SuperMap iClient3D for WebGL 中三维场景优化美化常用设置

一、背景需求：

SuperMap iClient3D for WebGL 的场景效果并不限于每次打开时看到的样子，它还可以进行各种个性化的设置。比如，制作一个晴朗的天空；调出夕阳西下的光影；打造科技感城市夜景特效；展现白模的边框轮廓；优化精模的纹理效果等等。从天空的效果，到场景的光照，及模型的外观均可以进行改变或设置。

本文档从以下几个方面分别进行说明：

1. 天空背景相关设置（天空盒，大气层，太阳光晕）；
2. 场景默认光照设置（环境光，太阳光）；
3. 自定义场景光源；
4. HDR 对场景的影响；
5. 模型烘焙纹理的处理；
6. 场景/图层颜色的调整；
7. 模型边界效果的优化；
8. 模型边框线的显示与优化

二、功能介绍：

2.1 天空背景的相关设置

与天空相关的功能主要包括：天空盒，大气层，太阳光晕等。

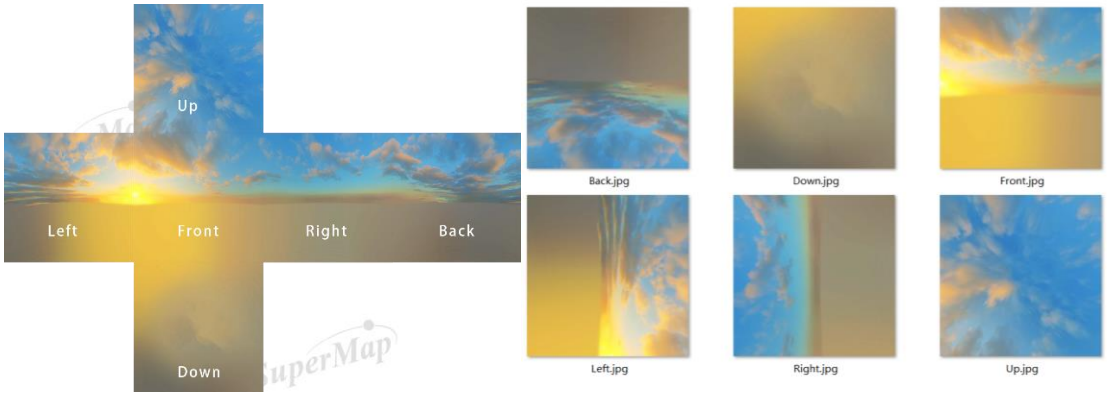
2.1.1 如何制作自定义天空盒效果

1.制作天空盒素材

①图像数量：一组六张形式的天空盒图片；

②图像的命名及摆放：六张图片按照左图上面标注的顺序进行排列；然后按照右图的形式旋转相应的角度；

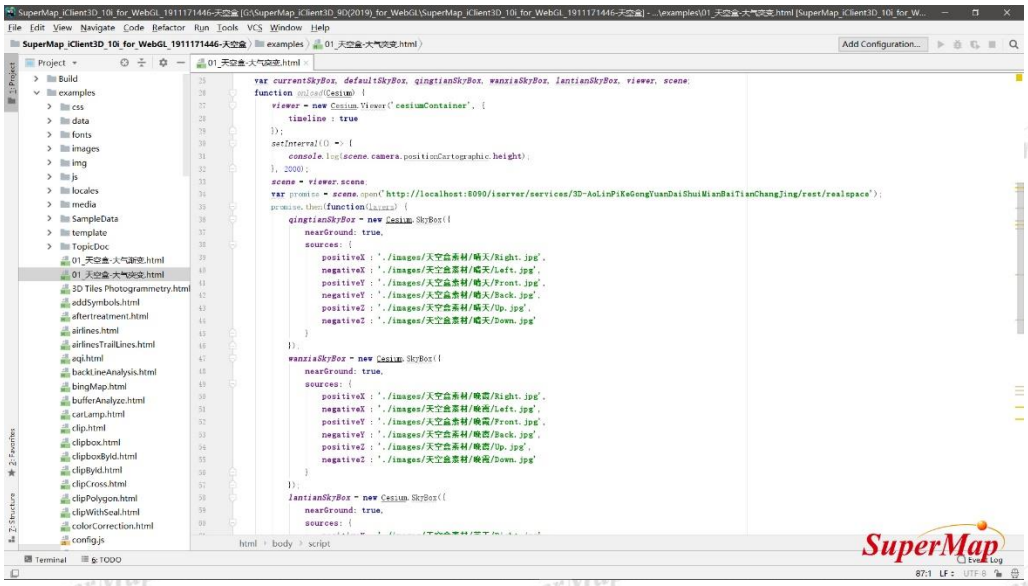
③图像的大小：1024×1024 像素



2.使用方法：

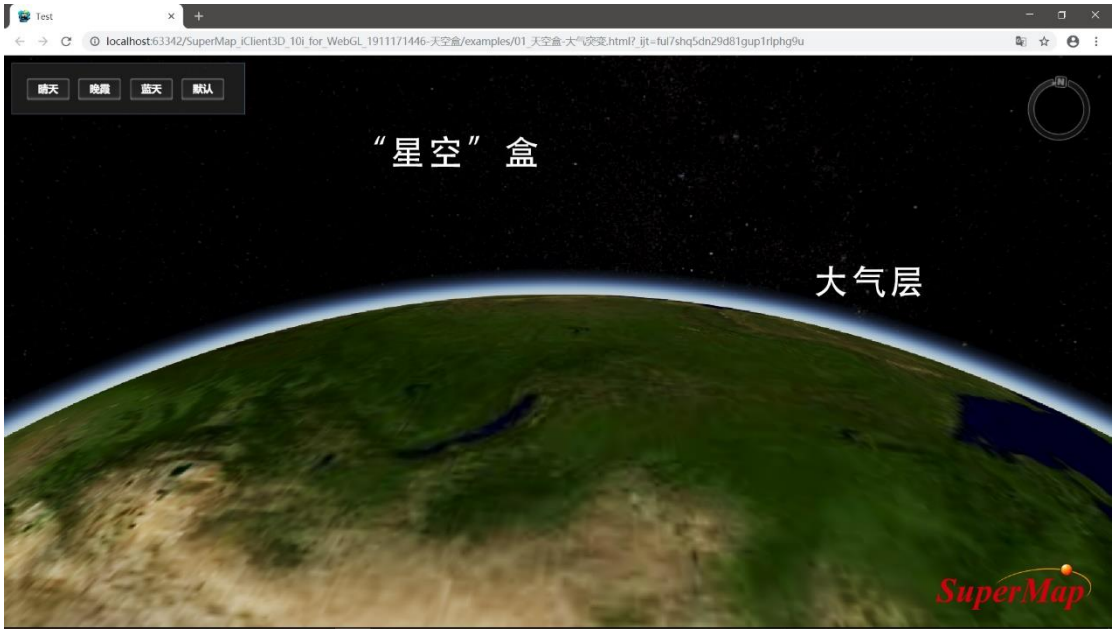
①创建一个自定义天空盒，并将图片对应到不同的方位下

```
sources: {
    positiveX : './images/天空盒素材/晴天/Right.jpg',
    negativeX : './images/天空盒素材/晴天/Left.jpg',
    positiveY : './images/天空盒素材/晴天/Front.jpg',
    negativeY : './images/天空盒素材/晴天/Back.jpg',
    positiveZ : './images/天空盒素材/晴天/Up.jpg',
    negativeZ : './images/天空盒素材/晴天/Down.jpg'
}
```



