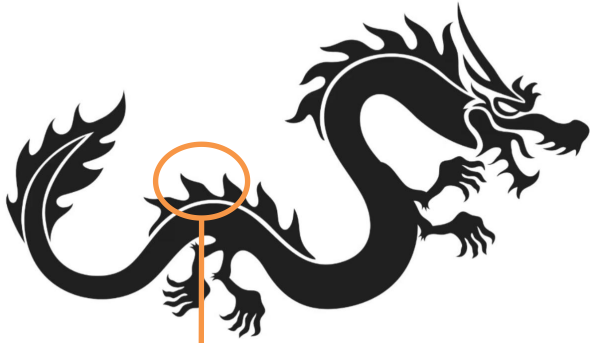
Chinese Dragon

Group 4 Final Presentation

Qianyu TANG
Dingnan SHI
Zhangyang NIE
Yufan LIU

Catalogue

Reference



Fin

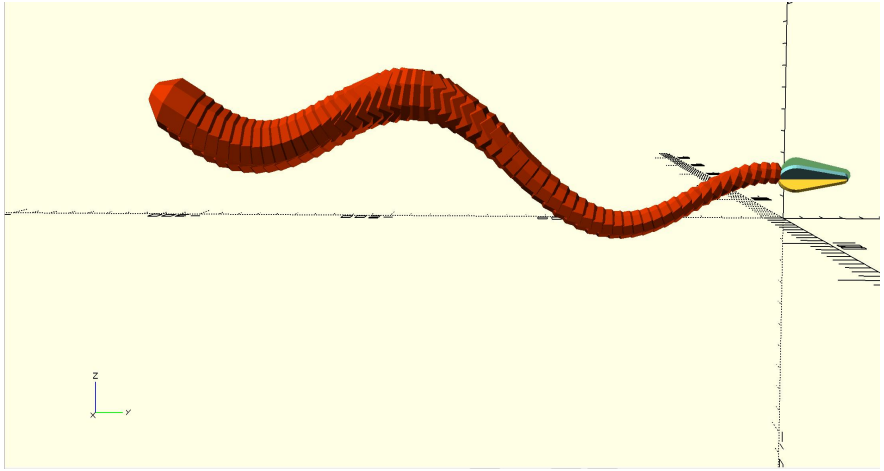
Body

Claw

Hea
d

Modeling

Belly and Tail

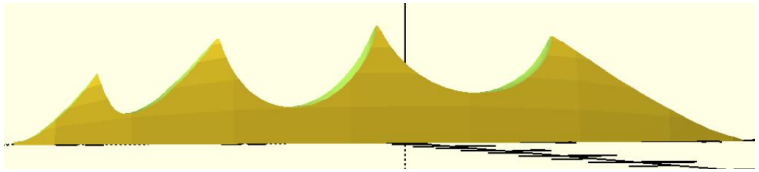


```
// belly
//element1
for (i=[1:60]){
    angle=-5*i;
    x_distance=i*0.5;
    y_distance=-i*5;
    angle_z=10*i;
    z_distance=20*cos(angle_z)+i;
    r_s=5+0.2*i;
    translate([x_distance,y_distance,z_distance])
    rotate([angle,0,0])
    scale([0.8,0.8,1])
    color([0.8,0.2,0])
    sphere(r_s,$fn = 8);

}
```

```
// tail
translate([0,6,8])
color([0.5,0.8,0.5])
scale([1,1,1])
leaf();
translate([-8,5,10])
color([0.5,0.8,0.8])
rotate([0,45,0])
scale([1,1,1])
leaf();
translate([8,5,10])
color([1,0.8,0.2])
rotate([0,-45,0])
scale([1,1,1])
leaf();
```

