**GROUP ONE**
INTRO TO COMPUTER GRAPHICS
PACSSR - **301055**

# WELCOME TO

# PRESENTATION

**by GROUP ONE**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

Group One

BSN

BIOGRAPHY SECTION

## STRUCTURAL PRESENTATION

# CG CTRL

SPACE STYLE DRAGON SCENE ANIMATION


**Frankel Zhao** — **Young**

**Kevin Zhen** — **Jack Zhang**


**SELECTING FILE**

POWERED BY **MAGIC CG**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

PERMISSION AUTHORIZED

**GROUP ONE**

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

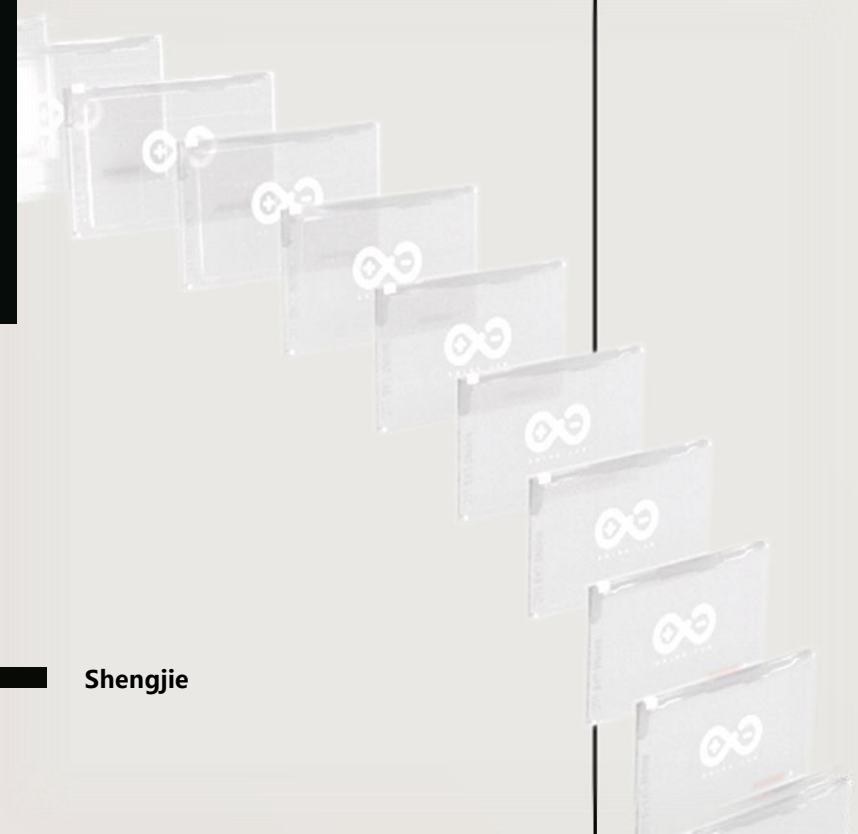
SELECTION

## CONTENTS

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



SECTION 1

# Intro

DESING EXPRESSING SECTION

**Shengjie**

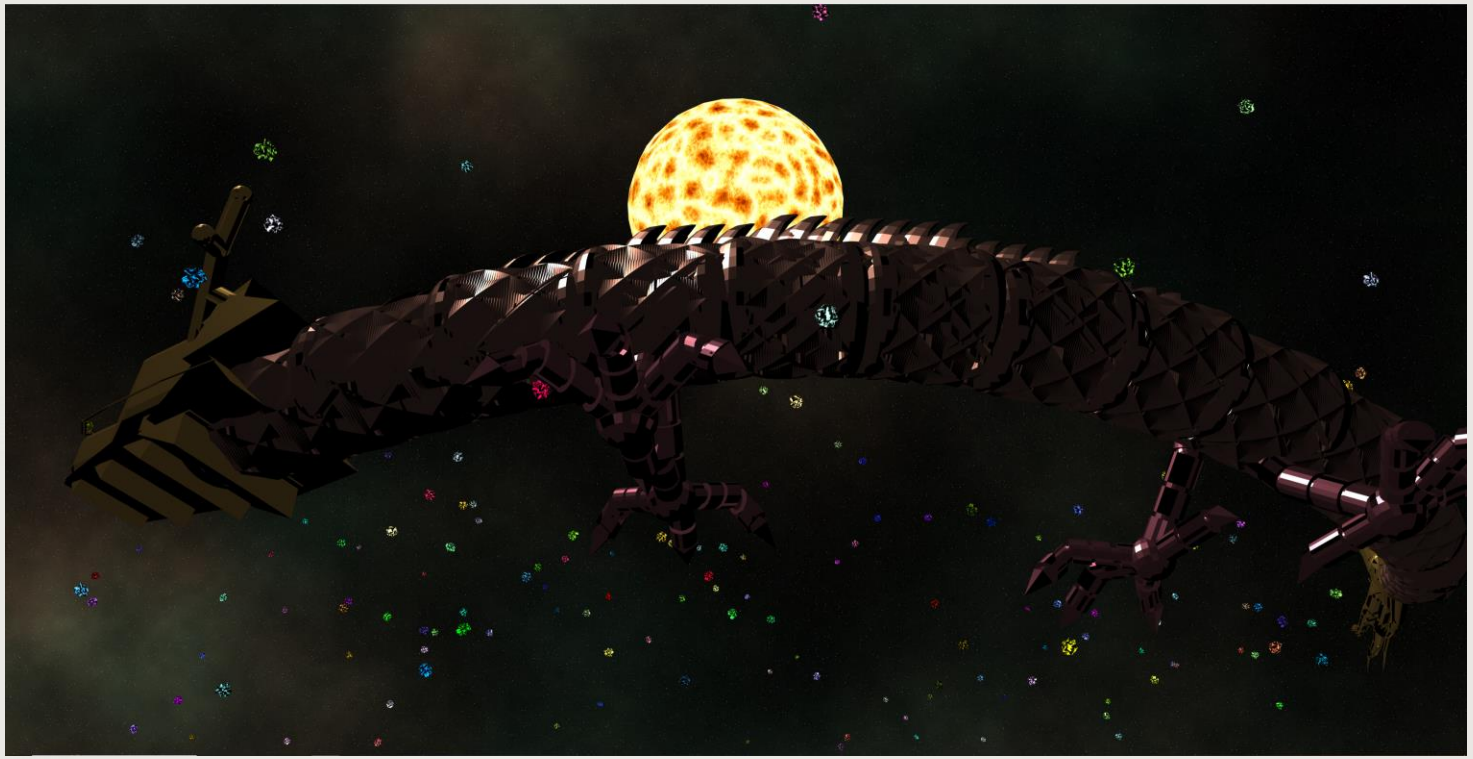
**Shengjie**

POWERED BY **MAGIC CG**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



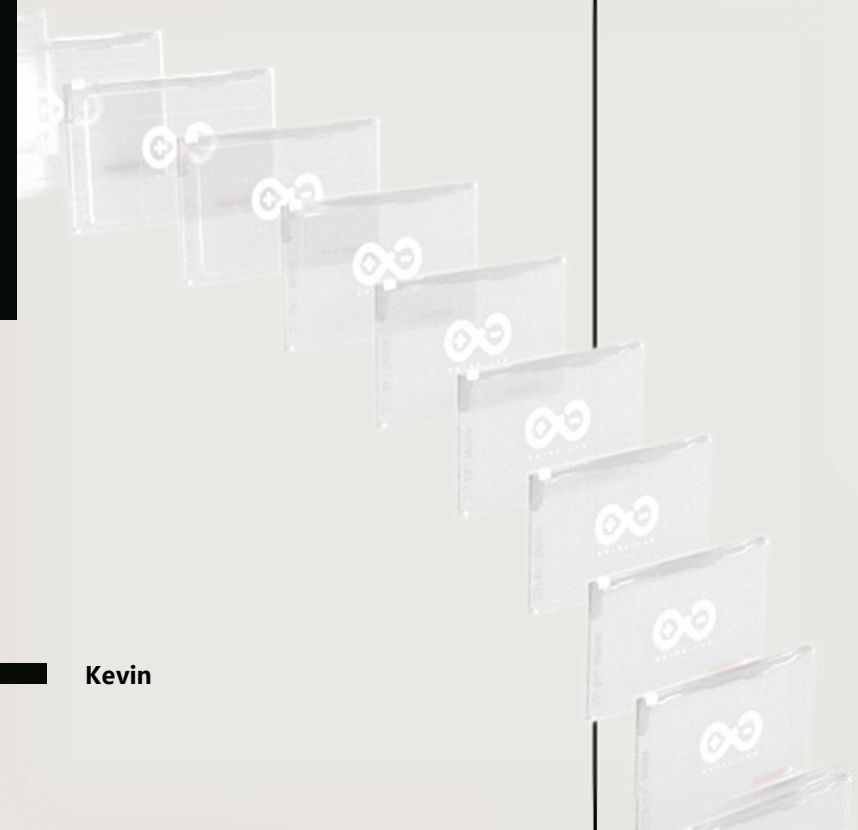
The divine Chinese dragon guarding the Sun in the universe.