③控制自定义天空盒的出现条件

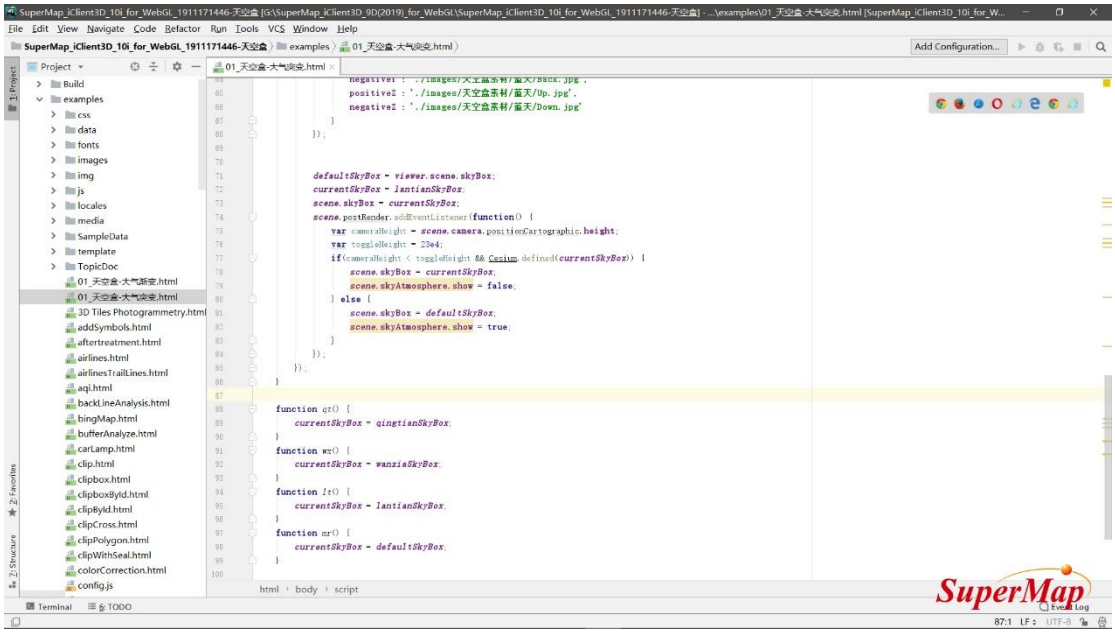
在创建场景时，系统默认会创建一个星空效果的“天空盒”及围绕地球的大气层。

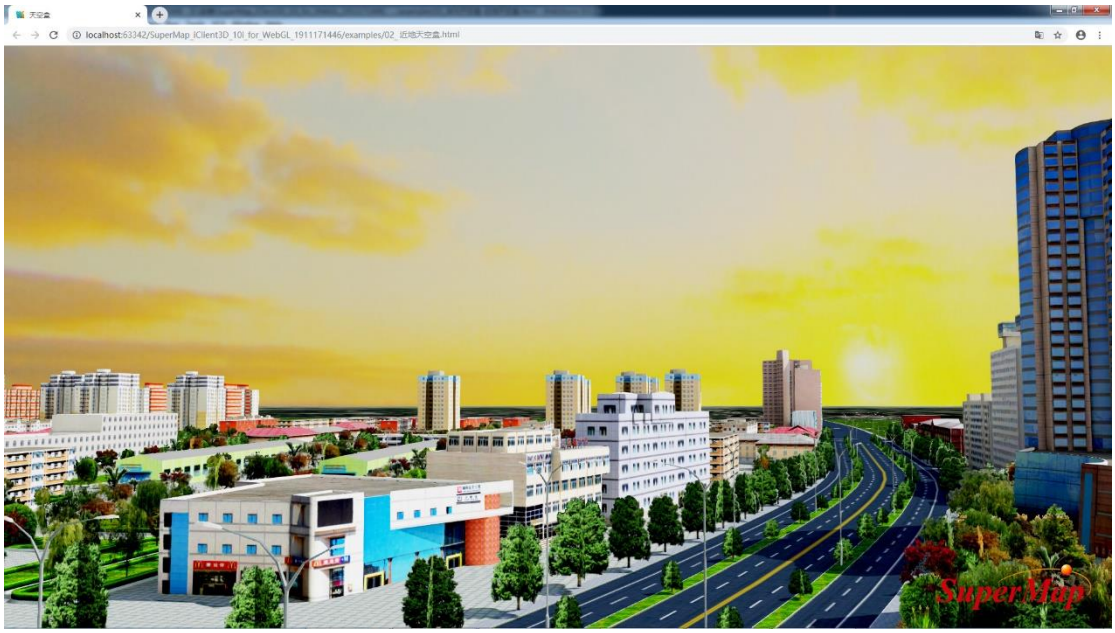


而当我们拉近视角，进入地平面时，需要切换为天空的效果。

这里我们设置一个高度，当大于这个高度时，我们距离地球比较远，看到的是地球周围的大气层与星空；当小于这个高度时，我们距离地球比较近，就会看到天空的效果。

```
scene.postRender.addEventListener(function() {
    var cameraHeight = scene.camera.positionCartographic.height;
    var toggleHeight = 23e4;
    if(cameraHeight < toggleHeight && Cesium.defined(currentSkyBox)) {
        scene.skyBox = currentSkyBox;
        scene.skyAtmosphere.show = false;
    } else {
        scene.skyBox = defaultSkyBox;
        scene.skyAtmosphere.show = true;
    }
});
```





