阴影对于创建逼真的图像以及为用户提供有关对象放置的视觉提示非常重要.本章重点介绍计算阴影的基本原理,并介绍最重要和最受欢迎的实时算法.我们还将简要讨论不太流行但体现重要原理的方法.在本章中,我们不会花时间讨论所有的选择和方法,因为有两本全面的书深入研究了阴影领域[**412,1902**].取而代之的是,我们专注于调查自发表以来出现的文章和演示,偏向于经过实际考验的技术.

在本章中使用的术语如图7.1所示,其中**遮挡体[occluder]**是将阴影投射到**接收体[receiver]**上的物体.精确光源,即那些没有面积的光源,只会产生完全阴影的区域,有时也称为**硬阴影[hard shadows]**.如果使用面光源或体积光源,则会产生柔和的阴影.这样,每个阴影都可以具有称为“**本影[umbra]**”的完全阴影区域和称为“**半影[penumbra]**”的部分阴影区域.软阴影可以通过其模糊的阴影边缘来识别.但是,重要的是要注意,通常仅通过低通滤镜模糊硬阴影的边缘无法正确渲染它们.从图7.2可以看出,正确的柔和阴影越接近阴影投射几何形状,就越清晰.柔和阴影的本影区域不等于由点光源产生的硬阴影.取而代之的是,在光源足够大且接收器距离遮挡物足够远的情况下,随着光源的变大,柔和阴影的本影区域的大小会减小,甚至可能消失.通常最好使用柔和的阴影,因为半阴影边缘会让观看者知道该阴影确实是阴影.硬边阴影通常看起来不太真实,有时可能被误解为实际的几何特征,例如表面上的折痕.但是,硬阴影比软阴影的渲染速度更快.

比半影更重要的是根本没有阴影.没有一些阴影作为视觉提示,场景通常令人难以置信,并且更难以感知.正如Wanger[1846]所言,阴影不准确通常比根本没有阴影要好,因为眼睛对阴影的形状相当宽容.例如,在地板上用作纹理的模糊黑圈可以将角色固定在地面上.

在以下各节中,我们将介绍这些简单的建模阴影,并介绍从场景中的遮挡物实时自动计算阴影的方法.第一部分处理在平面上投射阴影的特殊情况,第二部分介绍更通用的阴影算法,即将阴影投射到任意表面上.硬阴影和软阴影都将被覆盖.总之,提出了一些适用于各种阴影算法的优化技术.

**7.1 平面阴影** 2021年4月2日17点25分

当对象将阴影投射到平面上时,就会发生阴影的简单情况.本节介绍了几种用于平面阴影的算法,每种算法在阴影的柔和度和逼真度方面都有所不同.

**7.1.1 投射阴影** 2021年4月2日17点25分

在此方案中,第二次渲染三维对象以创建阴影.可以导出将物体的顶点投影到平面上的矩阵[**162,1759**].考虑图7.3中的情况,其中光源位于处,要投影的顶点在处,而投影的顶点在处.对于阴影平面为的特殊情况，我们将导出投影矩阵，然后将该结果推广到适用于任何平面. 我们首先得出坐标的投影.从图7.3左侧的相似三角形中,我们得到

以相同的方式获得坐标:,而坐标为零.现在,这些方程可以转换为投影矩阵:

验证很简单,这意味着确实是投影矩阵.

在一般情况下,应将阴影投射到任意平面不是平面.这种情况在图7.3的右边部分进行了描述.再次的目标是找到一个将投影到的矩阵.为此,在处发出的，穿过的光线与平面相交.得出投影点:

该方程也可以转换为投影式矩阵,如公式7.4所示,它满足:

如预期的那样,如果平面的,即且,则此矩阵变为公式7.2中的矩阵.