- OPENJSCAD
- THREE.JS
- SHADERFROG
- LATEX

**GROUP ONE**

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



**Kevin**

SECTION 2

# Model

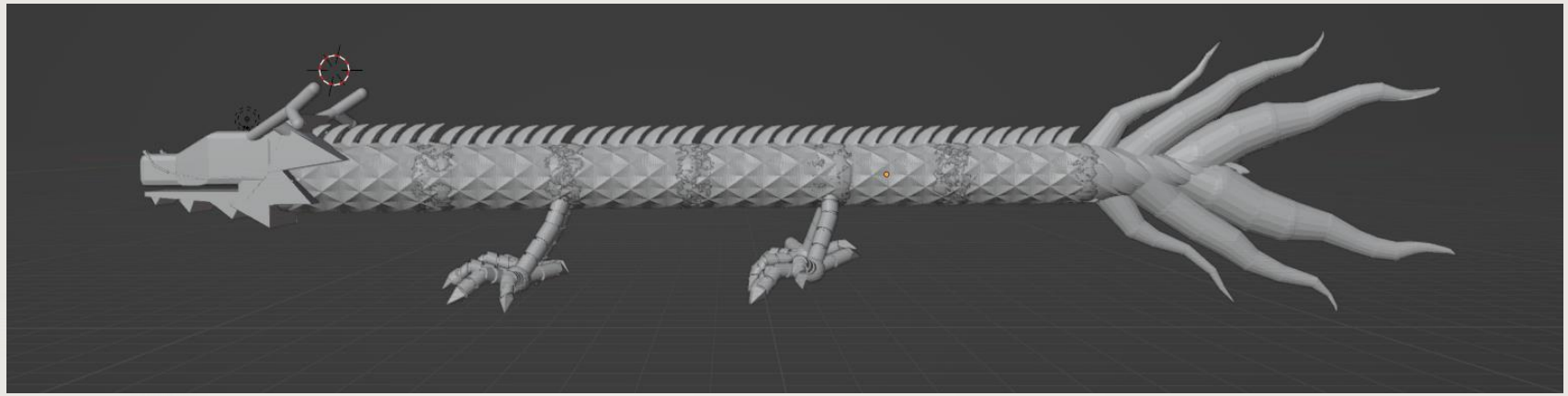
DESING REALIZATION SECTION 1

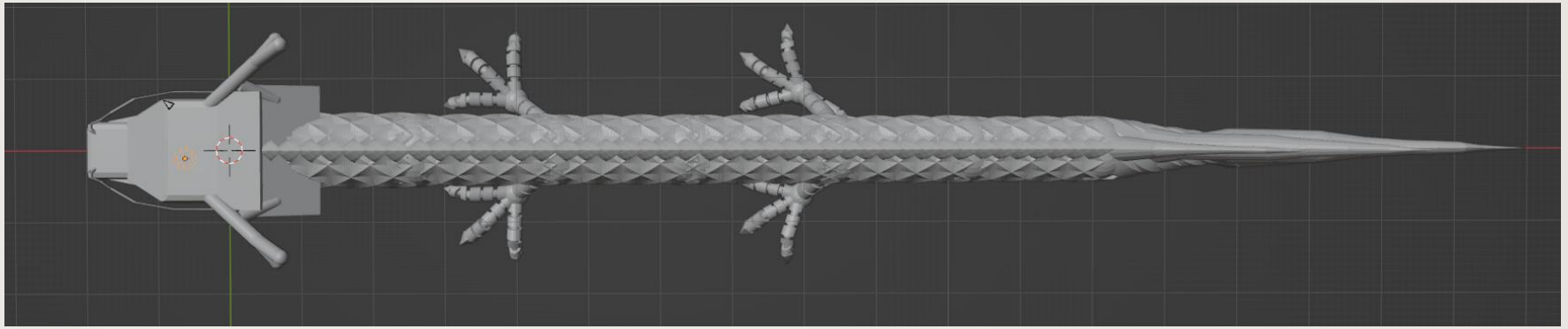
**Kevin**

POWERED BY **MAGIC CG**

# Modeling (in OpenJsCAD)

unrendered model in Blender

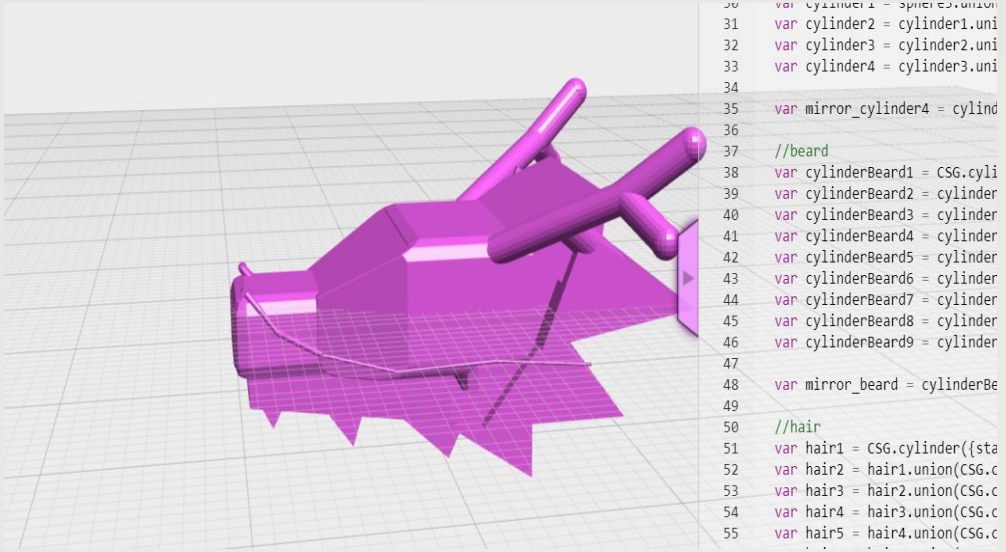




# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



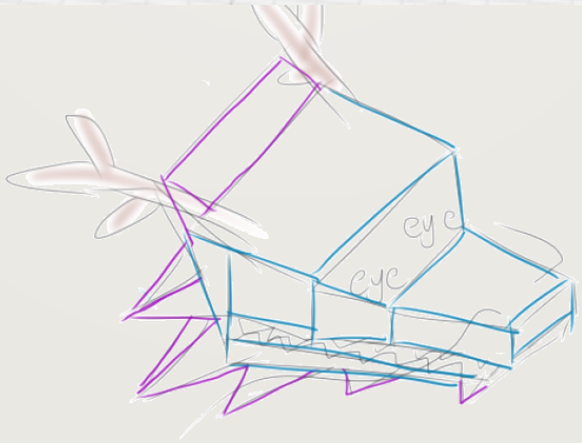
```
31  var cylinder2 = cylinder1.uni
32  var cylinder3 = cylinder2.uni
33  var cylinder4 = cylinder3.uni
34
35  var mirror_cylinder4 = cylind
36
37  //beard
38  var cylinderBeard1 = CSG.cyli
39  var cylinderBeard2 = cylinder
40  var cylinderBeard3 = cylinder
41  var cylinderBeard4 = cylinder
42  var cylinderBeard5 = cylinder
43  var cylinderBeard6 = cylinder
44  var cylinderBeard7 = cylinder
45  var cylinderBeard8 = cylinder
46  var cylinderBeard9 = cylinder
47
48  var mirror_beard = cylinderBe
49
50  //hair
51  var hair1 = CSG.cylinder({sta
52  var hair2 = hair1.union(CSG.c
53  var hair3 = hair2.union(CSG.c
54  var hair4 = hair3.union(CSG.c
55  var hair5 = hair4.union(CSG.c
```