④另外，如果想要在拉近场景时，有一个大气层逐渐消失，天空逐渐显现的渐变效果，可以通过控制二者的透明度来实现。

3.注意事项：

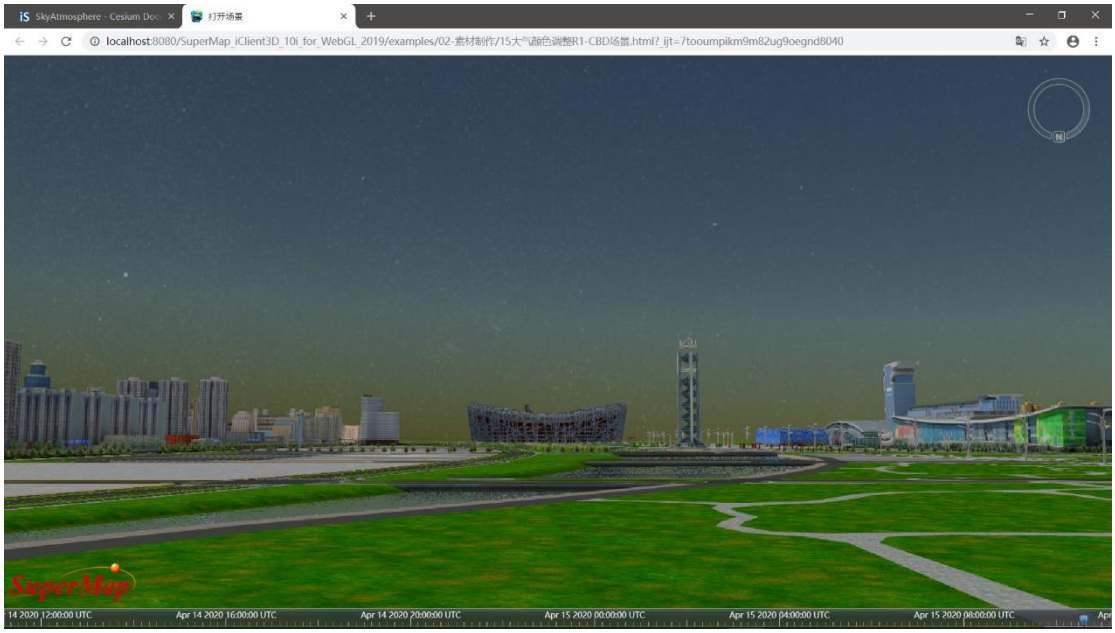
①六张天空盒图片的顺序与旋转角度必须保证正确，否则场景中的天空会显示不正常；

②在 2019 年 10 月之后的 SuperMap_iClient3D_10i_for_WebGL 版本才支持此功能。

2.1.2 如何更改大气层的渲染效果

1.快速改变大气层的渲染效果

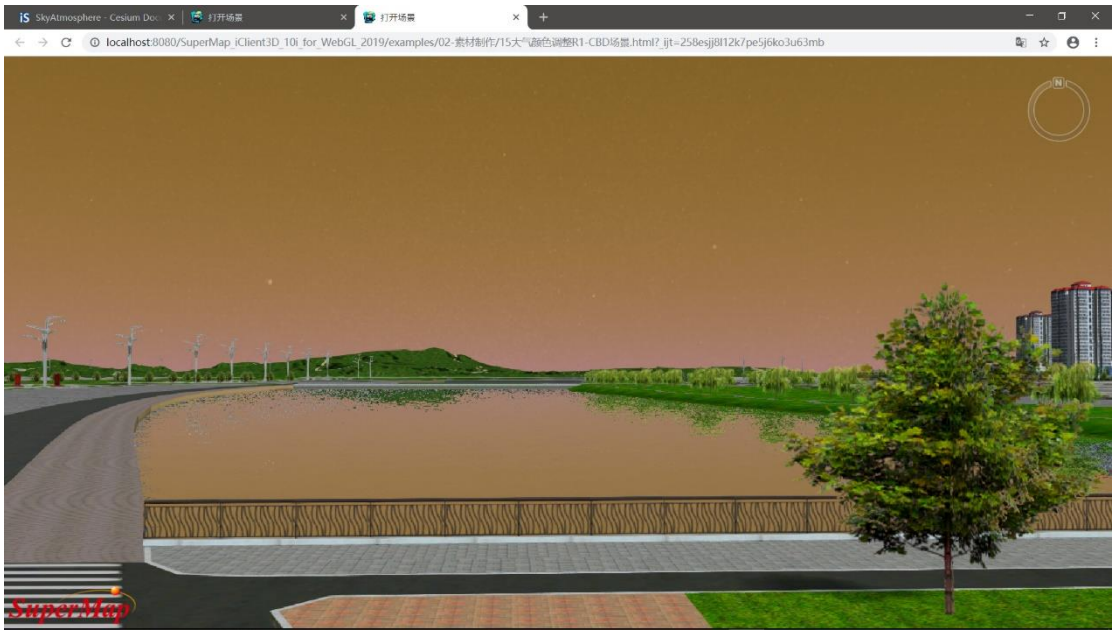
设置 viewer.scene.globe.enableLighting = true;同时开启时间轴，通过改变时间，即可快速调整大气层的效果。

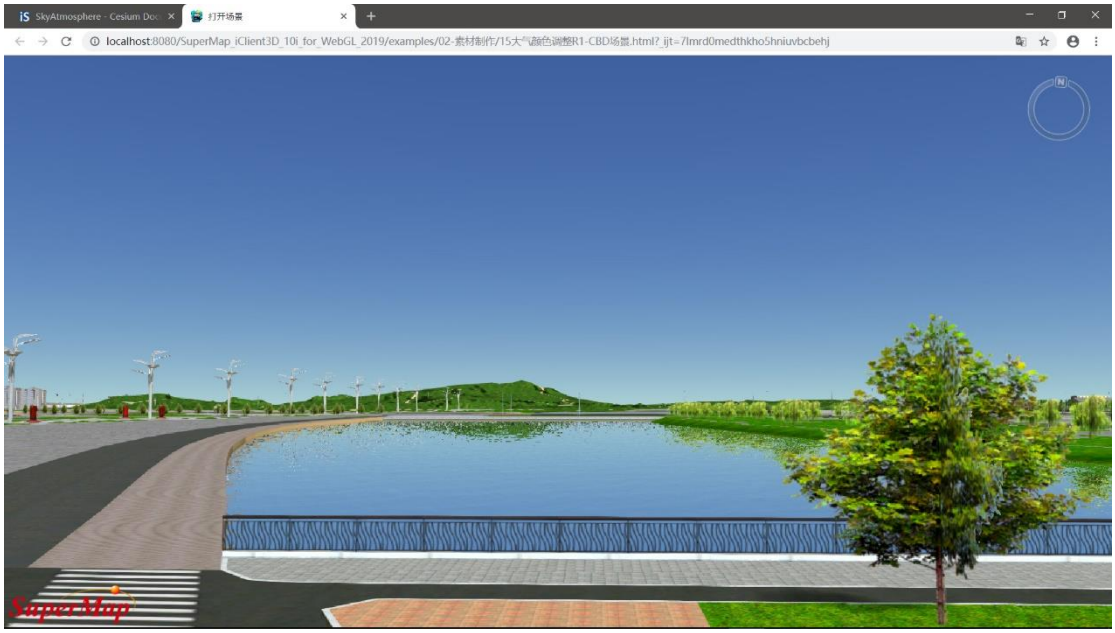


2.自定义控制大气层的渲染效果

如何制作一个暖色调的大气层？通过控制大气层的色相向橙黄色偏移，同时降低其亮度及饱和度。如果还想让星辰若隐若现，还可以降低大气层的透明度。

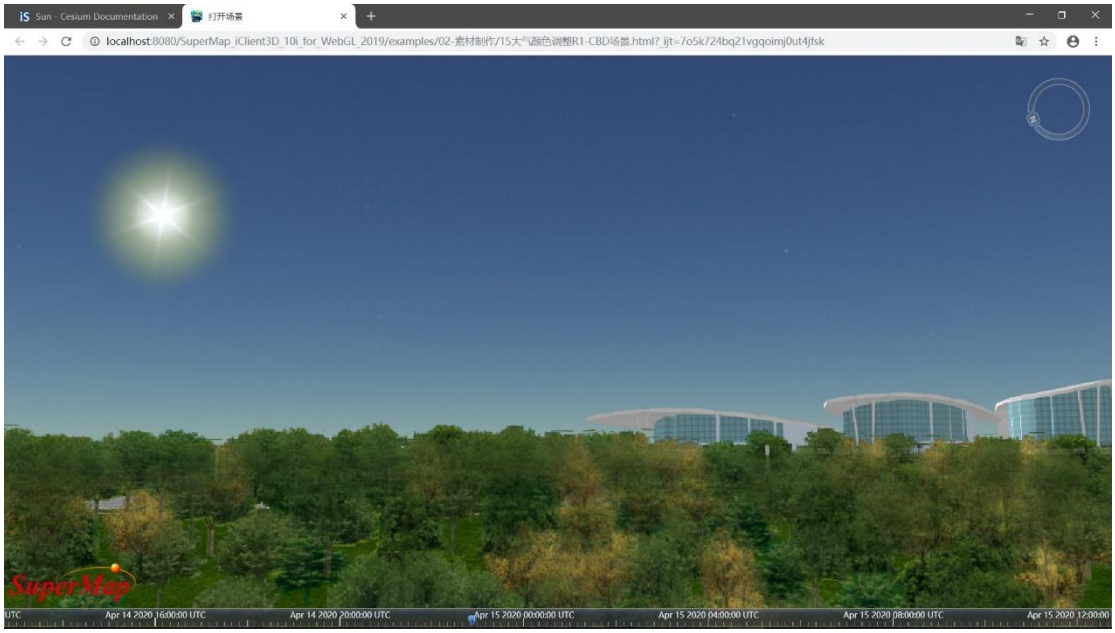
```
scene.skyAtmosphere.show =true;
scene.skyAtmosphere.brightnessShift=0.1;
scene.skyAtmosphere.hueShift=0.5;
scene.skyAtmosphere.saturationShift =0.1;
scene.skyAtmosphere.alpha =0.8;
```