Modeling

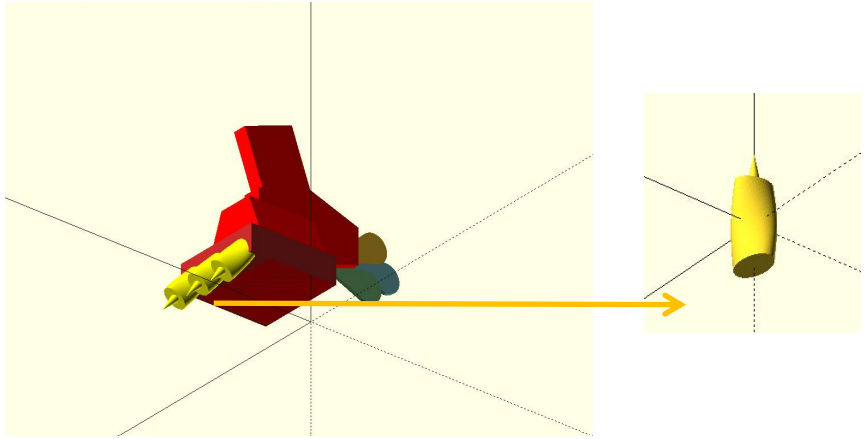
Fin



```
//fin
difference(){
    rotate([0,90,90])
    scale([1.5,0.3,5])
    sphere(1,$fn=20);
    translate([0.5,-5,0])
    rotate([0,90,90])
    scale([1.5,2,5])
    cube([1.5,0.5,5]);
    translate([0,0.8,1.6])
    rotate([0,-75,90])
    scale([1,1,1.2])
    sphere(1,$fn=40);
     translate([0,-2,4.7])
    rotate([0,20,90])
    scale([1,0.5,5])
    sphere(1,$fn=40);
    translate([0,-1.4,1.5])
    rotate([0,20,90])
    scale([1,0.6,1.2])
    sphere(1,$fn=40);
    translate([0,3.2,1.65])
    rotate([0,-53,90])
    scale([1,0.4,2.5])
    sphere(1,$fn=40);
    translate([0,-4.4,2.45])
    rotate([0,13,90])
    scale([1,0.4,2.5])
    sphere(1,$fn=40);
}
```

Modeling

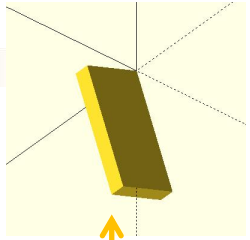
Claws(finger)



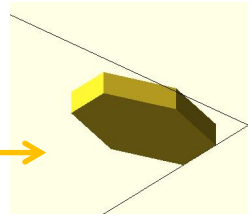
```
// Create finger module
module finger() {
    difference() {
        union() {
            translate([0, 0, finger_length-tip_length])
            cylinder(h=finger_length-tip_length, r1=finger_width/2,
r2=0);
            linear_extrude(height=30, center=true, twist=-80,
scale=0.8)
            scale([1.6,1,1])
            circle(d=10);
        }
    }
}
```

Modeling

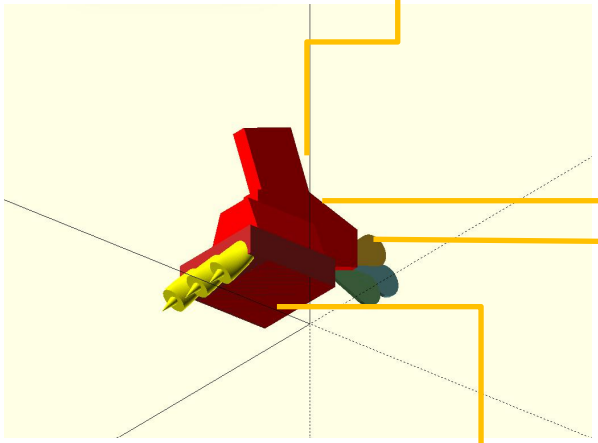
Claws

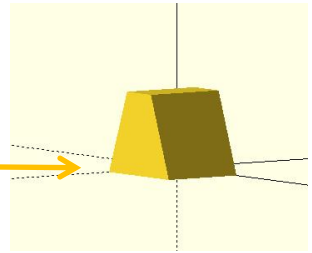


```
// Thigh
module thigh(){
    rotate([-20,0,0]){
        rotate([0,90,0]) cube([60,30,8]);
    }
}
```