## Dragon head

```
//mouth
var cubeMouth_1 = CSG.roundedCube({center: [4, 0, 0], radius: [4,7,4], roundradius: 1, resolution: resolution});
```

```
//hair
var hair1 = CSG.cylinder({start: [-3, 8, 4], end: [-3, -8, 4], radius: 4, resolution: 3}).rotateY(-10);
```

```
//horn
var sphere11 = CSG.sphere({center: [-10, -14, 10], radius: 1, resolution: 20});
```




hair

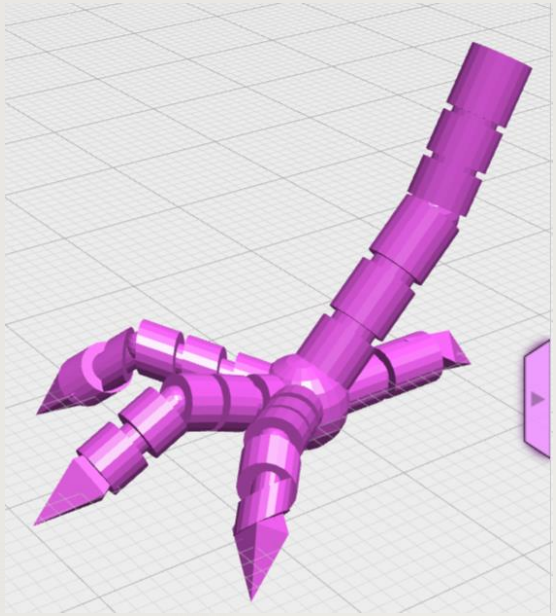
mouth

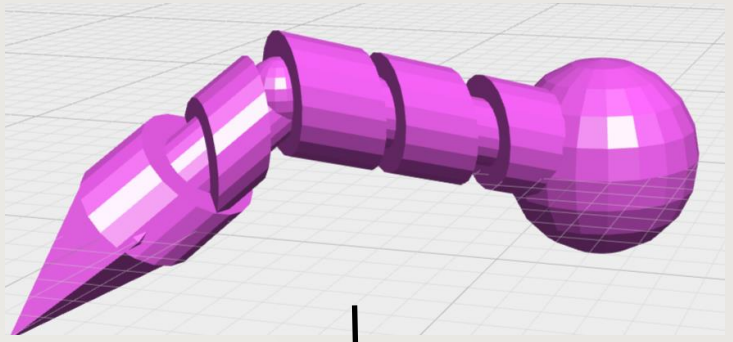
bone

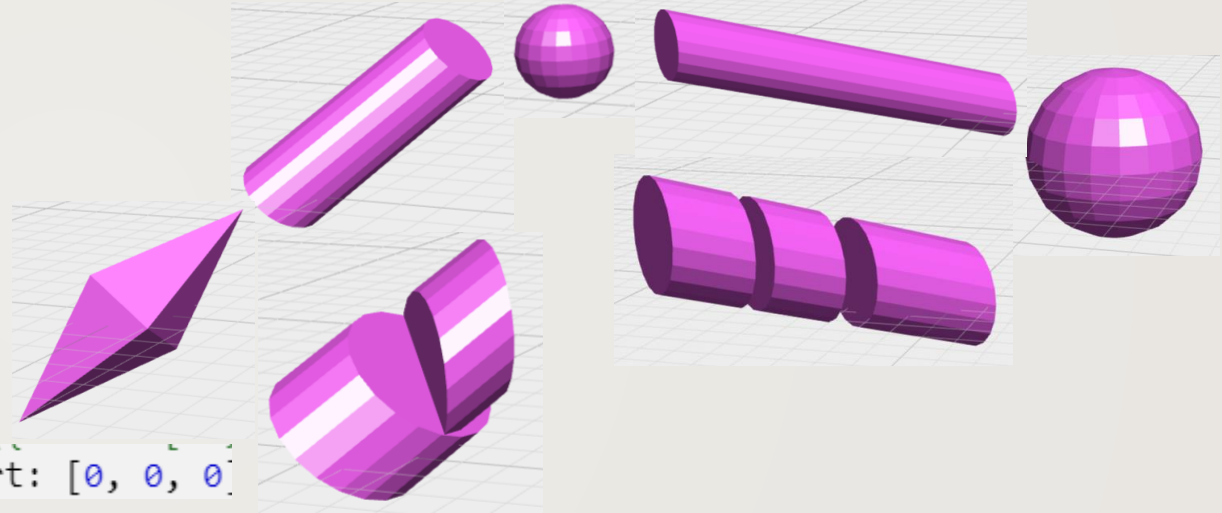
# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**







Dragon claw

```
var f1 = CSG.cylinder({start: [0, 0, 0
```

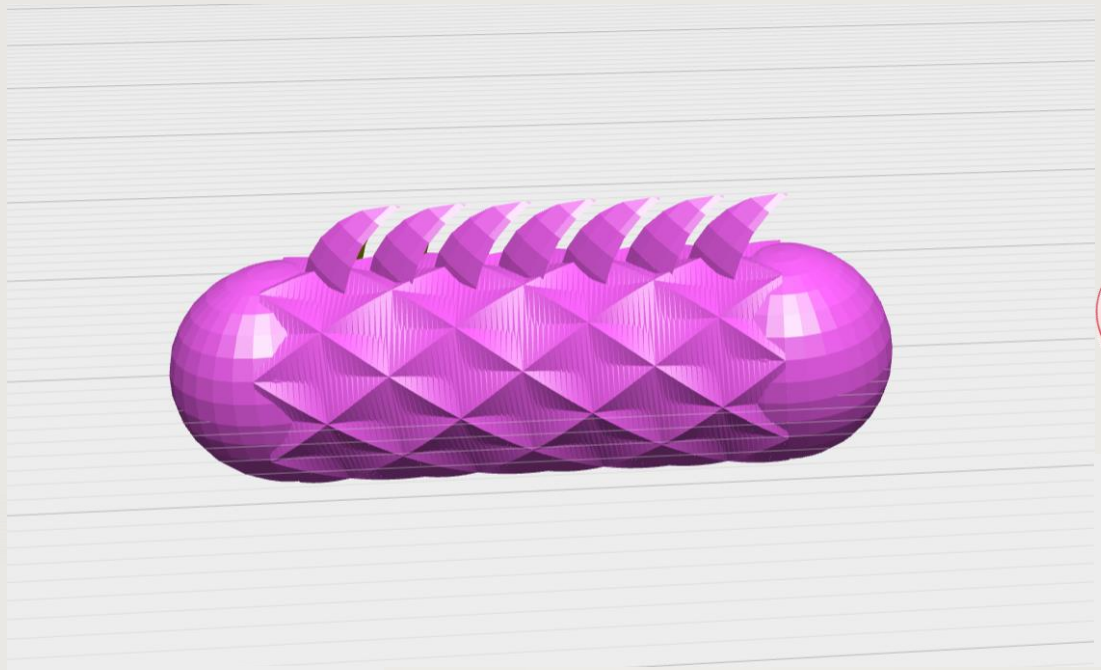
```
var s1 = CSG.sphere({center: [7, -7, 3], radius: 1, resolution: 20});
```

```
var z3 =
CSG.sphere({center: [10, -10, 0], radius: 1.3, resolution: 3})
.translate([-10, 10, 0])
.scale([1.2, 1.2, 3])
.rotateX(-50)
.rotateY(0)
.rotateZ(45)
.translate([10, -10, 0]);
```