2.1.3 如何改变太阳的显示效果

这里所说的太阳的显示效果，是指改变太阳的“外观”，而不是改变它发射出来的“光线”。通过调整 scene.sun.glowFactor 的值来实现。



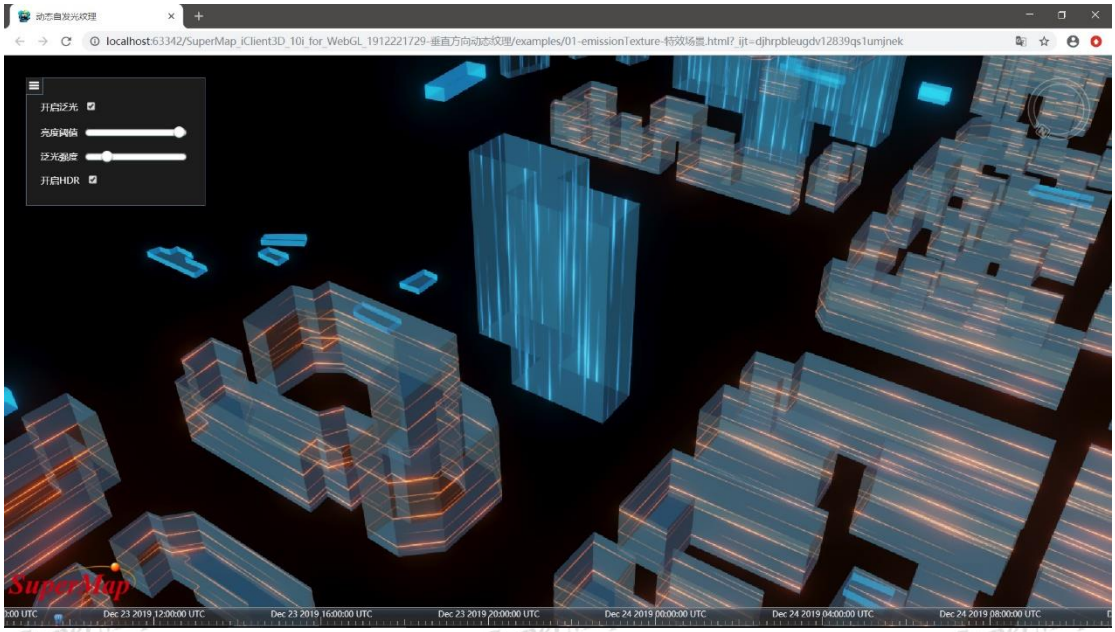
2.2 场景默认光照设置

在 SuperMap iClient3D for WebGL 中，场景中默认存在两个光源：1.环境光（ambientLight）；2.太阳光（sunLight）。二者共同决定了场景中的光照效果。

2.2.1 如何调整环境光的效果

环境光是从四周发射光线来照亮场景的，没有方向性。常见的需要调节环境光的场景有以下三种：1.夜景或比较暗的场景中，需要一个较暗的环境光；2.模型的纹理本身带有较重的烘焙阴影，需要一个较亮的环境光提高阴影部分的亮度；3.配合场景中其他的光源效果来调整环境光的参数。

下图所示的场景中，就是设置了一个很低值环境光的效果（为了突显模型上的自发光纹理效果，设置 scene.lightSource.ambientLightColor = new Cesium.Color(0.1, 0.1, 0.1, 0.1))。



2.2.2 修改光线后没有起作用

如果场景中的对象是以 entity 的方式添加的 glTF 模型，调整太阳光及环境光之后，模型的光照效果时没有变化的。那么如果想要制作一个夜景的效果要怎么设置？可以通过修改 imageBasedLightingFactor 的值来调整。

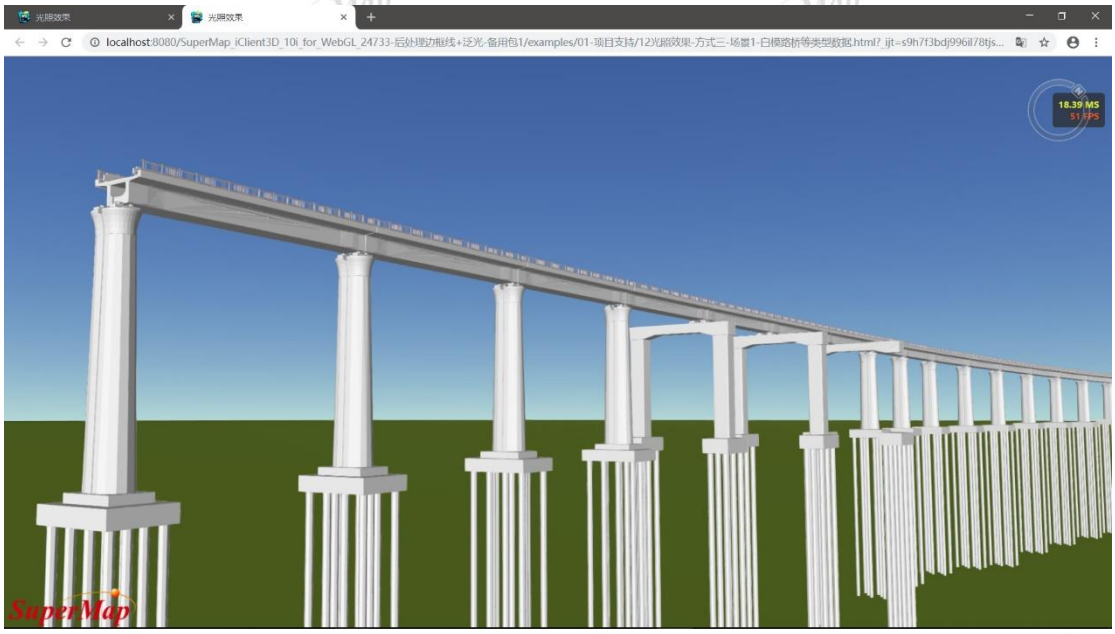
```
var gltf1 = viewer.entities.add({
    name: "gltf",
    position: new Cesium.Cartesian3.fromDegrees(xxx,yyy, 0),
    model: {
        uri: "../data（范例）/04-临时测试/Gltf/name.gltf"
    }
});
var imageBasedLightingFactor = new Cesium.Cartesian2(0.1, 0.1);
gltf1.model.imageBasedLightingFactor = imageBasedLightingFactor;
viewer.zoomTo(gltf1);
```

2.3 如何自定义场景中的光源效果

在 SuperMap iClient3D for WebGL 中，支持自定义设置光源，可以添加的光源类型有：平行光，点光源，聚光灯。

2.3.1 利用自定义的平行光优化白模的显示效果

如下图所示，通过添加自定义的平行光制作一个更具设计感的白模效果。



场景光线调整思路：

1. 环境光设置（由于模型本身没有纹理，只有颜色，环境光不宜太亮）；
2. 主光源平行光的设置（对于白模，使用侧平光更容易表现设计感）；
3. 最终场景的亮度，以能够显示最大范围的灰阶为好。

