郭隆邦 2023-02-04

20. Color颜色渐变插值

下面给大家介绍一个颜色对象`Color`颜色渐变插值方法`.lerpColors()`和`.lerp()`。

颜色对象`Color`颜色插值方法`.lerpColors()`

通过浏览器控制台测试`.lerpColors()`方法颜色插值的规律。

执行`.lerpColors(Color1,Color2, percent)`通过一个百分比参数`percent`，可以控制Color1和Color2两种颜色混合的百分比，Color1对应`1-percent`，Color2对应`percent`。

```
const c1 = new THREE.Color(0xff0000); //红色
const c2 = new THREE.Color(0x0000ff); //蓝色
const c = new THREE.Color();
```

颜色插值结果，和c1一样`rgb(1,0,0)`,100% c1 + 0% c2混合

```
c.lerpColors(c1,c2, 0);
console.log('颜色插值结果',c);
```

颜色插值结果`rgb(0.5,0,0.5)`，c1和c2各取50%：

```
c.lerpColors(c1,c2, 0.5);
console.log('颜色插值结果',c);
```

和c2一样rgb(0,0,1) 0% c1 + 100% c2混合

```
c.lerpColors(c1,c2, 1);
console.log('颜色插值结果',c);
```

颜色对象`Color`颜色插值方法`.lerp()`

`.lerp()` 和 `.lerpColors()` 功能一样，只是具体语法细节不同。

c1与c2颜色混合，混合后的rgb值，赋值给c1的 `.r` 、 `.g` 、 `.b` 属性。

```
const c1 = new THREE.Color(0xff0000); //红色
const c2 = new THREE.Color(0x0000ff); //蓝色
c1.lerp(c2, percent);
```

颜色克隆 `.clone()`

通过颜色对象克隆方法 `.clone()` 可以返回一个新的颜色对象。

注意 `c1.clone().lerp()` 和 `c1.lerp()` 写法是不同的，执行 `c1.clone().lerp()` ，c1和c2颜色混合后，不会改变c1的颜色值，改变的是 `c1.clone()` 返回的新颜色对象。

```
const c1 = new THREE.Color(0xff0000); //红色
const c2 = new THREE.Color(0x0000ff); //蓝色
const c = c1.clone().lerp(c2, percent);//颜色插值计算
```

`Color` 颜色插值应用

上节课"19.一段曲线颜色渐变"中颜色渐变是自己写的规则，这样比较麻烦，可以借助颜色插值方法 `Color.lerp()` 快速实现颜色渐变的计算。

```
const pos = geometry.attributes.position;
const count = pos.count; //顶点数量
// 计算每个顶点的颜色值
const colorsArr = [];
for (let i = 0; i < count; i++) {
    const percent = i / count;
    // 红色分量从0到1变化，蓝色分量从1到0变化
    colorsArr.push(percent, 0, 1 - percent); //蓝色到红色渐变色
}
```

```
// 根据顶点距离起点远近进行颜色插值计算
const c1 = new THREE.Color(0x00ffff); //曲线起点颜色 青色
const c2 = new THREE.Color(0xffff00); //曲线结束点颜色 黄色
for (let i = 0; i < count; i++) {
    const percent = i / count; //点索引值相对所有点数量的百分比
```

```
    //根据顶点位置顺序大小设置颜色渐变
    const c = c1.clone().lerp(c2, percent);//颜色插值计算
    colorsArr.push(c.r, c.g, c.b);
}
```

← 19. 一段曲线颜色渐变

21. 查看或设置gltf几何体顶点→