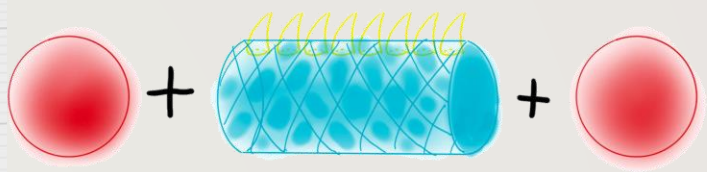
# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

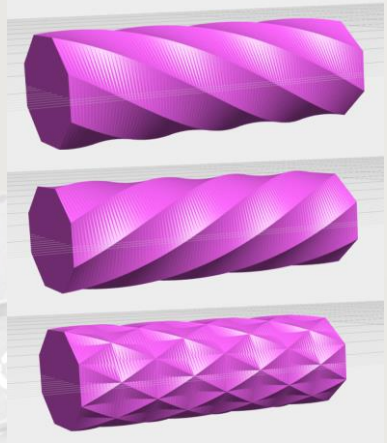


## Dragon body



## Dorsal Fin

```
s1 = translate([0, radius/2 * 1.2, 0], sphere({r:radius, center: true}));
s2 = translate([0, -radius/2 * 1.2, 0], sphere({r:radius, center: true}));
s3 = translate([radius/4 * 2 ,0, radius/6], sphere({r:radius, center: true}));
s4 = translate([0 ,0,-radius], cube({size: [radius * 2, radius , radius], center: true}));
inter_1 = translate([0,-10, 3],rotate([0,30,90], difference(difference(intersection(s1,s2), s3), s4)));
```



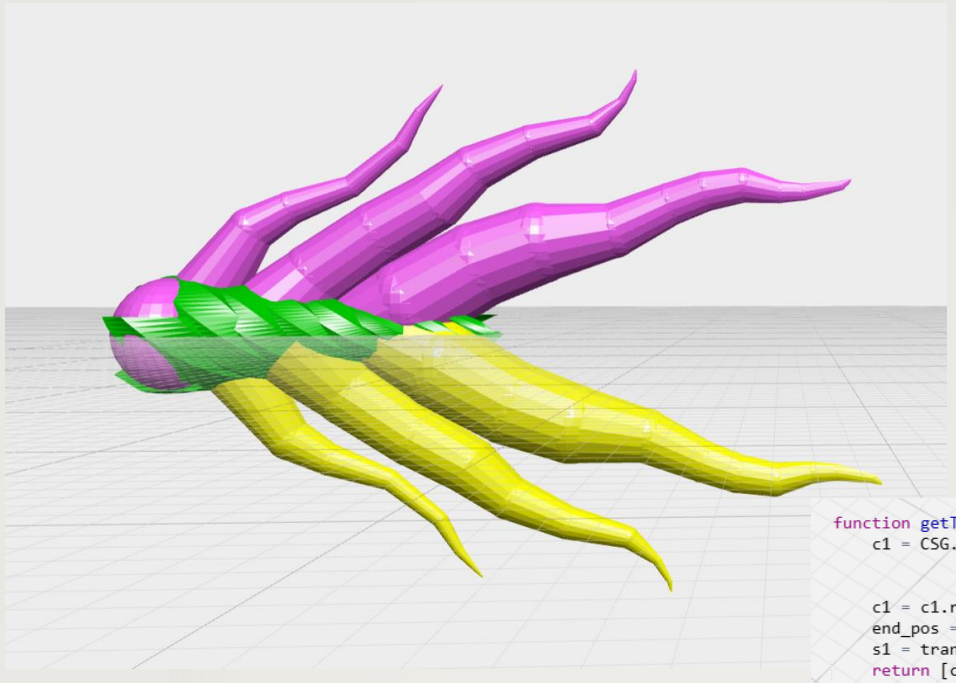
## Body

```
rotate([90,0,0],
    linear_extrude(
        {height: 20, twist: -180, slices : 100, center: true},
        cs3
        )
    ),
```

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

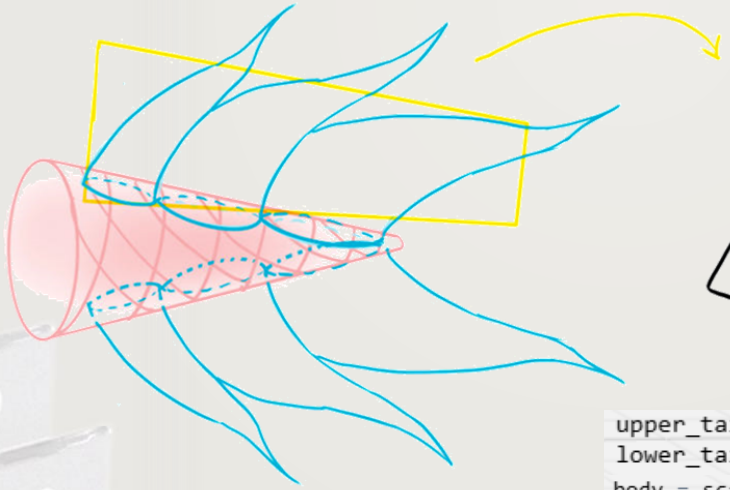


```
c4 = getTailPart([0,0,0], 15, 6,4, 35);
c5 = getTailPart(c4[2], 7, 4, 3, 60);
c6 = getTailPart(c5[2], 10, 3,2 , 70);
c7 = getTailPart(c6[2], 10, 2 , 1.5, 50);
c8 = getTailPart(c7[2], 6, 1.5, 1, 30);
c9 = getTailPart(c8[2], 6, 1, 0.3, 40)

part_side_1 = union(
    c4[0],c4[1],
    c5[0],c5[1],
    c6[0],c6[1],
    c7[0],c7[1],
    c8[0],c8[1],
    c9[0]
    );
part_side_2 = rotate([0,0,5] , translate([0,4,0], part_side_1));
tail1 = intersection(part_side_1, part_side_2);
tail1 = translate([-3,-2,4], rotate([0,0,-3], tail1));
```

```
function getTailPart( startpos, height, r1, r2, angle ){
    c1 = CSG.cylinderElliptic ({start:  startpos , end: [startpos[0], startpos[1] , startpos[2] +height],
                                radiusStart: r1, radiusEnd: r2, resolution: 20, center: startpos});

    c1 = c1.rotate(startpos,[0,1,0],-angle);
    end_pos = [startpos[0] - height * cos(90-angle), startpos[1] , startpos[2] + height * sin(90-angle)] ;
    s1 = translate(end_pos , sphere({r: r2, fn: 32 }));
    return [c1,s1, end_pos];
}
```