具体调整方法及参数设置：

1.修改默认光照的参数

关闭太阳光，调暗环境光，给一个较亮的大气层效果。

```
//默认场景设置
//设置太阳是否开启--关闭太阳光
scene.sun.show = false;
//设置环境光的强度
scene.lightSource.ambientLightColor = new Cesium.Color(0.5, 0.5, 0.5, 1);
//修改大气层的亮度
scene.skyAtmosphere.brightnessShift=0.4;
```

2.添加自定义的平行光光源

添加自定义的平行光光源。

```
// 添加光源
// 东南 45 方向
// 光源的倾角比较平，使得路面与桥墩的对比度比较高。
var position1 = new Cesium.Cartesian3.fromDegrees(xxx,yyy, 445);
```

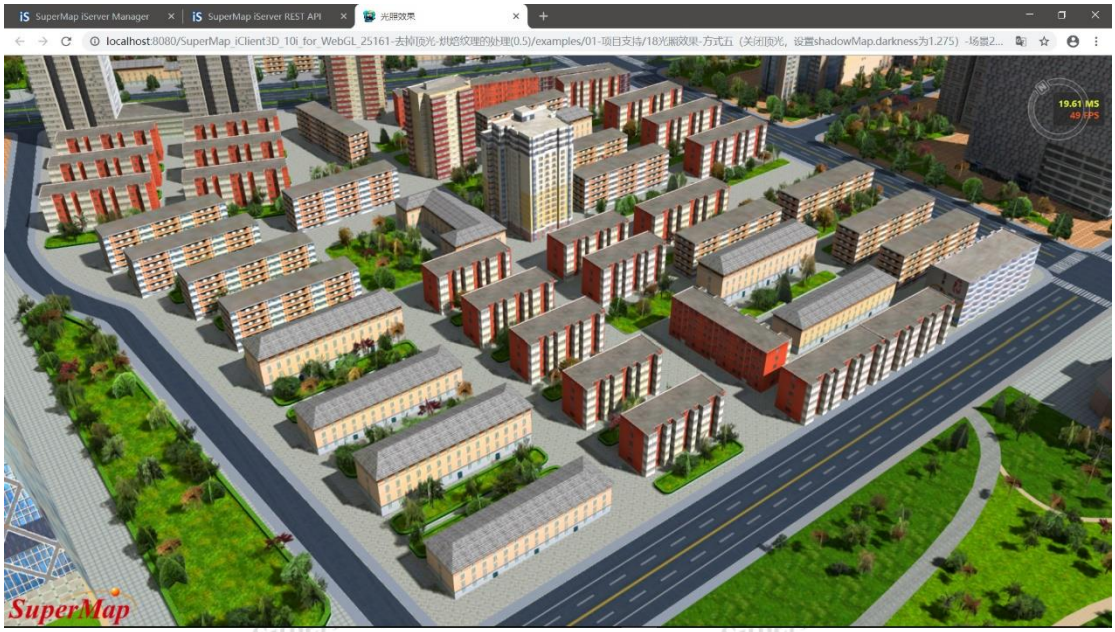
```
var targetPosition1 = new Cesium.Cartesian3.fromDegrees(xxz, yyz,430);
var dirLightOptions = {
    targetPosition: targetPosition1,
    color: new Cesium.Color(0.8, 0.8, 0.8, 1),
    intensity: 1
};
directionalLight_1 = new Cesium.DirectionalLight(position1, dirLightOptions);
scene.addLightSource(directionalLight_1);
```

其中，position1 为光源的位置，targetPosition1 为光源的目标点，二者决定了光线的角度。

2.3.2 利用自定义的平行光优化精模的显示效果

精模不同于白模，在还原纹理的基础上，更好的展现模型的立体效果。

如下图所示，以该场景为例进行说明：



场景光线调整思路：

1.环境光设置（由于模型本身带有明显的烘焙纹理，纹理本身的明暗效果已经比较明显，环境光只需要保证能够看清暗部的纹理即可）；

2.主光源平行光的设置（从模型本身的烘焙纹理中反推，数据在烘焙时的灯光设置：光源方位是东南侧，光线倾角相对较平。基于此，设置一个类似的光源，以保证数据的明暗效果相协调）

具体调整方法及参数设置：

1.修改默认光照的参数

关闭太阳光，稍微调暗环境光。

```
//默认场景设置
//设置太阳是否开启--关闭太阳光
scene.sun.show = false;
//设置环境光的强度
scene.lightSource.ambientLightColor = new Cesium.Color(0.65, 0.65, 0.65, 1);
```

```
//修改大气层的亮度
scene.skyAtmosphere.brightnessShift=0.4;
```

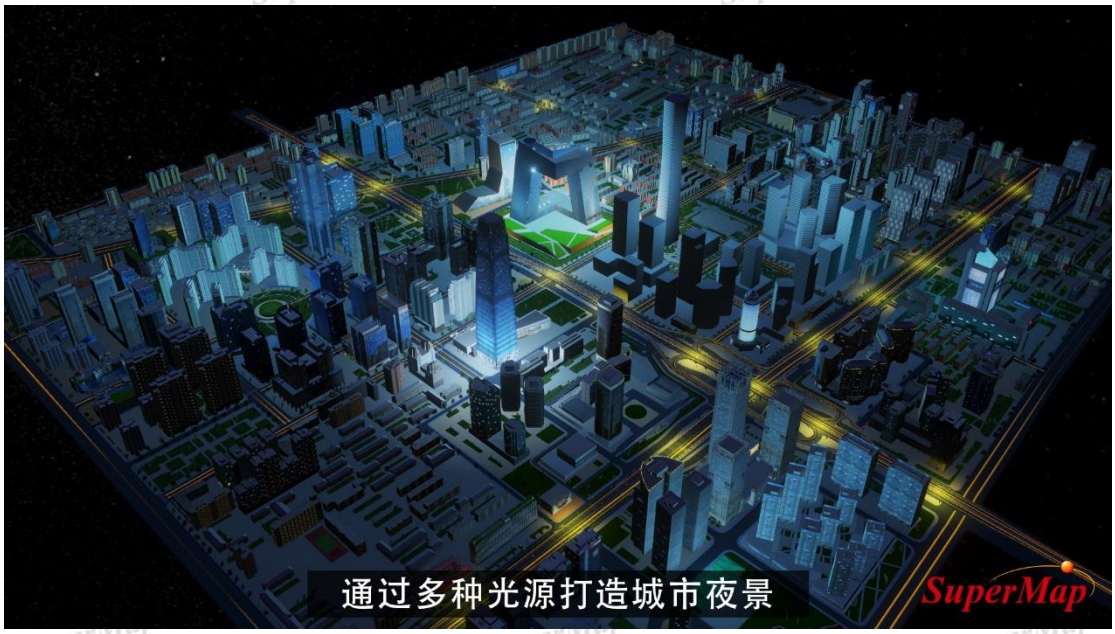
2.添加自定义的平行光光源

添加自定义的平行光光源。

```
// 添加光源
// 东南 45 方向  光线倾角 25° 左右
var position1 = new Cesium.Cartesian3.fromDegrees(xxx,yyy,, 480);
var targetPosition1 = new Cesium.Cartesian3.fromDegrees(xxz,yyz,,430);
var dirLightOptions = {
    targetPosition: targetPosition1,
    color: new Cesium.Color(1.0, 1.0, 1.0, 1),
    intensity: 0.55
};
directionalLight_1 = new Cesium.DirectionalLight(position1, dirLightOptions);
scene.addLightSource(directionalLight_1);
```

