升起

tx1 = tr1 \* Math.cos(i \* angle+n\*anglen);

ty1 = tr1 \* Math.sin(i \* angle+n\*anglen);

tx2 = tr2 \* Math.cos(i \* angle+n\*anglen);

ty2 = tr2 \* Math.sin(i \* angle+n\*anglen);

tx3 = tr3 \* Math.cos(i \* angle+n\*anglen);

ty3 = tr3 \* Math.sin(i \* angle+n\*anglen);

tx4 = tr4 \* Math.cos(i \* angle+n\*anglen);

ty4 = tr4 \* Math.sin(i \* angle+n\*anglen);

tx5 = tr5 \* Math.cos(i \* angle+n\*anglen);

ty5 = tr5 \* Math.sin(i \* angle+n\*anglen);

tx6 = tr6 \* Math.cos(i \* angle+n\*anglen);

ty6 = tr6 \* Math.sin(i \* angle+n\*anglen);

tx7 = tr7 \* Math.cos(i \* angle+n\*anglen);

ty7 = tr7 \* Math.sin(i \* angle+n\*anglen);

tx8 = tr8 \* Math.cos(i \* angle+n\*anglen);

ty8 = tr8 \* Math.sin(i \* angle+n\*anglen);

tx9 = tr9 \* Math.cos(i \* angle+n\*anglen);

ty9 = tr9 \* Math.sin(i \* angle+n\*anglen);

side1[n][i]=([x\_pos+tx1, y\_pos+ty1, z\_pos+z1\_pos+tz]);

side2[n][i]=([x\_pos+tx2, y\_pos+ty2, z\_pos+z2\_pos+tz]);

side3[n][i]=([x\_pos+tx3, y\_pos+ty3, z\_pos+z3\_pos+tz]);

side4[n][i]=([x\_pos+tx4, y\_pos+ty4, z\_pos+z4\_pos+tz]);

side5[n][i]=([x\_pos+tx5, y\_pos+ty5, z\_pos+z5\_pos+tz]);

side6[n][i]=([x\_pos+tx6, y\_pos+ty6, z\_pos+z6\_pos+tz]);

side7[n][i]=([x\_pos+tx7, y\_pos+ty7, z\_pos+z7\_pos+tz]);

side8[n][i]=([x\_pos+tx8, y\_pos+ty8, z\_pos+z8\_pos+tz]);

side9[n][i]=([x\_pos+tx9, y\_pos+ty9, z\_pos+z9\_pos+tz]);

}

tx11[n] = tr1 \* Math.cos(((-1)\*headLength) \* angle+n\*anglen);

ty11[n] = tr1 \* Math.sin(((-1)\*headLength) \* angle+n\*anglen);

theadz[n]=0;

sideHead[n]=[x\_pos+tx11[n], y\_pos+ty11[n], z\_pos+z9\_pos+theadz[n]];

tx99[n] = tr9 \* Math.cos((sides+endLength) \* angle+n\*anglen);

ty99[n] = tr9 \* Math.sin((sides+endLength) \* angle+n\*anglen);

tendz[n]=(sides-1) \*angle/2/Math.PI\*p;

sideEnd[n]=[x\_pos+tx99[n], y\_pos+ty99[n], z\_pos+z9\_pos+tendz[n]];

for ( i = 0; i < sides-1; i++) //用四邊形掃邊(不包含最後封口)

{

mesh.quad(side1[n][i], side1[n][i+1],side2[n][i+1],side2[n][i]);

mesh.quad(side2[n][i], side2[n][i+1],side3[n][i+1],side3[n][i]);

mesh.quad(side3[n][i], side3[n][i+1],side4[n][i+1],side4[n][i]);

mesh.quad(side4[n][i], side4[n][i+1],side5[n][i+1],side5[n][i]);

mesh.quad(side5[n][i], side5[n][i+1],side6[n][i+1],side6[n][i]);

mesh.quad(side6[n][i], side6[n][i+1],side7[n][i+1],side7[n][i]);

mesh.quad(side7[n][i], side7[n][i+1],side8[n][i+1],side8[n][i]);

mesh.quad(side8[n][i], side8[n][i+1],side9[n][i+1],side9[n][i]);

}

//起始掃邊封口

mesh.triangle( sideHead[n],side1[n][0], side2[n][0]);

mesh.triangle( sideHead[n],side2[n][0], side3[n][0]);

mesh.triangle( sideHead[n],side3[n][0], side4[n][0]);

mesh.triangle( sideHead[n],side4[n][0], side5[n][0]);

mesh.triangle( sideHead[n],side5[n][0], side6[n][0]);

mesh.triangle( sideHead[n],side6[n][0], side7[n][0]);

mesh.triangle( sideHead[n],side7[n][0], side8[n][0]);

mesh.triangle( sideHead[n],side8[n][0], side9[n][0]);

//最後掃邊封口

mesh.triangle( sideEnd[n],side9[n][sides-1], side8[n][sides-1]);

mesh.triangle( sideEnd[n],side8[n][sides-1], side7[n][sides-1]);

mesh.triangle( sideEnd[n],side7[n][sides-1], side6[n][sides-1]);

mesh.triangle( sideEnd[n],side6[n][sides-1], side5[n][sides-1]);

mesh.triangle( sideEnd[n],side5[n][sides-1], side4[n][sides-1]);

mesh.triangle( sideEnd[n],side4[n][sides-1], side3[n][sides-1]);

mesh.triangle( sideEnd[n],side3[n][sides-1], side2[n][sides-1]);

mesh.triangle( sideEnd[n],side2[n][sides-1], side1[n][sides-1]);

}

return Solid.make(mesh);

**}**