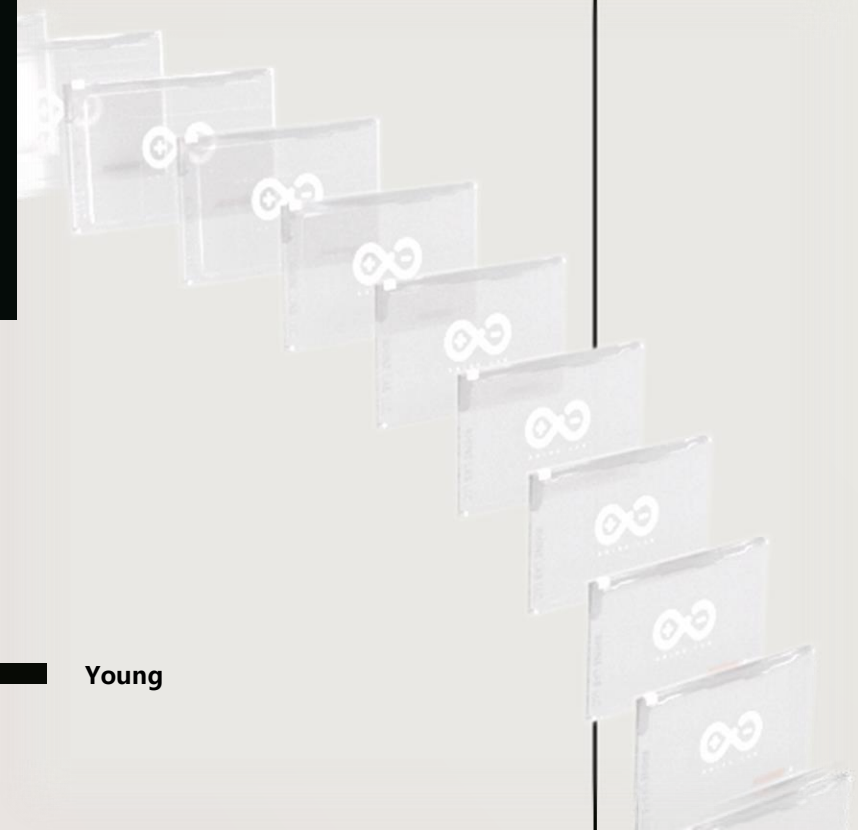
## Dragon tail

```
upper_tail = union(tail1, tail2, tail3);
lower_tail = translate([-5,0,0], rotate([180,0,0], upper_tail));
body = scale([0.8,0.3,0.3], translate([0,20,0], rotate([0,0,180], union(m1, m2, m3, m4))));
tail = union(body, upper_tail, lower_tail);
```

**GROUP ONE**

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



**Young**

SECTION 3

# Texture

DESING REALIZATION SECTION 2

**Young**

POWERED BY **MAGIC CG**

We used a BRDF shader as fragment shader(texture) to interact with light more obviously

```
////// Gathering Pieces
vec3 finalColor = kAmbi*ambient + kDiff*diffuse + kSpec*spec;


/* ================================================
 *   Return value
 * ================================================ */
gl_FragColor = vec4( finalColor, 1.0 );
```

It applied fresnel function to apply metal texture

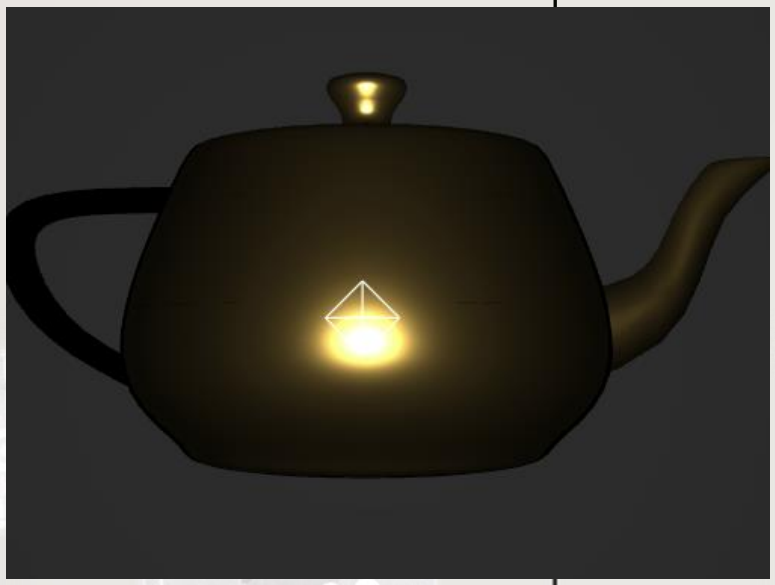
```
// Fresnel
vec3 fresnel = selectedMaterial + (vec3(1.0)-selectedMaterial) * pow( (1.0-ldh), 5.0 );
```

And modified material parameter to apply on three part of our model

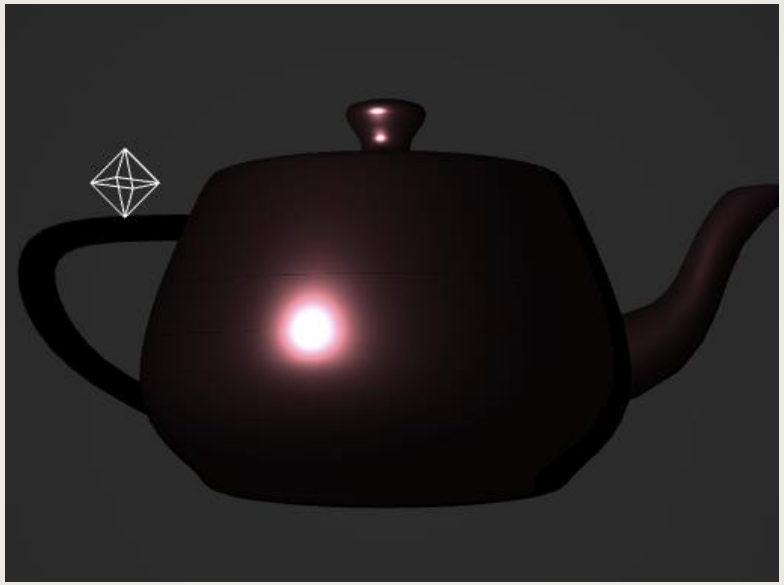
# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