2.3.3 利用自定义光源制作城市夜景特效

如下图所示，进行说明。



1.因为是夜景，首先关闭太阳光与环境光。

```
scene.sun.show = false;
scene.lightSource.ambientLightColor = new Cesium.Color(0, 0, 0, 1);
```

2.给场景添加一个平行光和点光源作为底光

平行光位置不限，保证整个场景都能有一个偏蓝的色调。点光源的置于重点建筑群附近，提升该区域的亮度，作为整个场景的视觉中心区域。

```
// 新增直射光-整个环境
var dirLightOptions = {
```

```
    targetPosition: targetPosition1,
    color: new Cesium.Color(0.01, 0.01, 0.3, 1.0),
    intensity: 0.1
};
// 新增点光源-整个环境
var pointLightOptions3 = {
    cutoffDistance: 2000,
    color: new Cesium.Color(0.04, 0.18, 0.43, 1.0),
    intensity: 0.001
};
```

3.依次给重点建筑单独打造灯光效果

这里以场景中的最高楼-国贸大厦为例，在其周围添加多个点光源，并设置光源颜色为偏青蓝色的冷色调。点光源的高度大致位于 80 米处，以打亮建筑的中下部分。

并用同样的方法给其它重点建筑添加灯光效果。

```
// 新增点光源-建筑
var pointLightPoszuigao1 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 80.0);
var pointLightOptionszuigao1 = {
    cutoffDistance: 360.0,
    color: new Cesium.Color(0.15, 0.45, 1.4, 1.0),
    intensity: 0.12
};
pointLightzuigao1 = new Cesium.PointLight(pointLightPoszuigao1,
pointLightOptionszuigao1);
scene.addLightSource(pointLightzuigao1);
var pointLightPoszuigao2 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 80.0);
pointLightzuigao2 = new Cesium.PointLight(pointLightPoszuigao2,
pointLightOptionszuigao1);
scene.addLightSource(pointLightzuigao2);
var pointLightPoszuigao3 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 80.0);
pointLightzuigao3 = new Cesium.PointLight(pointLightPoszuigao3,
pointLightOptionszuigao1);
scene.addLightSource(pointLightzuigao3);
var pointLightPoszuigao4 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 80.0);
pointLightzuigao4 = new Cesium.PointLight(pointLightPoszuigao4,
pointLightOptionszuigao1);
scene.addLightSource(pointLightzuigao4);
```

4.给道路添加路灯效果。

前面添加的灯光都是针对建筑添加的，道路上面也会有很多的路灯。这里通过聚光灯来模拟路灯的效果，因为路灯具有明显的方向性。因为之前添加的灯光都是冷色调的，这里给路灯设置一个比较明显的暖色调，平衡场景中的颜色分布。因为场景是支持 HDR 的，所以可以给光源颜色设置大于 1 的 RGB 值(6, 5, 0.2, 1)。另外给光源设置一个合适的衰减系数

（3），保证灯光在向周围传播的时候逐渐减弱。

至此，灯光系统的添加基本完成。

```
// 新增聚光灯-横向道路-路灯 1
var spotLightPosludeng1_1 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 30);
var spotLightTargetPosludeng1_1 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 0);
var spotLightOtionsludeng1_1 = {
    color: new Cesium.Color(6, 5, 0.2, 1),
    distance: 100,
    decay: 3,
    intensity: 13,
    angle: Math.PI / 2
};
spotLightludeng1_1 = new Cesium.SpotLight(spotLightPosludeng1_1,
spotLightTargetPosludeng1_1, spotLightOtionsludeng1_1);
scene.addLightSource(spotLightludeng1_1);
// 新增聚光灯-横向道路-路灯 2
var spotLightPosludeng1_2 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 30);
var spotLightTargetPosludeng1_2 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 0);
spotLightludeng1_2 = new Cesium.SpotLight(spotLightPosludeng1_2,
spotLightTargetPosludeng1_2, spotLightOtionsludeng1_1);
scene.addLightSource(spotLightludeng1_2);
// 新增聚光灯-横向道路-路灯 3
var spotLightPosludeng1_3 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 30);
var spotLightTargetPosludeng1_3 = new Cesium.Cartesian3.fromDegrees(xxx, yyy, 0);
spotLightludeng1_3 = new Cesium.SpotLight(spotLightPosludeng1_3,
spotLightTargetPosludeng1_3, spotLightOtionsludeng1_1);
scene.addLightSource(spotLightludeng1_3);
```

注:

1.每种灯光的具体用法，可以参考在线范例:

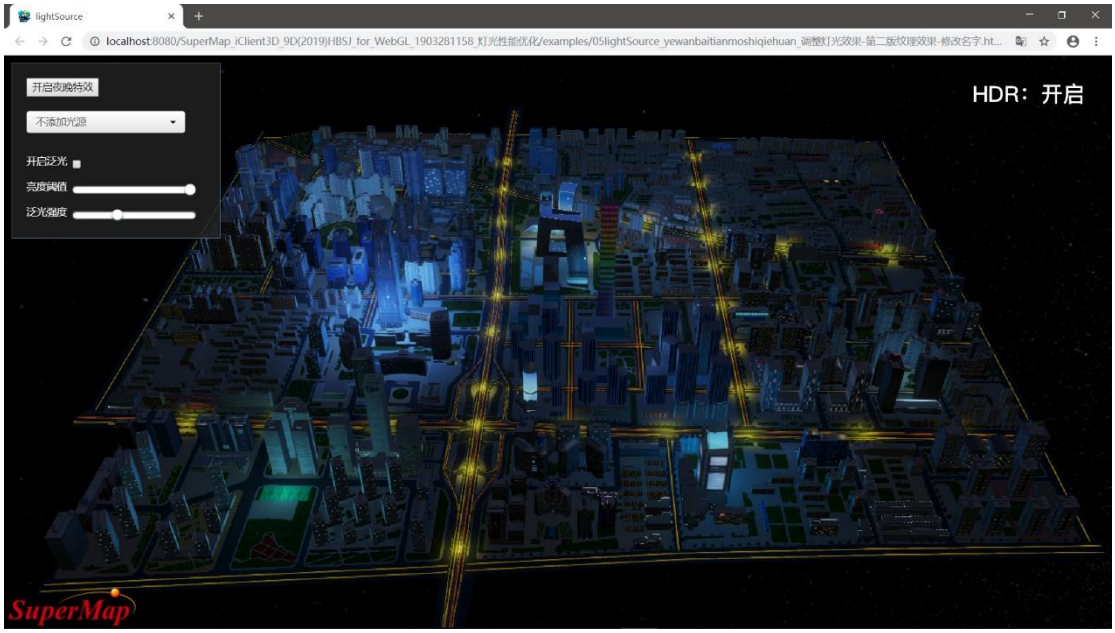
http://support.supermap.com.cn:8090/webgl/examples/editor.html#lightSource

