Shadow Map 是一种基于深度图（depth map）的阴影生成方法，由 Lance Williams 于 1978 年在文章“Casting curved shadows on curved surfaces”中首次提出。该方法的主要思想是：在第一遍渲染场景时，将场景的深度信息存放在纹理图片上，这个纹理图片称为深度图；然后在第二次渲染场景时，将深度图中的信息l**enth1**取出，和当前顶点与光源的距离**lenth2**做比较，如果**lenth1**小于**lenth2**，则说明当前顶点被遮挡处于阴影区，然后在片段着色程序中，将该顶点设置为阴影颜色。

13.1 什么是 depth map

深度图是一张**2D** 图片，每个像素都记录了从光源到遮挡物（遮挡物就是阴影生成物体）的距离，并且这些像素对应的顶点对于光源而言是**“可见的”**。这里的“可见”像素是指，以光源为观察点，光的方向为观察方向，设置观察矩阵并渲染所有遮挡物，最终出现在渲染表面上的像素。**Depth map** 中像素点记录的深度值记为**lenth1**；然后从视点的出发，计算物体顶点**v** 到光源的距离，记为**lenth2**；比较**lenth1**与**lenth2** 的大小，如果**lenth2 > lenth1**，则说明顶点**v** 所对应的 **depth texure** 上的像素点记录的深度值，并不是**v** 到光源的距离，而是**v** 和光源中间某个点到光源的距离，这意味着**“ v** 被遮挡”。

在一些教程中，往往将 depth map 翻译成阴影贴图（shdaow texture），这实在是一个误解，不光误解了两个名称，也混淆了 2 种阴影算法。

- 阴影贴图的英文为 **Shadow texture**，就是将日常所见的阴影保存为纹理图片；
- **Depth texture** 保存的是**“**从视点到物体顶点的距离，通常称为深度值”。

图左边的子图来自wikipid 上 shadow map 网页，请注意，下面表述为depth map；图右边则是一张普通的 shadow texture。

depth map 与 shadow texture 之对比

此外， Shadow texture 不但表示阴影贴图，也代表了一种阴影渲染方法，其实就是将阴影贴图作为纹理投影到物体上，投影的方法采用前面所讲述的 texture projective 方法。

13.2 Shadow map 与 shadow texture 的区别

在很多中文资料中，论述 Shadow map 技术时，容易将 Shadow map 与 shadow texture 这两个不同的概念混淆；

在英文中 map 有映射和图片的双重含义在内，shadow map 技术称为“shadow map”在英文中应该是准确的。中文翻译 shadow map 为阴影图，例如“实时计算机图形学第二版 153 页，第 6.12.4 节便将 shadow map 翻译为阴影图”，这种翻译已经是既成事实，那么我们也延续这种翻译方式。但是一定要知道“阴影图”和 shadow texture 所谓的阴影贴图是完全不同的两个概念。

- **Shadow map** 以 **depth map** 为技术基础，通过比较“光源可见点到光源的深度”和“任何点到光源的深度”来判断点是否被物体遮挡；
- 而 **shadow texture** 技术，将生成的阴影图形作为投影纹理来处理，也就是将一张阴影图投影映射到一个物体上（阴影接收体）。这种方法的缺点在于：设计者必须确认哪个物体是遮挡物，哪个物体是阴影接受体，并且不能产生自阴影现象（将一个物体的阴影贴图贴到物体身上，这是多么怪异）。

13.3 Shadow map 原理与实现流程

使用 Shadow Map 技术渲染阴影主要分两个过程：生成 **depth map**(深度图)和使用 **depth map** 进行阴影渲染。

生成 depth map 的流程为：

1. 以光源所在位置为相机位置，光线发射方向为观察方向进行相机参数设置；

2. 将世界视点投影矩阵 worldViewProjMatrix 传入顶点着色程序中，并在其中计算每个点的投影坐标，投影坐标的 Z 值即为深度值（将 Z 值保存为深度值只是很多方法中的一种）。在片段 shadow 程序中将深度值进行归一化，即转化到【0，1】区间。然后将深度值赋给颜色值（Cg 最的颜色值范围在 0-1 之间）。

