ESCENA TOTORO EN LA PARADA DEL BUS

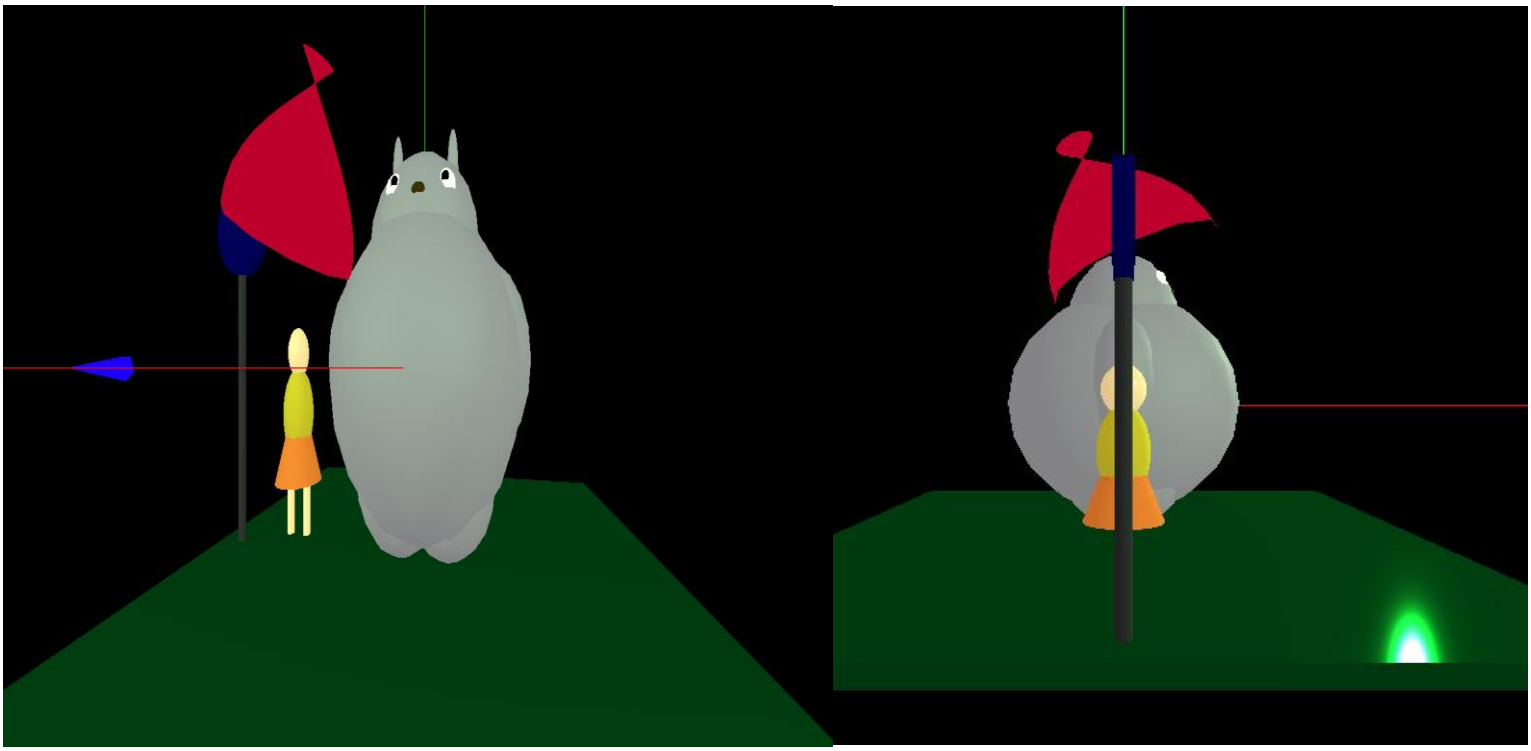
Ivan Montesino Linares (6000235)
Edwin Romero Castañeda (6000238)
Maria Paula Leon (6000281)

OBJETIVO: Recrear la escena icónica de la película mi vecino Totoro en 3D apoyados en THREE.JS

DESARROLLO DEL PROYECTO

Partimos de la escena anterior donde podemos encontrar dos personajes: Totoro y una niña con una sombrilla, esta escena está hecha en 2D (Animación 2D), realizando un análisis encontramos 4 objetos claves a la hora de recrear la escena en 3D como veremos en las siguientes imágenes:

Partiendo de este análisis y la obtención de estas 4 figuras el grupo de trabajo se puso a la tarea de recrearlas en 3D como lo vemos a continuación