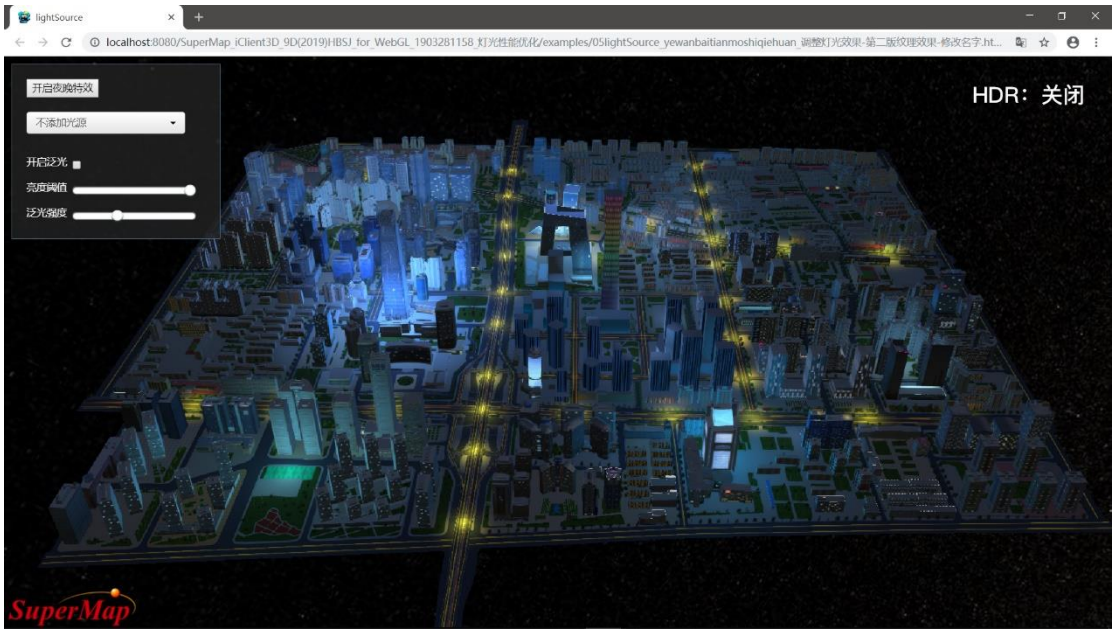
2.场景中的纹理自发光，泛光等其它特效可参考特效的制作流程文档。

2.4 HDR 对场景的影响

HDR 全称 High Dynamic Range(高动态范围图像),是采用更大的动态范围进行照明计算从而对画面图形进行渲染。如果没有 HDR，那么太暗的模型将会直接融入黑色环境，太亮的模型则直接显示为白色，缺少了色彩的渐变效果。HDR 可以保证亮的地方非常亮，暗的地方非常暗，同时亮暗部的细节都很明显。

只有在 HDR 模式下，场景中的光比才更加真实，光源及自发光纹理才能有正确的显示效果。比如，在有特效的场景中，通过设置多个光源及纹理自发光等，来表达不同区域之间丰富的明暗变化与光影效果，此时建议开启 HDR 模式。如果场景中只有太阳光，环境光等比较单一的光源，场景对象受光比较均匀，则没有必要开启 HDR。

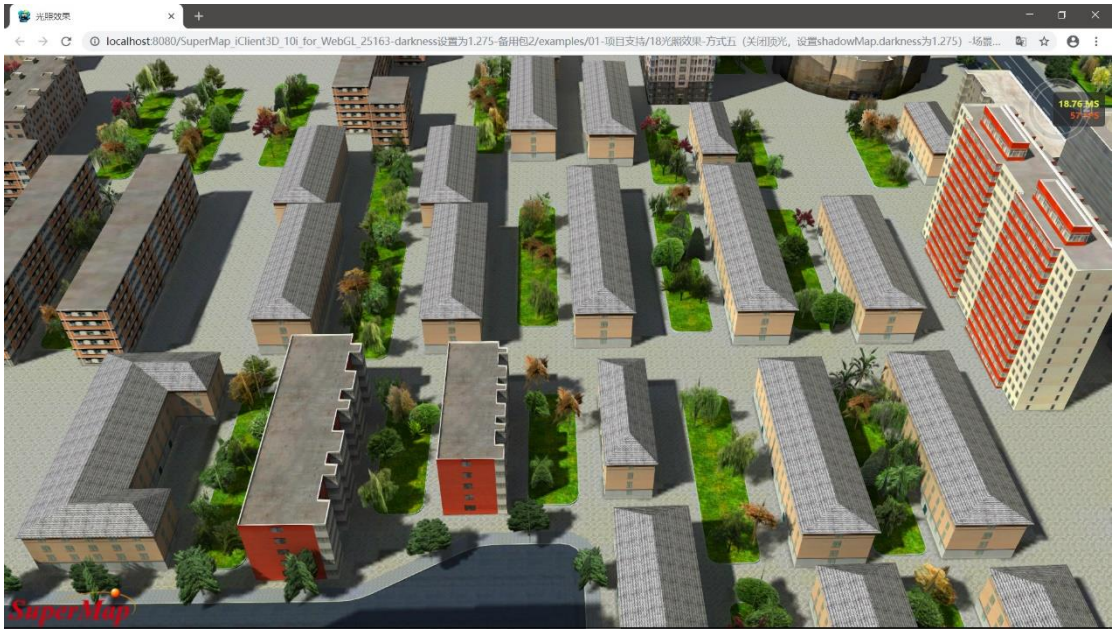


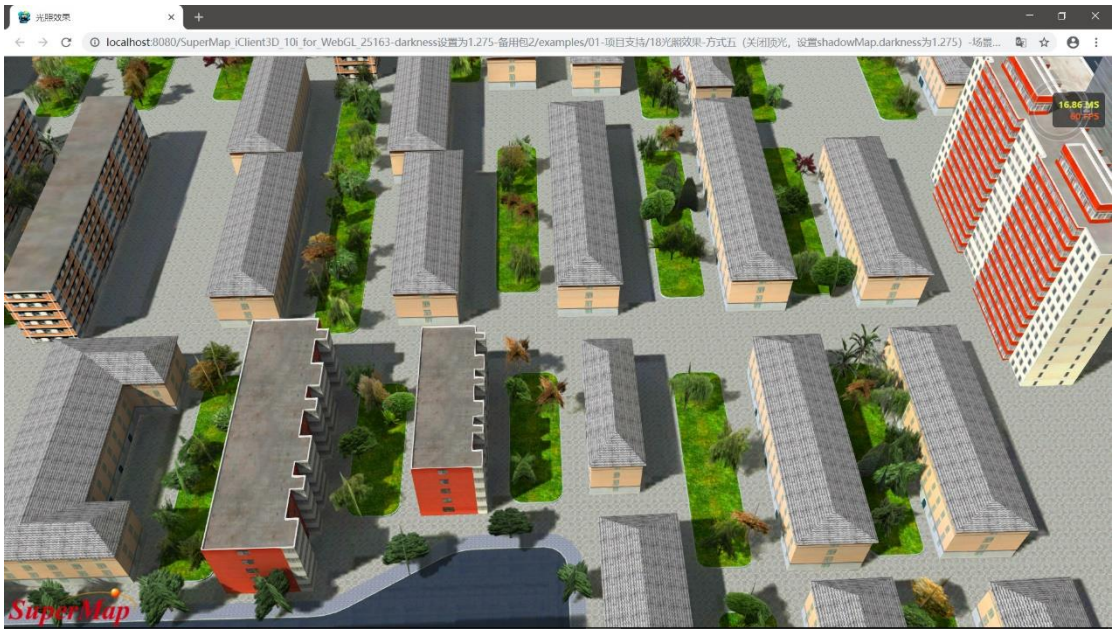


2.5 模型烘焙纹理的处理

有时会碰到带有两重纹理的模型，其中一层是常见的决定模型外观的漫反射纹理，另外一层则是制造模型光影明暗的烘焙纹理（一般通过 3ds Max 里面的烘焙功能制作）。在 SuperMap iClient3D for WebGL 中，提供了一个默认的参数（scene.shadowMap.darkness）来决定二者的叠加效果，来满足大多数的需求，如果想要自定义二者的叠加效果，可以对该参数进行修改。