3. 从 frame buffer 中读取颜色值，并渲染到一张纹理上，就得到了 depth map。注意：在实际运用中，如果遇到动态光影，则 depth map 通常是实时计算的，这就需要场景渲染两次，第一次渲染出 depth map，然后基于 depth map 做阴影渲染。

使用 depth map 进行阴影渲染的流程为：

1. 将纹理投影矩阵传入顶点着色程序中。注意，这个纹理投影矩阵，实际上就是产生深度图时所使用的 worldViewProjMatrix 矩阵乘上偏移矩阵（具体参见第 13 章），根据纹理投影矩阵，和模型空间的顶点坐标，计算投影纹理坐标和当前顶点距离光源的深度值lenth2（深度值的计算方法要和渲染深度图时的方法保持一致）。

2. 将 depth map 传入片段着色程序中，并根据计算好的投影纹理坐标，从中获取颜色信息，该颜色信息就是深度图中保存的深度值lenth1。

3. 比较两个深度值的大小，若lenth2 大于lenth1，则当前片断在阴影中；否则当前片断受光照射。

depth map 中保存的深度值到底是什么？

很多文献都将depth map 深度值解释成 Z Buffer 中的 Z 值，我对这种解释一直持怀疑态度！并不是说这种解释不对，而是指“这种解释有以偏概全的嫌疑”。我们通常所说的距离是指笛卡尔坐标空间中的欧几里得距离（Euclidean distance），Z 值本身并不是这个距离（参阅第 2.4.2 节），此外我在研究 GPU 算法的过程中，看到的关于depth map 中保存的深度值的计算方法远不止一种，有些直接计算顶点到视点的距离，然后归一化到【0，1】空间，同样可以有效的用于深度比较。由此可见，**epth map** 中保存的深度值，是衡量**“**顶点到视点的距离**”**相对关系的数据，计算深度值的重点在于**“**保证距离间相对关系的正确性**”**，至于采用什么样的计算方法倒在其次。

Shadow map 方法的优点：

可以使用一般用途的图形硬件对任意的阴影进行绘制，而且创建阴影图的代价与需要绘制的图元数量成线性关系，访问阴影图的时间也固定不变。此外，可以在基于该方法进行改进，创建软阴影效果。所谓软阴影就是光学中的半影区域。如果实时渲染软阴影，并运用到游戏中，是目前光照渲染领域的一个热门研究方向。

Shadow map 方法同样存在许多不足之处：

- 其一：阴影质量与阴影图的分辨率有关，所以很容易出现阴影边缘锯齿现象；
- 其二：深度值比较的精确度和正确性，有赖于 depth map 中像素点的数据精度，当生成深度图时肯定会造成数据精度的损失。要知道，深度值最后都被归一化到 0，1 空间中，所以看起来很小的精度损失也会影响数据比较的正确性，尤其是当两个点相聚非常近时，会出现 z-fighting 现象。所以往往在深度值上加上一个偏移量，人为的弥补这个误差；
- 其三：自阴影走样（Self-shadow Aliasing）,光源采样和屏幕采样通常并不一定在完全相同的位置，当深度图保存的深度值与观察表面的深度做比较时，其数值可能会出现误差，而导致错误的效果，通常引入偏移因子来避免这种情况；
- 其四：这种方法只适合于灯类型是聚光灯（Spot light ）的场合。如果灯类型是点光源（Point light）的话，则在第一步中需要生成的不是一张深度纹理，是一个立方深度纹理（cube texture）。如果灯类型是方向光（Directional light）的话,，则产生深度图时需要使用平行投影坐标系下的 worldViewProjMatrix 矩阵；