要渲染阴影,只需将此矩阵应用于以下对象即可:应该在平面上投射阴影,并用深色和无照明效果渲染此投影对象.在实践中,您必须采取措施避免允许将投影的三角形渲染在接收三角形的曲面下方.一种方法是在投影的平面上增加一些偏差,以使阴影三角形始终呈现在曲面的前面.

一种更安全的方法是先绘制地平面,然后在关闭深度测试的情况下绘制投影的三角形,然后照常渲染其余的几何图形.由于不进行深度比较,因此总是将投影的三角形绘制在地平面的顶部. 如果接地平面有一个限制(例如,它是一个矩形),则投影的阴影可能会落在其外部,从而破坏幻觉.为了解决这个问题,我们可以使用模板缓冲区.首先，将接收器拉到屏幕和模板缓冲区中.然后,在关闭z缓冲区的情况下,仅在绘制接收器的位置绘制投影的三角形,然后正常渲染场景的其余部分.

另一种阴影算法是将三角形渲染为纹理,然后将其应用于地平面.这种纹理是一种光照贴图,一种可以调节下层表面强度的纹理(第11.5.1节).可以看到,将阴影投影渲染为纹理的想法还允许半曲面和阴影出现在曲面上.该技术的一个缺点是纹理可以被放大,单个纹理像素覆盖多个像素,从而破坏了幻觉.

如果阴影情况在帧与帧之间没有变化,即光和脚轮之间没有相对移动,则可以重新使用此纹理.如果没有发生变化,则大多数阴影技术都可以受益于帧与帧之间重用中间计算结果.

所有的阴影投射器必须位于灯光和地面接收器之间.如果光源在对象的最高点以下,则会生成反阴影[162],因为每个顶点都投影在光源的点上.正确的阴影和反阴影如图7.4所示.如果我们在接收平面下方投射一个对象,也会发生错误,因为它也不应投射阴影.

当然可以明确地剔除和修整阴影三角形以避免此类伪影.接下来介绍的一种更简单的方法是,使用现有的GPU管道执行带有裁剪的投影.

**7.1.2 软阴影**

通过使用多种技术,也可以使投射阴影变柔和.在这里,我们描述了一种来自Heckbert和Herf[**697,722**]的算法,该算法产生柔和的阴影.该算法的目标是在显示柔和阴影的地平面上生成纹理.然后,我们描述精度较低,速度较快的方法.

只要光源有区域,就会出现柔和的阴影.估算区域光效果的一种方法是通过使用放置在其表面上的多个点光源对其进行采样.对于这些精确光源中的每一个,图像都被渲染并累积到缓冲区中.这些图像的平均值就是带有柔和阴影的图像.请注意,从理论上讲,任何生成硬阴影的算法都可以与这种累积技术一起使用,以产生半影.实际上,由于涉及执行时间,因此以交互速率进行操作通常是站不住脚的.

Heckbert和Herf使用基于视锥的方法来产生阴影.这个想法是将光视为观察者,并且地平面形成了视锥的远剪切平面.截锥体的宽度足以容纳封堵器.

通过生成一系列地平面纹理来形成柔和的阴影纹理.面光源在其表面上进行采样,每个位置都用于对代表地面的图像进行阴影处理,然后将投射阴影的对象投影到该图像上.将所有这些图像相加并取平均值,以生成底面阴影纹理.有关示例,请参见图7.5的左侧.

采样面积光方法的一个问题是它看起来像是这样:来自点光源的多个重叠阴影.同样,对于n次阴影遍历,只能生成n+1个不同的阴影.大量通过可以得到准确的结果,但是要付出高昂的代价.该方法对于获得(字面)“真实”图像以测试其他更快算法的质量很有用.

一种更有效的方法是使用卷积,即过滤.在某些情况下,模糊从单个点生成的硬阴影就足够了,并且可以生成可以与实际内容合成的半透明纹理.参见图7.6.但是,在物体与地面接触的位置附近,均匀的模糊可能无法令人信服.