如图所示，在同样的光照条件下，上图设置 darkness 的值为 1.0，整体场景的亮度稍低，并且色调偏黄偏红；下图设置 darkness 的值为 1.275，整体场景的亮度有所提升，并且消除了原有的偏色现象，与原始模型的效果相比，还原度更高。





2.6 场景/图层颜色的调整

2.6.1 场景颜色的调整

SuperMap iClient3D for WebGL 中，提供了用于修改场景颜色的一系列参数：亮度，对比度，色调，饱和度等。修改该参数时，场景中的所有对象均发生改变。

具体用法及效果可参考在线范例：

http://support.supermap.com.cn:8090/webgl/examples/editor.html#colorCorrection

2.6.2 图层颜色的调整

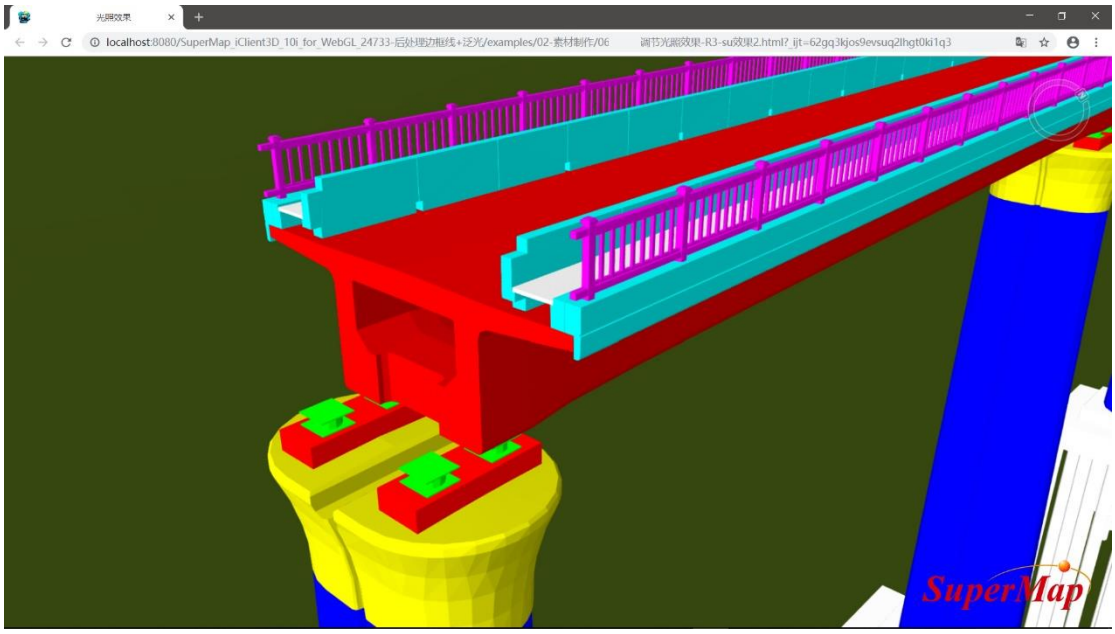
如果要对场景中的某一个或者某些图层单独做调整，可以使用图层颜色校正功能。如图所示，单独对树木图层的色相进行修改（色相值由 0 调整为-1），将秋天的树木“修改”为夏天的树木。设置代码如下：

```
var layer1 = scene.layers.find("c 树木 11640@鸟巢");
layer1.hue = -1;
layer1.brightness= 1;
layer1.contrast = 1;
```

2.7 通过开启硬件反走样优化模型边界效果

在 SuperMap iClient3D for WebGL 中支持开启硬件反走样，用于提升模型的显示效果，尤其是电力塔，栅栏，路灯等细长、有镂空的模型，优化效果更明显。硬件反走样开启方法：初始化 viewer 部件时，在 contextOptions 属性中，设置 requestWebgl2 为 true，msaaLevel 值为 2-8，具体代码设置如下。

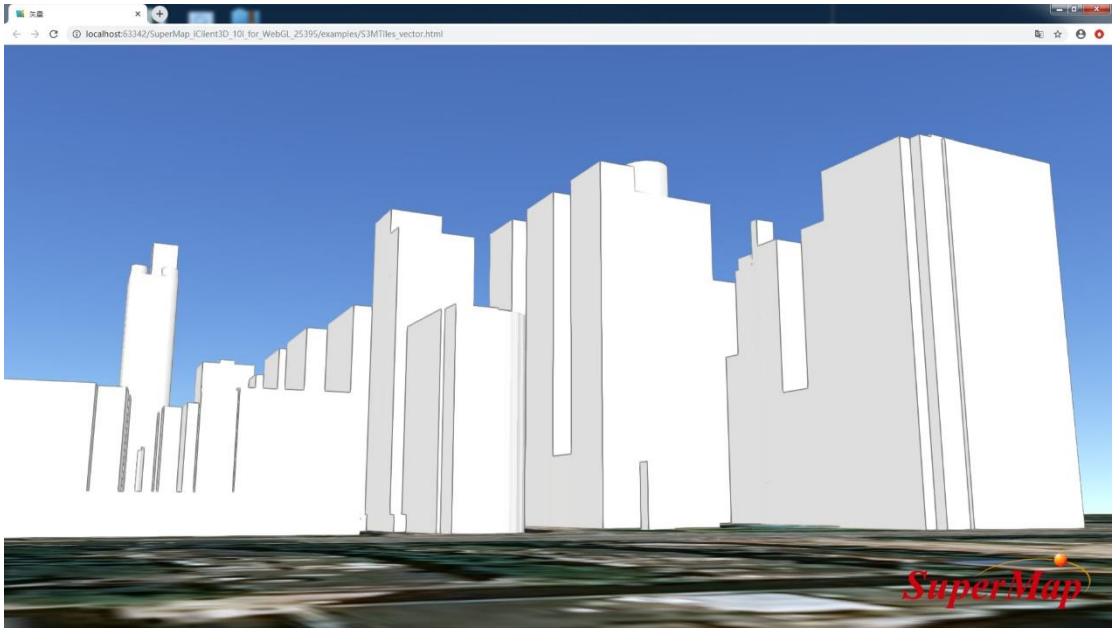
```
var viewer = new Cesium.Viewer('cesiumContainer',{
    //反走样
    contextOptions:{
        //硬件反走样，默认值为 1
        msaaLevel:8,
        requestWebgl2:true
    },
```



2.8 通过边框线优化模型显示效果

在设计类软件中，常常通过显示模型的边框线来优化模型的显示效果，使其更有设计感，因此在 SuperMap iClient3D for WebGL 上优化了线的绘制效果。这里以草图模式的边框线为例进行说明，默认线框的颜色为白色，并可根据需要修改颜色，代码设置如下。

```
layer.style3D.fillStyle = Cesium.FillStyle.Fill_And_WireFrame;
layer.wireFrameMode = Cesium.WireFrameType.Sketch;
layer.style3D.lineColor = new Cesium.Color(127/255, 127/255, 127/255, 1);
```


localhost:63342/SuperMap_iClient3D_10i_for_WebGL_25395/examples/S3MTiles_vector.html
SuperMap