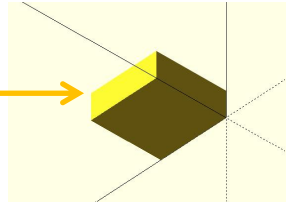


```
// Lower leg
module leg(){
    points = [
        [20, 0],
        [40, 0],
        [50, 30],
        [40, 60],
        [20, 60],
        [10, 30]
    ];

    linear_extrude(height = 10)
    polygon(points=points);
}
```





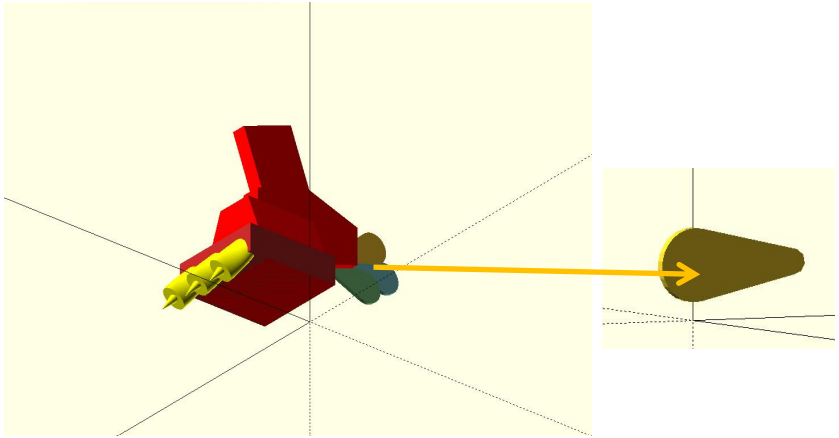
```
// Twist for heel
module twist(height, top_width, bottom_width) {
    rotate([90,0,0]){
        difference() {
            linear_extrude(height = height)
            polygon(points=[[0, 0], [top_width, 0], [top_width -
(top_width-bottom_width)/2, height], [(top_width-bottom_width)/2,
height]]);
            translate([0, 0, -height])
            polygon(points=[[0, 0], [top_width, 0], [top_width -
(top_width-bottom_width)/2, height], [(top_width-bottom_width)/2,
height]]);
        }
    }
}
```



```
// Create palm module
module palm() {
    cube([palm_width, palm_width, palm_length]);
}
```

Modeling

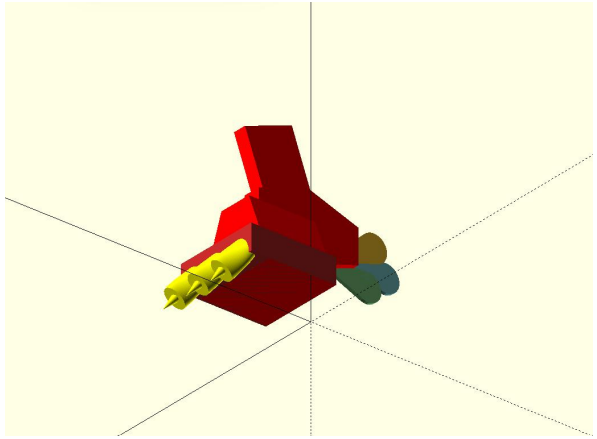
Claws(fur)



```
// Leaf module
module leaf() {


    rotate([0,90,0]){
    scale([4, 4, 2]) {
        translate([-3, 0, 0])
            difference() {
                hull() {
                    translate([0, 0, 0])
                    scale([1,1,0.2])
                    sphere(2); // Bottom circular part of the leaf
                    translate([0, 6, 0])
                   scale([0.1,1,1])
                    sphere(0.7); // Top circular part of the leaf
                }

            }
    }
}
}
```

