

这个案例是官方案例，个人通过一个一个测试备注，感受模型如何配置参数，来生成理想的形状。three.js内置的这些几何确实够平常使用，还可以通过顶点实现效果。

案例查看地址：<http://www.wjceo.com/blog/threejs/2018-02-09/9.html>

案例代码：

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="UTF-8">
5. <title>Three框架</title>
6. <script src="build/three.js"></script>
7. <style type="text/css">
8. html, body {
9. margin: 0;
10. height: 100%;
11. }
13. canvas {
14. display: block;
15. }
17. </style>
18. <script>
19. *//渲染器*
20. var renderer;
21. function initRender() {
22. width = window.innerWidth;
23. height = window.innerHeight;
24. renderer = new THREE.WebGLRenderer({
25. antialias: true
26. });
27. *//设置canvas尺寸*
28. renderer.setSize(width, height);
29. *//设置背景*
30. renderer.setClearColor(0x000000, 1.0);
31. *//设置设备像素比*
32. renderer.setPixelRatio( window.devicePixelRatio );
33. *//添加到dom*
34. document.body.appendChild(renderer.domElement);
35. }
37. *//相机*
38. var camera;
39. function initCamera() {
40. camera = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 2000 );
41. camera.position.y = 400;
42. }
44. *//场景*
45. var scene;
46. function initScene() {
47. scene = new THREE.Scene();
48. }
50. *//光源*
51. var light;
52. function initLight() {
53. *//添加环境光*
54. scene.add(new THREE.AmbientLight( 0x404040));
56. *//添加平衡光*
57. light = new THREE.DirectionalLight( 0xffffff );
58. light.position.set(0,1,0);
59. scene.add(light);
60. }
62. *//模型*
63. function initModel() {
64. *//通过加载图片生成一个纹理*
65. var map = new THREE.TextureLoader().load("examples/textures/UV\_Grid\_Sm.jpg");
66. *//定义纹理在水平和垂直方向简单的重复到无穷大。*
67. map.wrapS = map.wrapT = THREE.RepeatWrapping;
68. *//定义纹理的各向异性*
69. map.anisotropy = 16;
71. *//定义兰伯特网孔材质*
72. var material = new THREE.MeshLambertMaterial({map:map,side:THREE.DoubleSide});
74. *//球形网格 （半径长度，水平块的密度，垂直块的密度）*
75. object = new THREE.Mesh( new THREE.SphereGeometry(75,20,10), material);
76. object.position.set( -400, 0, 200 );
77. scene.add(object);
79. *//二十面体 （图形大小半径，大于零将不是二十面体，越大越圆滑）*
80. object = new THREE.Mesh( new THREE.IcosahedronGeometry( 75, 0 ), material );
81. object.position.set( -200, 0, 200 );
82. scene.add( object );
84. *//八面体（图形大小半径，大于零将不是八面体，越大越圆滑）*
85. object = new THREE.Mesh( new THREE.OctahedronGeometry( 75, 0 ), material );
86. object.position.set( 0, 0, 200 );
87. scene.add( object );
89. *//四面体（图形大小半径，大于零将不是四面体，越大越圆滑）*
90. object = new THREE.Mesh( new THREE.TetrahedronGeometry( 75, 0 ), material );
91. object.position.set( 200, 0, 200 );
92. scene.add( object );
94. *//长方形平面 （x轴宽度，y轴高度，x方向分段数，y方向分段数）*
95. object = new THREE.Mesh( new THREE.PlaneGeometry( 100, 100, 1, 1 ), material );
96. object.position.set( -400, 0, 0 );
97. scene.add( object );
99. *//立方体 （x轴宽度，y轴高度，z轴深度，沿宽面分段数，沿高度面分段数，沿深度面分段数）*
100. object = new THREE.Mesh( new THREE.BoxGeometry( 100, 100, 100, 1, 1, 1 ), material );
101. object.position.set( -200, 0, 0 );
102. scene.add( object );
104. *//圆形平面 （半径，顶点密度，绘制起点弧度，绘制弧度）*