还有许多其他方法可以提供更好的近似效果,但需要额外开销.例如,Haines[**644**]从投影的硬阴影开始,然后用从中央的暗到边缘的白色的渐变渲染轮廓边缘,以创建合理的半影.参见图7.5的右侧.但是,这些半影在物理上是不正确的,因为它们还应该扩展到轮廓边缘内的区域. Iwanicki[**356,806**]借鉴了球谐函数的思想,并用椭圆体近似遮盖字符以产生柔和的阴影.所有这些方法都有各种近似和缺点,但比平均大量阴影图像有效得多.

7.2 阴影和曲面 2021年4月2日17点46分

将平面阴影的概念扩展到曲面的一种简单方法是将生成的阴影图像用作投影纹理[1192、1254、1272、1597]。从光的角度考虑阴影。无论看见什么，都照亮；它看不到的东西在阴影中。假设从光源的角度来看，遮挡物被渲染为黑色，否则呈现白色纹理。然后可以将此纹理投影到要接收阴影的表面上。有效地，接收器上的每个顶点都有一个为其计算的（u，v）纹理坐标，并对其应用了纹理。这些纹理坐标可以由应用程序显式计算。这与上一部分中的地面阴影纹理有所不同，在前一部分中，对象被投影到特定的物理平面上。在这里，图像是从光的视角看的，就像放映机中的胶卷一样。渲染后，投影的阴影纹理会修改接收器表面。它也可以与其他阴影方法结合使用，有时主要用于帮助感知对象的位置。例如，在跳台游戏中，即使角色处于完全阴影状态，也可能总是在其正下方给主角一个投影[1343]。更精细的算法可以提供更好的结果。例如，Eisemann和D´ecoret [411]假设使用矩形头灯，并创建对象水平切片的阴影图像堆栈，然后将其转换为mipmap或类似图像。通过使用其mipmap，可以按与切片到接收器的距离成比例的方式访问每个切片的相应区域，这意味着距离较远的切片将投射更柔和的阴影。纹理投影方法有一些严重的缺点。首先，应用程序必须确定哪些对象是遮挡物，哪些是它们的接收者。程序必须将接收器保持在比遮挡器还远的地方，否则阴影会“向后投射”。此外，遮挡对象也无法遮挡自身。接下来的两个部分介绍了无需进行此类干预或限制即可生成正确阴影的算法。请注意，通过使用预构建的投影纹理可以获得各种照明图案。聚光灯只是一个正方形投影的纹理，其内部有一个圆圈定义了光源。可以通过由水平线组成的投影纹理来创建百叶帘效果。这种类型的纹理称为光衰减蒙版，Cookie纹理或图案映射。通过将两个纹理简单地相乘，可以将预构建的图案与动态创建的投影纹理相结合。在第6.9节中将进一步讨论此类灯。

7.3 阴影体积 2021年4月2日17点50分

由海德曼（Heidmann）在1991年提出[701]，一种基于Crow阴影体积的方法[311]可以通过巧妙地使用模板缓冲区将阴影投射到任意对象上。 它可以在任何GPU上使用，因为唯一的要求就是模板缓冲区。 它不是基于图像的（不同于下面描述的阴影贴图算法），因此避免了采样问题，从而在各处产生正确的清晰阴影。 有时这可能是一个缺点。 例如，角色的衣服可能会折叠起来，形成薄而坚硬的阴影，从而严重混叠。 由于影子卷的成本不可预测，因此如今很少使用它们[1599]。 我们在这里对算法进行简要说明，因为它说明了一些重要的原理，并在此基础上继续进行研究。