Modeling

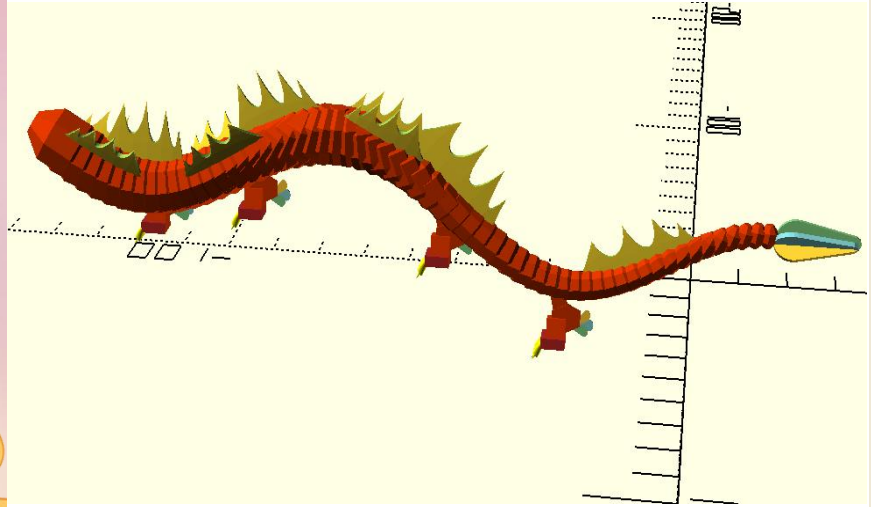
Claws



```
// Combine claws
module claw() {
    union() {
        color([0.7,0.15,0.15]){
            translate([0, -palm_length, palm_length+10]) palm();
        }
        color([1,0,0]){
            translate([0, palm_length+10, palm_length+10])
twist(40,50,20);
        }
        color([1,0,0]){
            translate([20, palm_length-60, palm_length+80]){rotate
([0,90,0]) leg(); }
            translate([20, palm_length-20, palm_length+100]) thigh();
        }
        color([0.5, 0.8, 0.8]){
            translate([25, palm_length-80, palm_length+30]) leaf();
        }
        color([0.5, 0.8, 0.5]){
            rotate([10,0,0]){translate([25, palm_length-60,
palm_length+30]) leaf();}
        }
        color([1, 0.8, 0.2]){
            rotate([-30,0,0]){translate([25, palm_length-100,
palm_length+5]) leaf();}
        }
        color([1, 1, 0])
        rotate([40,0,0]){
            mirror([0, 0, 1]) {
                translate([palm_width/3-finger_length/2, palm_width-
finger_length/3, palm_length-2*finger_width]) finger();
                translate([2*palm_width/3-finger_length/2,
palm_width-finger_length/3, palm_length-2*finger_width]) finger();
                translate([palm_width-finger_length/2, palm_width-
finger_length/3, palm_length-2*finger_width]) finger();
            }
```

Modeling

Assembling Fins



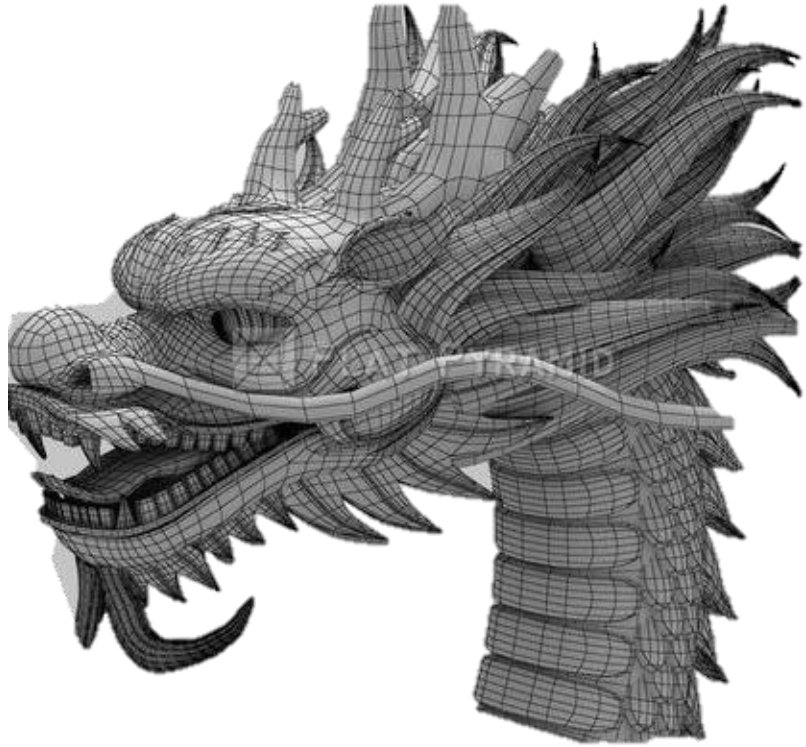
```
// Assembling fin
translate([100,-1400,225])
rotate([-30,-30,20])
scale([20,20,50])
fin();
translate([185,-1400,235])
rotate([-35,30,-10])
scale([20,20,50])
fin();
translate([140,-1340,225])
rotate([5,0,-175])
scale([20,20,80])
fin();
translate([155,-1150,245])
rotate([30,20,20])
scale([20,20,50])
fin();
translate([80,-1150,245])
rotate([20,-30,-10])
scale([20,20,50])
fin();
translate([110,-1050,300])
rotate([-25,0,-175])
scale([20,20,80])
fin();
translate([60,-820,285])
rotate([-30,-10,10])
scale([20,20,50])
fin();
translate([110,-820,285])
rotate([-30,10,5])
scale([20,20,50])
fin();
translate([75,-700,200])
rotate([45,0,-176])
scale([20,30,80])
fin();
translate([30,-300,30])
rotate([-25,0,-176])
scale([10,30,60])
fin();
}
```