105. object = new THREE.Mesh( new THREE.CircleGeometry( 50, 20, 0, Math.PI \* 2 ), material );
106. object.position.set( 0, 0, 0 );
107. scene.add( object );
109. *//空心圆平面 （内圆半径，外圆半径，分割面越大越圆滑，垂直外边分割面，开始绘制弧度，绘制弧度）*
110. object = new THREE.Mesh( new THREE.RingGeometry( 10, 50, 10, 5, 0, Math.PI \* 2 ), material );
111. object.position.set( 200, 0, 0 );
112. scene.add( object );
114. *//圆柱体 （头部圆的半径，底部圆半径，高度，上下圆顶点个数，上下面切割线条数，上下面是否显示，开始弧度，绘制弧度）*
115. object = new THREE.Mesh( new THREE.CylinderGeometry( 25, 75, 100, 40, 5 ), material );
116. object.position.set( 400, 0, 0 );
117. scene.add( object );
119. *//车床模型*
120. var points = [];
122. for ( var i = 0; i < 50; i ++ ) {
124. points.push( new THREE.Vector2( Math.sin( i \* 0.2 ) \* Math.sin( i \* 0.1 ) \* 15 + 50, ( i - 5 ) \* 2 ) );
126. }
128. *//（一个vector2的数组分别代表xy轴，生成圆周段的数目，开始弧度，绘制弧度）*
129. object = new THREE.Mesh( new THREE.LatheGeometry( points, 20 ), material );
130. object.position.set( -400, 0, -200 );
131. scene.add( object );
133. *//救生圈 （救生圈半径，管道直径，基于管道横切顶点数，救生圈横切顶点个数）*
134. object = new THREE.Mesh( new THREE.TorusGeometry( 50, 20, 20, 20 ), material );
135. object.position.set( -200, 0, -200 );
136. scene.add( object );
138. *//环面扭结模型 （图形半径，管道直径，基于管道横切定点数，根据图形半径横切顶点数，绕旋转对称轴的圈数，绕环面的圆的圈数）*
139. object = new THREE.Mesh( new THREE.TorusKnotGeometry( 50, 10, 50, 20 ), material );
140. object.position.set( 0, 0, -200 );
141. scene.add( object );
143. *//轴辅助 （每一个轴的长度）*
144. object = new THREE.AxisHelper( 50 );
145. object.position.set( 200, 0, -200 );
146. scene.add( object );
148. *//箭头辅助（箭头头的方向必须是vecteor3，箭头起点必须是vector3，箭头长度，颜色）*
149. object = new THREE.ArrowHelper( new THREE.Vector3( 0, 1, 0 ), new THREE.Vector3( 0, 0, 0 ), 50 ,0x00ffff);
150. object.position.set( 400, 0, -200 );
151. scene.add( object );

154. }
156. function animate() {
158. requestAnimationFrame( animate );
160. render();
161. *//stats.update();*
163. }
165. function render() {
167. var timer = Date.now() \* 0.0001;
169. camera.position.x = Math.cos( timer ) \* 800;
170. camera.position.z = Math.sin( timer ) \* 800;
172. camera.lookAt( scene.position );
174. for ( var i = 0, l = scene.children.length; i < l; i ++ ) {
176. var object = scene.children[ i ];
178. object.rotation.x = timer \* 5;
179. object.rotation.y = timer \* 2.5;
181. }
183. renderer.render( scene, camera );
185. }
187. *//窗口变动触发的函数*
188. function onWindowResize() {
190. camera.aspect = window.innerWidth / window.innerHeight;
191. camera.updateProjectionMatrix();
193. renderer.setSize( window.innerWidth, window.innerHeight );
195. }
197. *//绘制*
198. function draw() {
199. initRender();
200. initCamera();
201. initScene();
202. initLight();
203. initModel();
205. animate();
207. window.addEventListener( 'resize', onWindowResize, false );
208. }


212. </script>
213. </head>
215. <body onload="draw();">
216. </body>
217. </html>