首先，想象一个点和一个三角形。将线从点延伸到三角形的顶点到无穷远会产生一个无限的三棱锥。三角形下面的部分（即不包括该点的部分）是截断的无限金字塔，而上部只是一个金字塔。如图7.7所示。现在想象一下，该点实际上是一个点光源。然后，位于截断的金字塔体积内（在三角形下方）的对象的任何部分都处于阴影中。该体积称为阴影体积。假设我们查看某个场景，然后从眼睛通过像素跟踪光线，直到光线照射到要在屏幕上显示的对象。当光线在到达该对象的途中时，每当光线穿过阴影体积的面向正面（即面向观察者）的面时，我们都会增加一个计数器。因此，每次光线进入阴影时，计数器都会增加。以相同的方式，每当光线穿过截顶的金字塔的背面时，我们就减少相同的计数器。然后，光线从阴影中射出。我们继续进行操作，递增和递减计数器，直到射线照射到要显示在该像素上的对象为止。如果计数器大于零，则该像素处于阴影中；否则，该像素处于阴影中。否则，不是。当有多个三角形投射阴影时，此原理也适用。参见图7.8。

用射线进行此操作很耗时。但是有一个更聪明的解决方案[701]：模板缓冲区可以为我们做计数。首先，清除模板缓冲区。其次，将整个场景仅使用未照明材质的颜色绘制到帧缓冲区中，以将这些阴影分量存储在颜色缓冲区中，并将深度信息获取到z缓冲区中。第三，关闭z缓冲区更新和写入颜色缓冲区（尽管仍进行z缓冲区测试），然后绘制阴影体积的正面三角形。在此过程中，将模板操作设置为在绘制三角形的任何地方增加模板缓冲区中的值。第四，使用模板缓冲区进行另一遍处理，这次仅绘制阴影体积的背面三角形。对于此过程，绘制三角形时，模版缓冲区中的值将减小。仅当渲染的阴影体积面的像素可见（即没有被任何实际几何图形隐藏）时，才执行递增和递减操作。此时，模板缓冲区将为每个像素保留阴影状态。最后，再次渲染整个场景，这次仅使用受光影响的活性材料的成分，并仅在模板缓冲区中的值为0的位置显示。值为0表示光线已熄灭进入阴影体积的阴影的数量是多少倍，即该位置被灯光照亮。这种计数方法是阴影量背后的基本思想。由阴影体积算法生成的阴影示例如图7.9所示。有有效的方法可以在单遍中实现算法[1514]。但是，当物体穿透相机的近平面时，就会发生计数问题。称为z-fail的解决方案包括计算隐藏在可见表面后面而不是前面的交叉点[450，775]。图7.8给出了这种选择的简要概述。

为每个三角形创建四边形会产生大量的透支。也就是说，每个三角形将创建三个必须渲染的四边形。由一千个三角形组成的球体将创建三千个四边形，并且这些四边形中的每一个都可以跨越屏幕。一种解决方案是沿着对象的轮廓边缘仅绘制那些四边形，例如，我们的球体可能仅具有五十个轮廓边，因此仅需要五十个四边形。几何着色器可用于自动生成此类轮廓边缘[1702]。剔除和夹紧技术还可用于降低填充成本[1061]。但是，阴影体积算法仍然有一个可怕的缺点：极端的可变性。想象一下一个单一的小三角形。如果摄像机和光源的位置完全相同，则阴影体积的成本将降至最低。所形成的四边形不会覆盖任何像素，因为它们在视图上处于边缘位置。只有三角形本身很重要。假设观众现在绕着三角形旋转，并保持在可见状态。随着摄像机远离光源，阴影体积的四边形将变得更加可见，并覆盖更多的屏幕，从而导致更多的计算发生。如果观看者碰巧进入了三角形的阴影，则阴影体积将完全填满屏幕，与原始视图相比，需要花费大量时间进行评估。这种可变性使阴影体积无法在交互式应用中使用，在这些应用中，一致的帧速率很重要。与其他情况一样，朝着光线看可能会导致算法成本发生巨大且无法预测的跳跃。由于这些原因，影子卷在很大程度上已被应用程序所放弃。但是，鉴于在GPU上访问数据的新方法和新方法的不断发展，以及研究人员对此类功能的巧妙运用，影子卷可能有一天会重新投入使用。例如，Sintorn等。 [1648]给出了阴影体积算法的概述，这些阴影体积算法提高了效率并提出了自己的分层加速结构。提出的下一个算法阴影映射具有可预测的成本，并且非常适合GPU，因此构成了许多应用程序中阴影生成的基础。