Modeling

Head

- **Designing the Dragon Head**

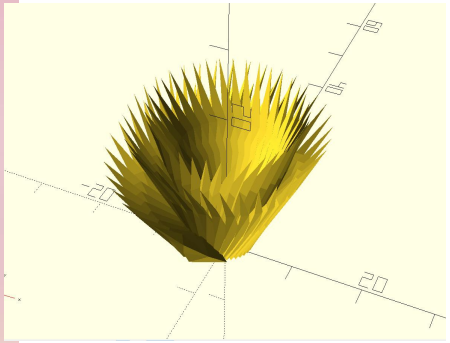
- Basic structure:
- Eyes
- Nose
- Horns
- Mane/Beard
- …



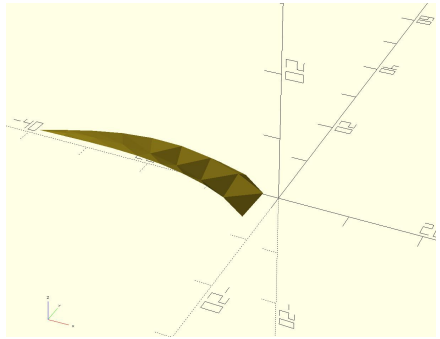
Modeling

Method:

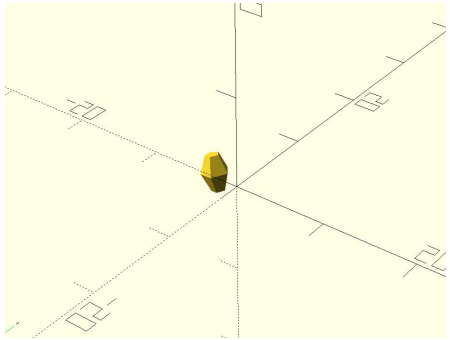
linear_extrude



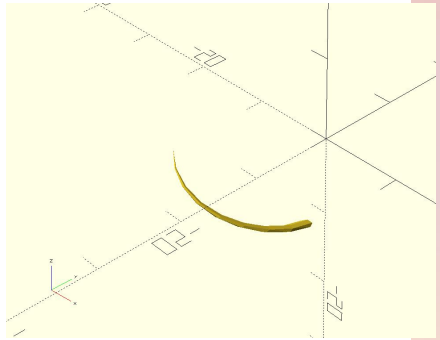
Mane(circle)



Horn(circle)

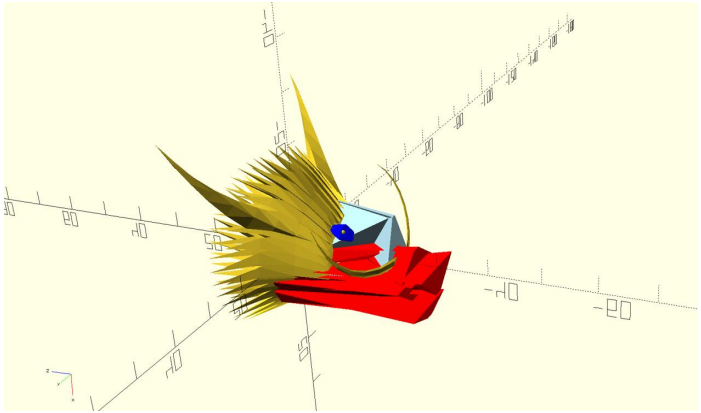


Eye(sphere)