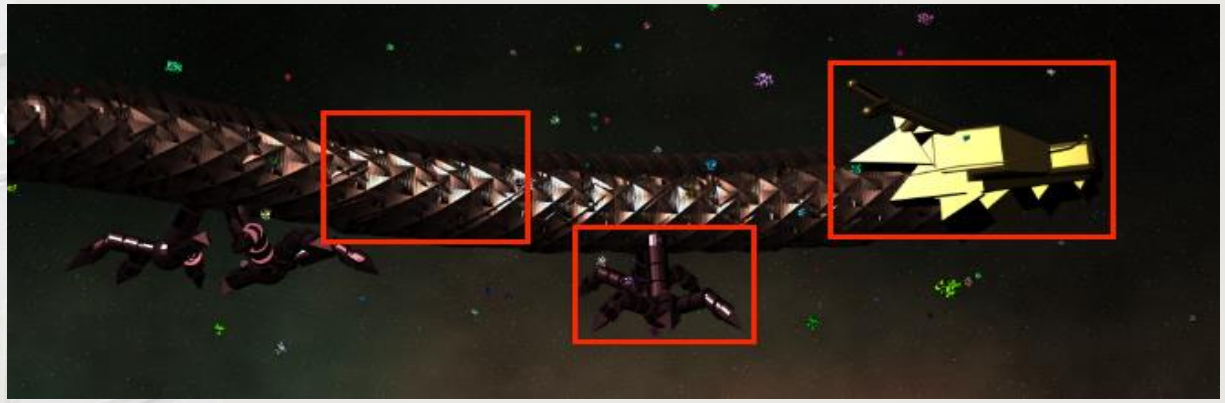


Gold to head



Tungsten to claw

Copper to body

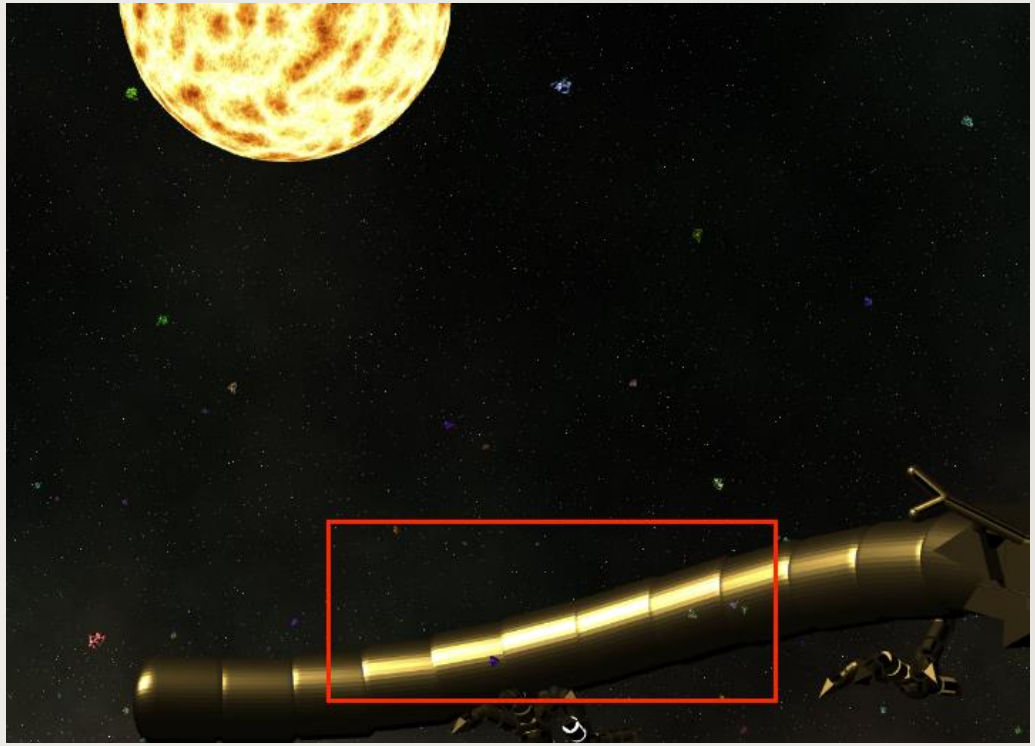


# Modifying the dynamic light positon to static light in our scene

MeronSoda_s_BRDF.json

F: > localGit > MomentaryRainY.github.io > assets > threeJs > {} MeronSoda_s_BRDF.json > {} uniforms > {} spe

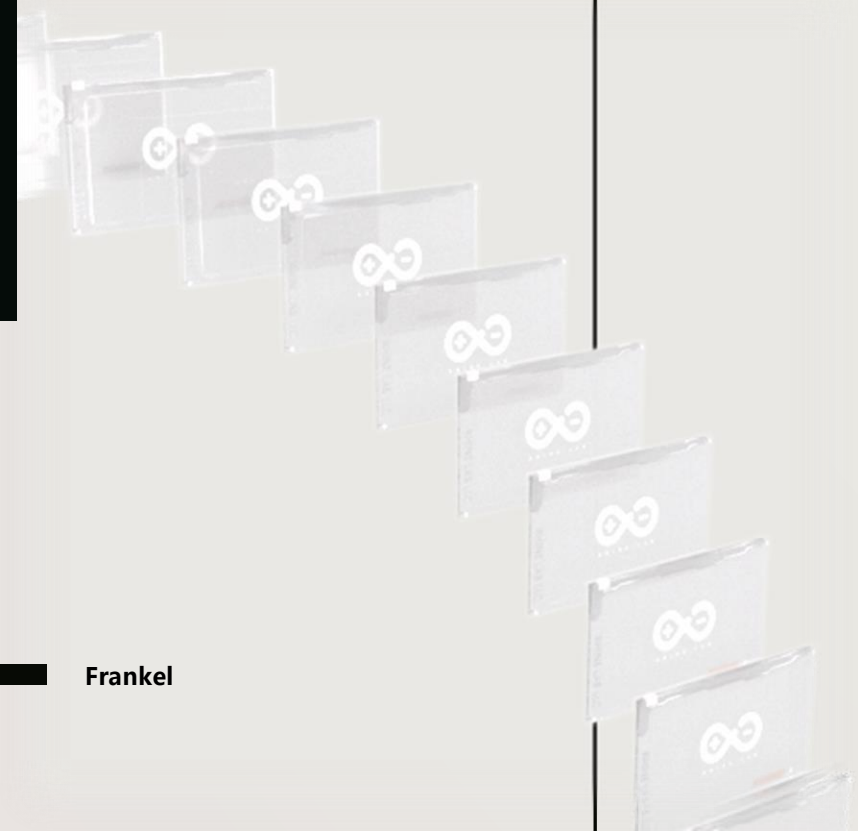
```
{
  "id": 1418,
  "name": "MeronSoda's BRDF",
  "fragment": "precision highp float;\n\nuniform mat4 modelMatrix;\nuniform mat
  "vertex": "precision highp float;\nprecision highp int;\n\nuniform mat4 model
  "uniforms": {
    "lightPosition": {
      "name": "lightPosition",
      "value": {
        "x": 3,
        "y": 20,
        "z": 20
      },
      "displayName": null,
      "type": "v3",
      "glslType": "vec3",
      "useGridHelper": false,
      "useRange": false,
      "range": null,
```



**GROUP ONE**

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**



SECTION 4

# Animate

DESING REALIZATION SECTION 3

**Frankel**

**Frankel**

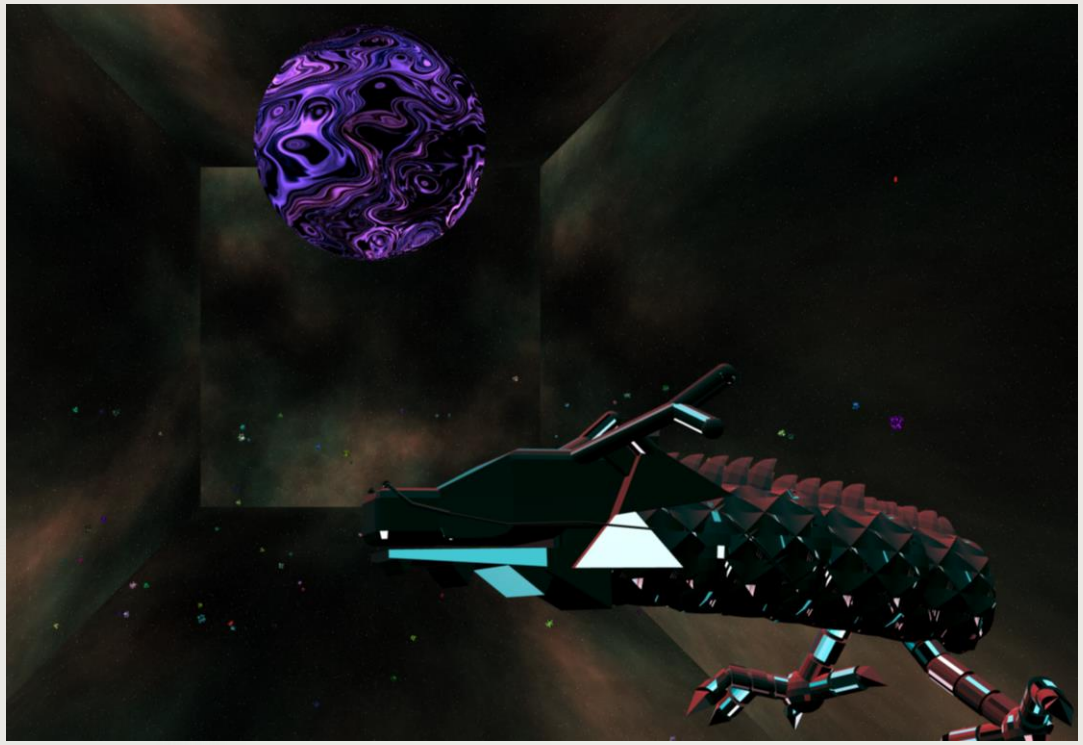
POWERED BY **MAGIC CG**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