7.4 阴影映射 2021年4月2日18点18分

1978年，Williams [1888]提出可以使用基于z缓冲区的通用渲染器在任意对象上快速生成阴影。这个想法是使用z缓冲区从要投射阴影的光源位置渲染场景。无论“看到”的光是什么，其余的都在阴影中。生成此图像时，仅需要z缓冲。可以关闭照明，纹理化和将值写入颜色缓冲区。现在，z缓冲区中的每个像素都包含最靠近光源的对象的z深度。我们将z缓冲区的全部内容称为阴影图，有时也称为阴影深度图或阴影缓冲区。要使用阴影贴图，将第二次渲染场景，但这是相对于查看器而言的。渲染每个绘图图元时，会将其在每个像素处的位置与阴影贴图进行比较。如果渲染的点比阴影贴图中的对应值离光源更远，则该点在阴影中，否则就不在阴影中。通过使用纹理映射来实现此技术。参见图7.10。阴影映射是一种流行的算法，因为它是相对可预测的。建立阴影贴图的成本大致与渲染图元的数量成线性关系，并且访问时间是恒定的。阴影贴图可以一次生成，并且可以在光线和对象不移动的场景（例如计算机辅助设计）的每一帧中重复使用。当生成单个z缓冲区时，灯光只能像摄像机一样在特定方向上“看”。对于诸如太阳之类的遥远定向光，该光的视图设置为包含所有将阴影投射到眼睛所见的可视对象中的对象。光线使用正交投影，并且其视线需要在x和y方向上足够宽和足够高才能查看这组对象。本地光源需要尽可能进行类似的调整。如果局部光距离阴影对象足够远，则单个视锥就足以包含所有这些内容。或者，如果局部光是聚光灯，则它具有与之相关的自然视锥，并且视锥外的所有内容都不会被照亮。

如果本地光源在场景内部并且被阴影投射器包围，则典型的解决方案是使用六视图立方体，类似于立方体环境映射[865]。这些称为全向阴影贴图。全向贴图的主要挑战是如何避免在两个单独的贴图相遇的接缝处出现伪影。 King和Newhall [895]深入分析了问题并提供了解决方案，而Gerasimov [525]提供了一些实施细节。 Forsyth [484，486]提出了一种用于全向光的通用多视锥分割方案，该方案还在需要时提供了更多的阴影贴图分辨率。 Crytek [1590，1678，1679]根据每个视点投影的视锥的屏幕空间覆盖范围，为点光源设置六个视点的分辨率，所有地图都存储在纹理地图集中。并非场景中的所有对象都需要渲染到光源的视区中。首先，仅需要渲染可以投射阴影的对象。例如，如果已知地面只能接收阴影而不能投射阴影，则不必将其渲染到阴影贴图中。