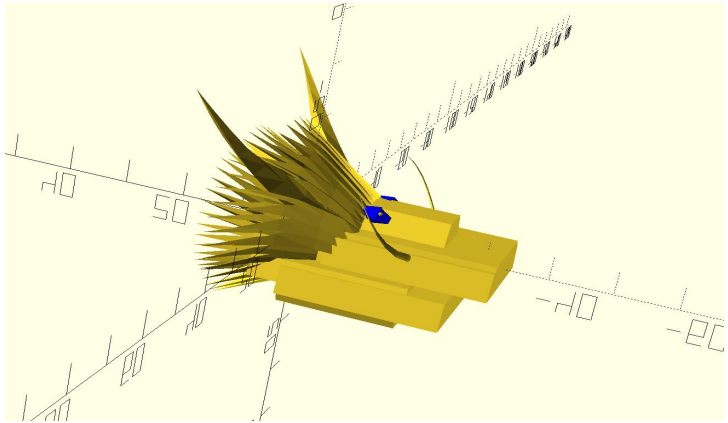


Beard(circle)

Modeling



With polyhedron()



Use linear_extrude

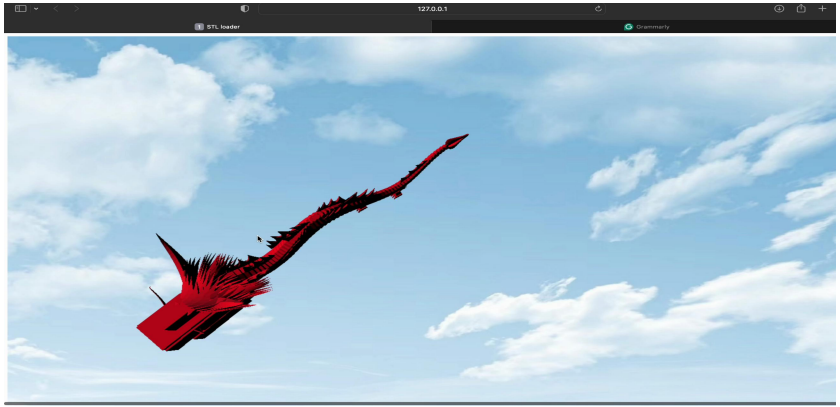
Rendering

Rendering

```
<script>
  // const spaceTexture = new THREE.TextureLoader('./wood.jpeg');
  // test.scene.background = spaceTexture;
  var scene = new THREE.Scene();
  var textureLoader = new THREE.TextureLoader();
  var texture = textureLoader.load(
    './k.jpg'
  );
  scene.background = texture
  // cmt this out if you like...

  const light = new THREE.SpotLight();
  light.position.set(300, 300, 300);
  scene.add(light);

  const camera = new THREE.PerspectiveCamera(

      75,
      window.innerWidth / window.innerHeight,
      0.1,
      10000
  );


  camera.position.set(50,50,50);
```

```
const loader = new THREE.STLLoader();
loader.load('./Dragon_1.stl',function (geometry) {
const material = new THREE.MeshPhysicalMaterial({
    color:0xff0000
});

        const mesh = new THREE.Mesh(geometry, material)
        mesh.scale.set(0.05, 0.05, 0.05);
       mesh.position.set( 8, 8, 0 );
       mesh.rotation.set( - Math.PI / 2, 0,   Math.PI / 2);
        scene.add(mesh)
    },
    (xhr) => {
        console.log((xhr.loaded / xhr.total) * 100 + '% loaded')
    },
    (error) => {
        console.log(error)
    }
);
```

Animation




```
//document.body.appendChild(stats.dom)

function animate() {
    requestAnimationFrame(animate);
    controls.update();
    render();
    //stats.update()
```