大镜(两个球体的交集),它的效果就会像一个物理上真实的放大镜效果!

5.6 超采样

超采样(supersampling)或多或少与子采样相反。在这种情况下,你需要的是准确性而不是性能。假设对应于两个相邻像素的射线击中了不同的物体。你将使用相应的颜色绘制每个像素。

但请记住我们开始时做的类比:每条射线都应该决定我们正在查看的"网格"中每个方块的"代表性"颜色。通过每个像素使用一条射线,我们可以任意决定穿过正方形中央的射线的颜色代表整个正方形,但这可能不是真实的。

解决这个问题的方法就是在每个像素上追踪更多的射线——4、9、16,你想要多少就追踪多少——然后对它们获得的颜色值求平均值来得到最终像素的颜色。

当然,这会让你的光线追踪渲染器速度变为原来的1/4、1/9或1/16,这和子采样让它快N倍的原因是一样的。幸运的是,有一个折中的方法。你可以假设物体的属性在它们的表面上平滑地变化,所以每个像素发射4条射线,在稍微不同的位置"击中"同一个物体可能不会对场景有太大改善。所以你可以从每个像素一条射线开始,然后比较相邻的射线:如果它们"击中"不同的物体,或者如果颜色的差异超过某个阈值,你可以对两者都应用像素细分。

5.7 总结

在本章中,我们简要介绍了几个你可以自己探索的想法。这些想法以新颖有趣的方式修改了我们一直在开发的基础光线追踪渲染器——使其更高效,能够表示更复杂的物体,或者以更接近我们的物理世界的方式对光线进行建模。

本书的第一部分已经证明了光线追踪渲染器是一款精美的应用程序,它可以使用简单、直观的算法和简单的数学运算生成令人惊叹的精美图像。

遗憾的是,这种纯粹是有代价的:性能。虽然有许多方法可以优化和并行化光