从定义上讲，阴影投射器是指位于灯光视锥中的角色。可以通过多种方式增加或收紧该截锥体，从而使我们能够安全地忽略某些影子脚轮[896，1812]。考虑一下肉眼可见的阴影接收器组。这组对象在沿光源的视线方向的最大距离内。超出此距离的任何物体都不能在可见的接收器上蒙上阴影。同样，可见接收器的集合可能会比光源的原始x和y视图范围小。见图7.11。另一个例子是，如果光源在眼睛视锥内部，则在此视锥之外的任何物体都不能在接收器上投射阴影。仅渲染相关对象不仅可以节省渲染时间，还可以减小光源的视锥所需的大小，因此可以提高阴影贴图的有效分辨率，从而提高质量。此外，如果视锥台面的近平面距离光源越远越好，并且远平面越近越好，这也有帮助。这样做可以提高z缓冲区的有效精度[1792]（第4.7.2节）。阴影映射的一个缺点是阴影的质量取决于阴影图的分辨率（以像素为单位）以及z缓冲区的数值精度。由于阴影图是在深度比较期间采样的，因此该算法容易出现混叠问题，尤其是在对象之间的接触点附近。一个常见的问题是自阴影混叠，通常被称为“表面粉刺”或“阴影粉刺”，其中三角形被错误地认为会对其自身进行阴影。此问题有两个来源。一种就是处理器精度的数值极限。另一个来源是几何图形，这是因为使用了点样本的值来表示区域的深度。也就是说，为光生成的样本几乎永远不会与屏幕样本位于相同的位置（例如，像素经常在其中心进行采样）。将灯光的存储深度值与查看的曲面的深度进行比较时，灯光的值可能会略低于曲面的深度，从而导致自阴影。这种错误的影响如图7.12所示。

帮助避免（但不总是消除）各种阴影贴图伪像的一种常用方法是引入偏差因子。在检查阴影图中发现的距离与被测位置的距离时，会从接收器的距离中减去一个小的偏差。见图7.13。该偏差可能是一个恒定值[1022]，但是当接收器未完全面向光线时，这样做可能会失败。一种更有效的方法是使用与接收器相对于光的角度成比例的偏置。表面远离光的倾斜程度越大，偏差就越大，可以避免此问题。这种类型的偏差称为斜率比例偏差。可以通过使用诸如OpenGL的glPolygonOffset之类的命令来施加两个偏差，以使每个多边形远离光线。请注意，如果表面直接面对光，则坡度比例偏差根本不会将其向后偏压。因此，将恒定偏差与斜率比例偏差一起使用以避免可能的精度误差。斜率比例偏差通常也被限制在某个最大值，因为当从光的角度看表面接近于边缘时，切线值可能会非常高。

Holbert [759，760]引入了法线偏移偏置，该偏移首先使接收器的世界空间位置沿表面法线方向稍微偏移一点，与光的方向和几何法线之间的角度的正弦成正比。请参阅第250页的图7.24。这不仅会改变深度，还会改变在阴影贴图上测试样本的x坐标和y坐标。随着光的角度向表面变浅，此偏移量增加，希望样品在表面上方足够远以避免自遮蔽。将样品移至接收器上方的“虚拟表面”时，该方法可以可视化。该偏移量是一个世界空间距离，因此Pettineo [1403]建议根据阴影图的深度范围对其进行缩放。 Pesce [1391]提出了沿摄像机视图方向偏置的想法，该想法也可以通过调整阴影贴图坐标来实现。在第7.5节中讨论了其他偏置方法，因为此处介绍的阴影方法还需要测试几个相邻的样本。太大的偏斜会引起称为漏光或彼得潘宁（Peter Panning）的问题，其中物体似乎漂浮在下方表面上方。之所以会出现这种伪影，是因为物体接触点下方的区域（例如脚下的地面）被向前推得太远，因此没有阴影。避免自遮蔽问题的一种方法是仅将背面渲染为阴影贴图。这种方案称为第二深度阴影映射[1845]，在许多情况下都可以很好地工作，特别是对于不能手动调整偏差的渲染系统尤其如此。当对象是双面的，薄的或彼此接触时，就会出现问题情况。如果对象是网格的两面都可见的模型（例如，棕榈叶或纸），则由于背面和正面位于相同的位置，因此会发生自阴影。同样，如果不执行任何偏压操作，则轮廓边缘或薄物体附近可能会出现问题，因为在这些区域中，背面接近正面。增加一个偏压可以帮助避免表面痤疮，但是该方案更容易漏光，因为接收器和封堵器背面之间在接触点之间没有分隔。见图7.14。选择哪种方案可能取决于情况。例如，Sousa [1679]发现使用正面作为太阳阴影，背面使用为室内照明最适合其应用。