Para la creación de las figuras se emplea diferentes métodos de construcción de solidos desde primitivas creadas a partir de three js hasta métodos de graficación por puntos:

Para crear una Geometria de un Cubo lo hacemos , usando BoxGeometry. Hay que pasar tres valores: Dimensiones de su ancho (x) Dimensiones de su alto (y) Dimensiones de su profundidad (z)

Y Para obtener las demás Primitivas Three js. Se utilizan las siguientes líneas de código

```
var geometry_suelo = new THREE.BoxGeometry(10, 0.25, 10 );
   var geometry_pansa = new THREE.SphereBufferGeometry( 2, 10, 10 );
   var geometry_ojos = new THREE.SphereBufferGeometry( 0.2, 10, 10 );
   var geometry_ojos2 = new THREE.SphereBufferGeometry( 0.1, 10, 10 );
   var geometry_nariz = new THREE.SphereBufferGeometry( 0.1, 10, 10 );
   var geometry_cabeza = new THREE.SphereBufferGeometry( 1.2, 10, 10 );
   var geometry_orejas = new THREE.SphereBufferGeometry( 0.2, 10, 10 );
   var geometry_patas = new THREE.SphereBufferGeometry( 1, 10, 10 );
   var geometry_brazos = new THREE.SphereBufferGeometry( 1, 10, 10 );
   var geometry_poste = new THREE.CylinderBufferGeometry( 0.08, 0.08, 4, 50 );
   var geometry_letrero = new THREE.CylinderBufferGeometry( 0.6, 0.6, 0.2, 50 );

     var geometry_blusa = new THREE.SphereBufferGeometry( 0.3, 0.3, 20 );
   var geometry_falda = new THREE.CylinderBufferGeometry( 0.28, 0.5, 0.6, 50 );
   var geometry_piernas = new THREE.CylinderBufferGeometry( 0.08, 0.08, 1, 50  );
   var geometry_cabezan = new THREE.SphereBufferGeometry    ( 0.18, 0.2, 20 );
   var geometry_sombrilla = new THREE.LatheBufferGeometry( points );
```

Arreglo de puntos

En este caso se utilizó el arreglo de puntos para crear la sombrilla

```
var points = [];
  for ( var i = 0; i < 10; i ++ ) {
      points.push( new THREE.Vector2( Math.cos( i * 0.2 ) * 1.1, ( i +10 ) * 0.2 ) );
  }
```

```
var geometry_sombrilla = new THREE.LatheBufferGeometry( points );
```

```
var sombrilla = new THREE.Mesh( geometry_sombrilla, material10 );
```

Para visualizar mejor la escena se crea una iluminación haciendo que las texturas resalten

```
var ambient = new THREE.AmbientLight( 0x888888, 1 );
scene.add( ambient );

var pointLight1 = new THREE.PointLight( 0x497645, .5, 100 );
pointLight1.position.set( 10, 5, -10 );
scene.add( pointLight1 );

var pointLight2 = new THREE.PointLight( 0x497645, .5, 100 );
pointLight2.position.set( 10, 5,10  );
scene.add( pointLight2 );
```

Y para Crear un material, necesitamos utilizar la clase MeshBasicMaterial, que necesita que le indiquemos el color hexadecimal con el que pintar la figura. Vamos a hacer 6 materiales, uno por cara.

Para realizar una pequeña prueba de animacion se le aplica una transformación en la posición y en la rotación gracias a la variable tiempo

```
var render = function(){


     requestAnimationFrame( render );


    var timer = Date.now() - start;

 pansa.position.y = Math.sin(timer*0.002)*0.5;
 sombrilla.position.y = Math.sin(timer*0.001)*0.5;
 sombrilla.rotation.y = Math.sin(timer*0.001)*2;
```

Todo esto se realiza a parir de la Herencia de objetos

CONCLUSIONES

-Three.js es la librería principal para trabajar con WebGL Que a diferencia de trabajar con OPENGL esta nos facilita la creación de objetos ya que sus librerías y demás ya las tiene incluidas

• El procedimiento para implementar objetos en 3D es sencillo lo complicado son los detalles que en parte mejorarían la calidad de la presentación del trabajo y para este se utilizaron detalles texturas y colores sencillos

• La aplicación que se desarrolló con la expectativa de mostrar una escena en 3d y se logró el objetivo de la misma y de esta manera dar a conocer con nuestro código y este documento la realización de la misma.