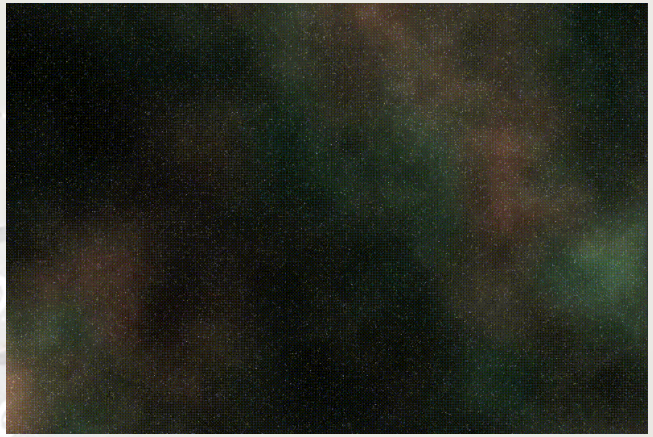
PACSSR - **301055**

## Bonding Box

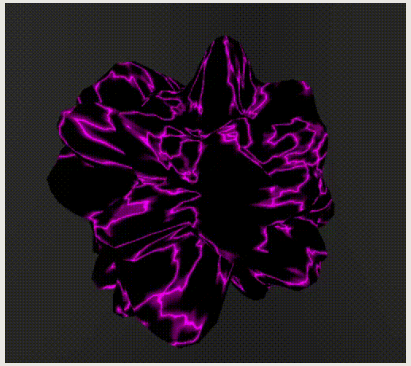


## Fluid Analogy

## Star Field



## Cosmic Dash



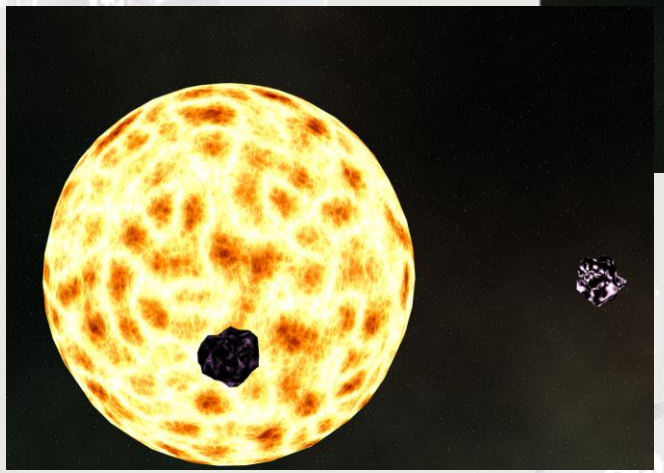
```
"name": "200",
"fragment": "#define PI 3.14159265358979323846264338327 9\n\n
"vertex": "precision highp float;\nprecision highp int;\nuni
    "gl_Position = Big_Wiggles1542394934505_111_main() + \n"
    "Caustic_Image_Based1542394968990_145_main(); }\n",
"uniforms": {
```

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

## Cosmic Dash generation



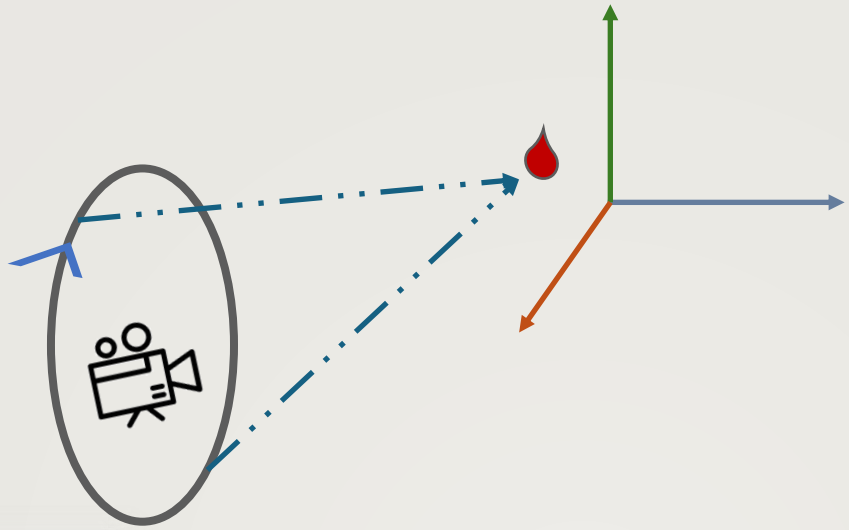
```
var meshBalls = [];
for (let id = 0; id<200;id++)
{
var ball = new THREE.SphereGeometry( 0.03+0.03*Math.random(), 20, 20 );
meshBalls.push( new THREE.Mesh( ball ) );
meshBalls[id].position.x = -20+40*Math.random();
meshBalls[id].position.z = -20+40*Math.random();
meshBalls[id].position.y = -4+8*Math.random()-5;
meshBalls[id].rotation.x = Math.PI*(1-2*Math.random());
meshBalls[id].rotation.z = Math.PI*(1-2*Math.random());
meshBalls[id].rotation.y = Math.PI*(1-2*Math.random());
}
var RGB = [];
var mltBalls = [];
for (let id = 0; id<200;id++)
{
    RGB.push ({"r": Math.random(), "g": Math.random(), "b": Math.random() });
    runtime.load(['dash/'+id.toString()+'.json'], function(shaders) {
        mltBalls.push ( runtime.get(shaders[0].name) );
        mltBalls[id].uniforms.image.value = new THREE.ImageUtils.loadTexture('./thumb_contra
        mltBalls[id].uniforms.Caustic_Image_Based1542394968990_145_color.value = RGB[id];
        mltBalls[id].uniforms.brightness.value = 4;
        meshBalls[id].material = mltBalls[id];
        scene.add( meshBalls[id] );});
}
```

**Frankel**

POWERED BY **MAGIC CG**

**GROUP ONE**
INTRO TO COMPUTER GRAPHICS
PACSSR - **301055**

# Camera Motion



## Circular Movement with Fixed Looking at

```
camera.position.z = 5+Math.sin( timer );
camera.position.x = 30+Math.cos(0.3*time)*5;
camera.position.y = 0+Math.sin(0.3*time)*5-5;
camera.lookAt(new THREE.Vector3(0, 0, 0));
```

**Induced Problem:**
**how to make background relative static to camera for less noise?**

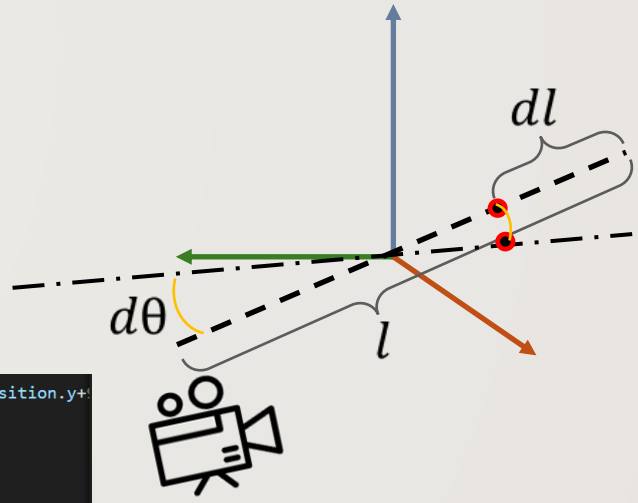
$$\vec{p} = \frac{l-l_0}{l}\overrightarrow{p_c}$$

$$\theta_y = -tan^{-1}\left(\frac{|p_{c,z}-p_{0,z}|}{|p_{c,x}-p_{0,x}|}\right)$$

$$\theta_z = -tan^{-1}\left(\frac{|p_{c,y}-p_{0,y}|}{|p_{c,xz}-p_{0,xz}|}\right)$$

```
var len = Math.sqrt(camera.position.x*camera.position.x+camera.position.y*camera.position.y+
var diff = len-base;
plane.position.x = camera.position.x*diff/len;
plane.position.y = camera.position.y*diff/len;
plane.position.z = camera.position.z*diff/len;

var angly = Math.atan(camera.position.z/camera.position.x);
plane.rotation.y =  - angly ;
var anglz = Math.atan(
Math.sqrt(camera.position.y*camera.position.y)
/
Math.sqrt(camera.position.x*camera.position.x+camera.position.z*camera.position.z)
);
plane.rotation.z = -anglz;
```

