Three.js

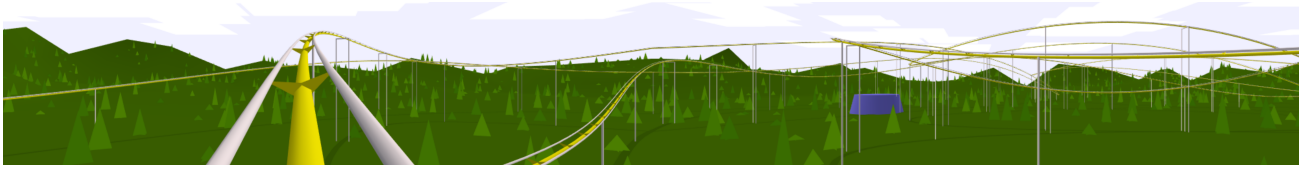
Three.js is a powerful 3D graphics engine built on WebGL - threejs.org

We have written a Javascript module, IglooRenderer.js, for integrating sites and applications built with three.js. The integration workflow draws a cubemap to the Igloo Web canvas, which is in turn shared with the Igloo Warper as an equirectangular texture.

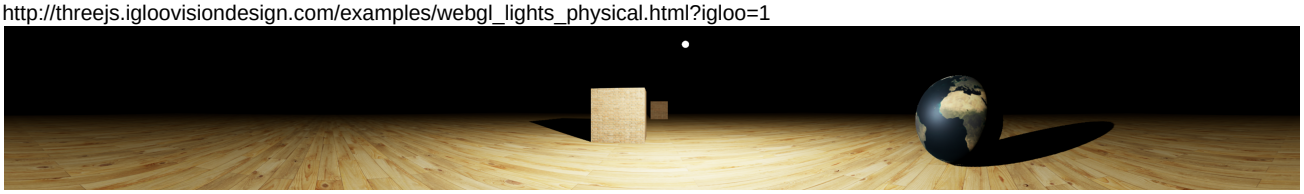
There are a number of sites listed below, which are part of the standard set of three.js examples, and have been modified to include the Igloo Renderer module. The Igloo integration is activated by adding igloo=1 as a URL parameter. Omitting this parameter, or by using igloo=0, will display the site normally.

NB The Igloo Web canvas should have a 6:1 aspect ratio to properly display the cubemap, e.g. 6000 x 1000.

http://threejs.igloovisiondesign.com/examples/webxr_vr_rollercoaster.html?igloo=1

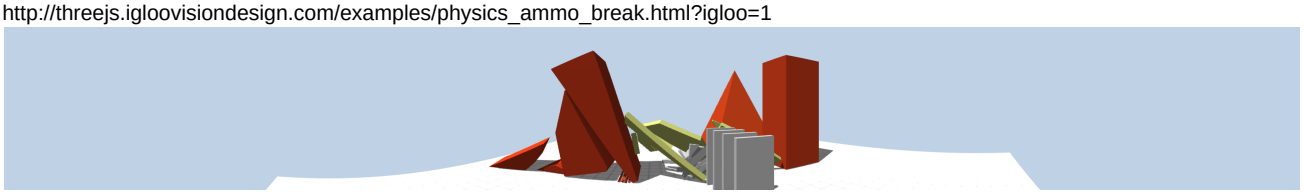


http://threejs.igloovisiondesign.com/examples/webgl_lights_physical.html?igloo=1

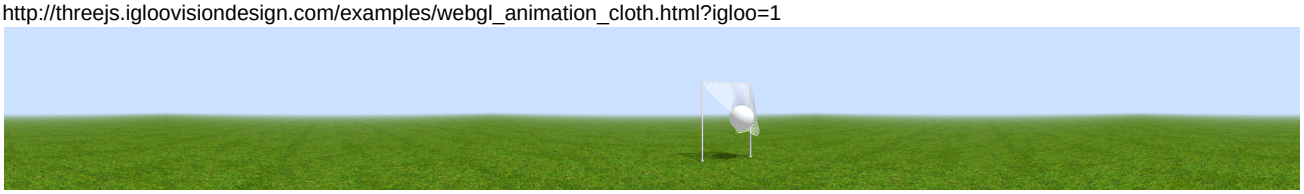


http://threejs.igloovisiondesign.com/examples/webgl_loader_md2_control.html?igloo=1

http://threejs.igloovisiondesign.com/examples/physics_ammo_break.html?igloo=1



http://threejs.igloovisiondesign.com/examples/webgl_animation_cloth.html?igloo=1



Applying the Igloo integration

The examples above have all been modified in the same way. The steps below show modifications to the webgl_animation_cloth.html file.

The Igloo module can be downloaded from: http://threejs.igloovisiondesign.com/igloo/IglooRenderer.js

First step is to include the Igloo module, line 14 below (line 32 in webgl_animation_cloth.html):

```
<body>
                <div id="info">Simple Cloth Simulation<br/>
                        Verlet integration with relaxed
constraints<br/>
                </div>

                <script type="module">

                        import * as THREE from '../build/three.
module.js';

                        import Stats from './jsm/libs/stats.module.
js';
                        import { GUI } from './jsm/libs/dat.gui.
module.js';

                        import { OrbitControls } from './jsm
/controls/OrbitControls.js';
                        import { IglooRenderer } from './jsm
/renderers/IglooRenderer.js';
```

Add a global variable, line 3 below (line 410 in webgl_animation_cloth.html):

```
            var container, stats;
            var camera, scene, renderer;
                        var igloo;
```

Next step is to create an instance of the Igloo renderer, line 19 below (line 574 in webgl_animation_cloth.html). This should follow the creation of the scene, camera and renderer objects.

```
                // renderer

                                renderer = new THREE.WebGLRenderer(
{ antialias: true } );
                                renderer.setPixelRatio( window.
devicePixelRatio );
                                renderer.setSize( window.
innerWidth, window.innerHeight );

                                container.appendChild( renderer.
domElement );

                                renderer.outputEncoding = THREE.
sRGBEncoding;

                                renderer.shadowMap.enabled = true;

                                // controls
                                var controls = new OrbitControls(
camera, renderer.domElement );
                                controls.maxPolarAngle = Math.PI *
0.5;
                                controls.minDistance = 1000;
                                controls.maxDistance = 5000;

                                igloo = new IglooRenderer(camera,
renderer, scene);
```

Include an update on the window resize event, line 8 below (line 614 in webgl_animation_cloth.html):

```
            function onWindowResize() {

                                camera.aspect = window.innerWidth /
window.innerHeight;
                                camera.updateProjectionMatrix();

                                renderer.setSize( window.
innerWidth, window.innerHeight );

                                igloo.update();

                        }
```

Finally, modify the render scene step, lines 19 to 22 (lines 647 to 650 in webgl_animation_cloth.html).:

```
                        function render() {

                                var p = cloth.particles;

                                for ( var i = 0, il = p.length; i <
il; i ++ ) {

                                        var v = p[ i ].position;

                                        clothGeometry.attributes.
position.setXYZ( i, v.x, v.y, v.z );

                                }

                                clothGeometry.attributes.position.
needsUpdate = true;

                                clothGeometry.
computeVertexNormals();

                                sphere.position.copy( ballPosition
);

                                if (igloo.render() != true) {
                                        renderer.render( scene,
camera );
                                }
                                //renderer.render( scene, camera );

                        }
```