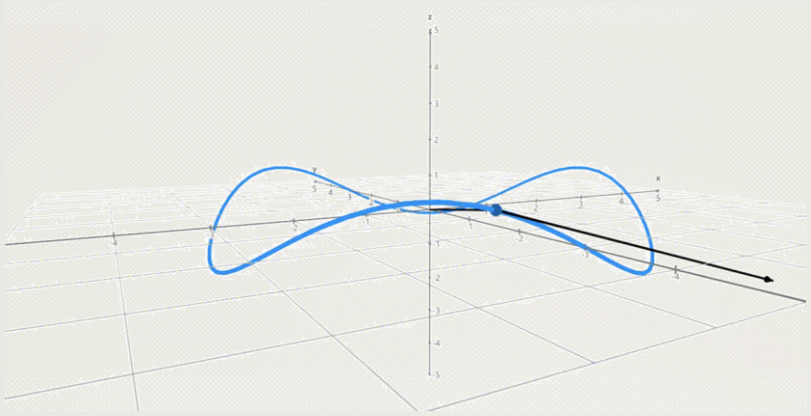


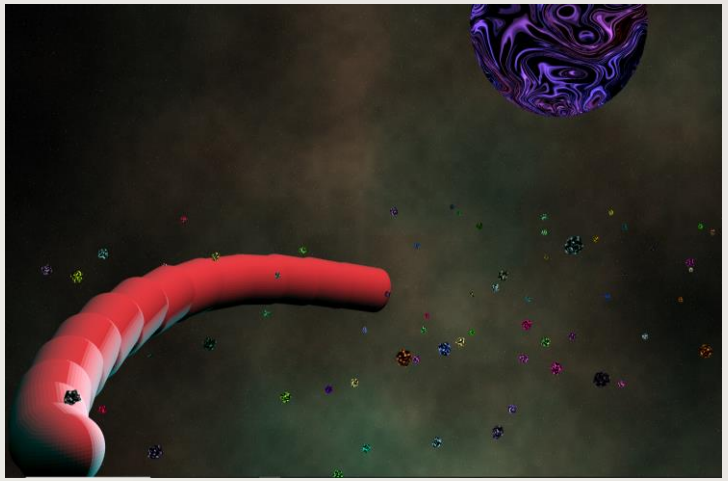
# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

# Dragon Motion Design



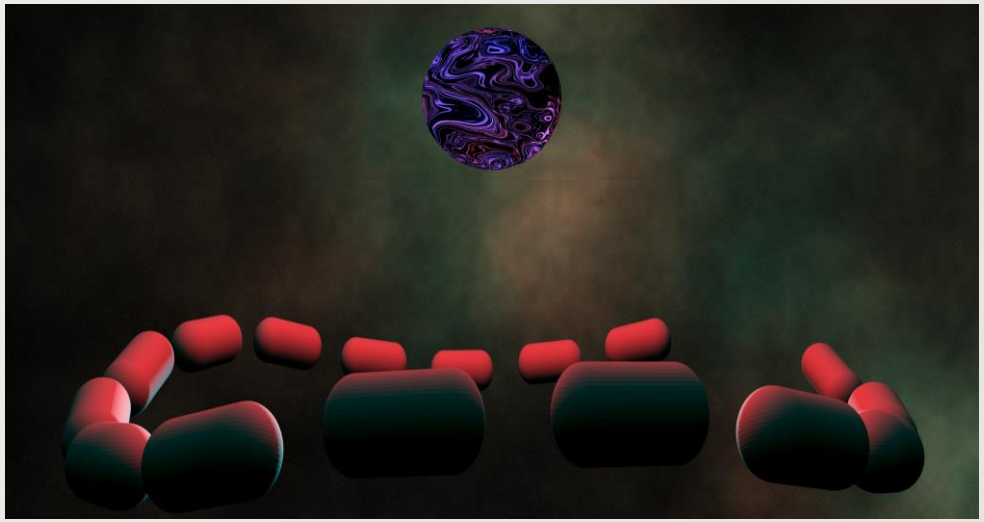


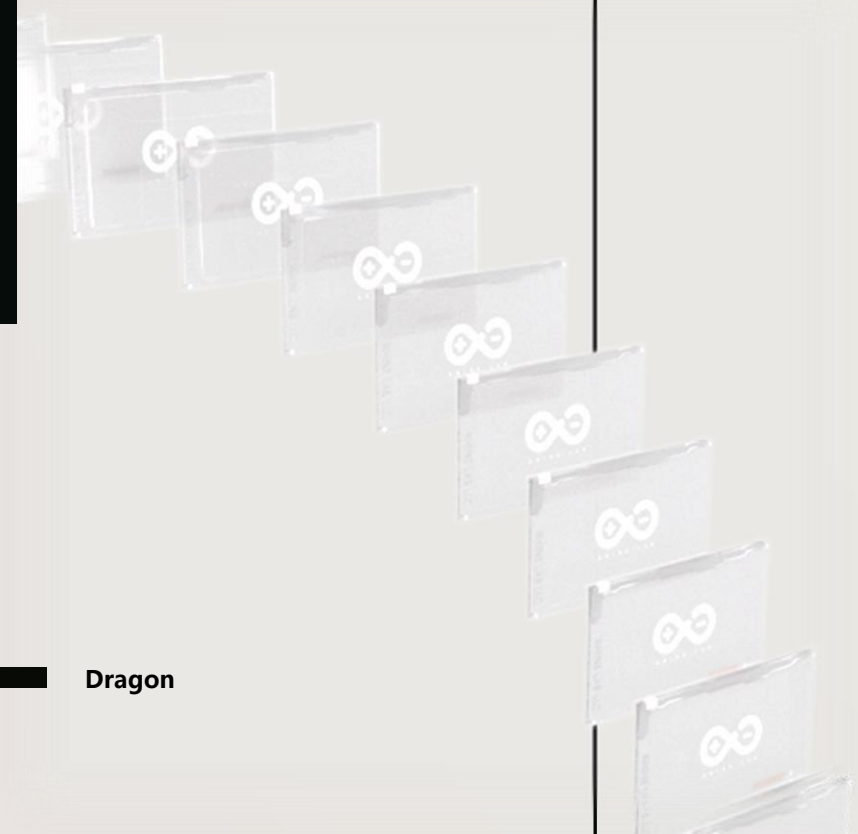
$$forAll(dargon_{section}, i)$$
$$scale_i = l_0 - dl * \sin(3t_i);$$
$$pos_i = [\mathrm{R}\sin(t_i), \mathrm{R}\cos(t_i), \mathrm{H}\cos(3t_i)];$$
$$\theta_y = -tan^{-1}\left(\frac{6H}{R}\sin(3\mathrm{t_i})\right);$$
$$\theta_Z = -\mathrm{t_i};$$

```
meshB[i].scale.y = 0.3-0.07*Math.abs(Math.cos(3*theta));

meshB[i].position.x = Math.cos(theta)*15;
meshB[i].position.z = Math.sin(theta)*15;
meshB[i].position.y = Math.cos(3*theta)*1-5;

meshB[i].rotation.y = -Math.atan(Math.sin(3*theta)*2.*3/15);
meshB[i].rotation.z = -theta;
```





SECTION

# Result

PRODUCT DEMOSTRATION SECTION

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

**Dragon**

POWERED BY **MAGIC CG**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

**SELECTING FILE**

# Thanks for watching

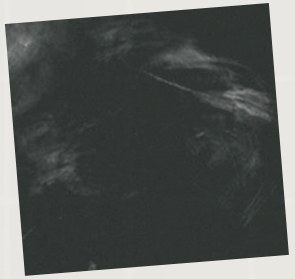
**STUDENTS** IN COMPUTER GRAPHICS

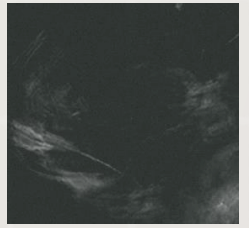
POWERED BY **MAGIC CG**

**GROUP ONE**
INTRO TO COMPUTER GRAPHICS
PACSSR - **301055**

SEARCHING

BSN
BIOGRAPHY SECTION



POWERED BY **MAGIC CG**

# GROUP ONE

INTRO TO COMPUTER GRAPHICS

PACSSR - **301055**

## CHAPTER 1

## SELECTING FILES