请注意，对于阴影贴图，对象必须是“水密的”（流形和封闭的，即实心；第16.3.3节），或者必须在地图上同时显示正面和背面，否则对象可能无法完全投射阴影。 Woo [1900]提出了一种通用方法，该方法试图在字面上仅使用正面或背面进行阴影之间成为一种快乐的媒介。这个想法是将实体对象渲染到阴影贴图，并跟踪两个最接近光照的表面。可以通过深度剥离或其他与透明度相关的技术来执行该过程。两个对象之间的平均深度形成一个中间层，该中间层的深度用作阴影图，有时也称为双重阴影图[1865]。如果物体足够厚，则自阴影和漏光伪影会降至最低。 Bavoil等人[116]讨论了解决潜在伪像的方法，以及其他实现细节。主要缺点是与使用两个阴影贴图相关的额外费用。 Myers [1253]讨论了遮挡物和接收器之间由艺术家控制的深度层。随着观看者的移动，灯光的观看量通常会随着阴影脚轮的设置而改变大小。这样的变化又会导致阴影在各帧之间略有偏移。发生这种情况是因为光源的阴影贴图采样的是与光源不同的一组方向，并且这些方向与前一组方向不一致。对于定向光，解决方案是强制生成的每个后续阴影贴图在世界空间中维持相同的相对纹理像素光束位置[927、1227、1792、1810]。也就是说，您可以将阴影贴图视为在整个世界上施加二维网格化参照系，每个网格单元代表贴图上的一个像素样本。在您移动时，会为这些相同的网格单元的不同集合生成阴影图。换句话说，光线的视线投影被强制移至此网格以保持帧与帧之间的连贯性。

7.4.1 分辨率增强 2021年4月2日18点27分

与使用纹理的方式类似，理想情况下，我们希望一个阴影贴图纹理覆盖大约一个图像像素。如果我们的光源与眼睛位于同一位置，则阴影贴图与屏幕空间像素完美地一对一映射（并且没有可见的阴影，因为光线恰好照亮了眼睛所看到的）。光线的方向一改变，此每像素比率就会改变，这可能会导致伪影。一个例子如图7.15所示。由于前景中的大量像素与阴影贴图的每个纹理像素相关联，因此阴影块状且定义不清。这种不匹配称为透视混叠。如果一个表面接近光线边缘，但面向观察者，则单个阴影贴图纹理像素也可以覆盖许多像素。这个问题被称为投影混叠[1792]。见图7.16。可以通过增加阴影贴图分辨率来降低块状性，但要以增加内存和处理为代价。还有另一种方法来创建灯光的采样图案，使其更接近于相机的图案。这是通过更改场景向光投射的方式来完成的。通常，我们认为视图是对称的，视图矢量位于视锥的中心。然而，观看方向仅定义了观看平面，而不定义了哪些像素被采样。定义视锥的窗口可以在此平面上移动，倾斜或旋转，从而创建四边形，从而提供不同的世界视图空间映射。四边形仍按固定间隔进行采样，因为这是线性变换矩阵的性质以及GPU对其的使用。可以通过更改光源的查看方向和查看窗口的范围来修改采样率。见图7.17。

将光线的视线映射到眼睛的视线有22个自由度[896]。对这种解决方案空间的探索导致了几种不同的算法，这些算法试图使光的采样率与眼睛的采样率更好地匹配。方法包括透视阴影图（PSM）[1691]，梯形阴影图（TSM）[1132]和光空间透视阴影图（LiSPSM）[1893、1895]。有关示例，请参见第254页的图7.15和图7.26。此类技术称为透视变形方法。这些矩阵变形算法的一个优点是，除了修改光源矩阵外，不需要其他工作。每种方法都有其优点和缺点[484]，因为每种方法都可以帮助在某些几何形状和光照情况下匹配采样率，而在另一些情况下则会降低采样率。劳埃德（Lloyd）等人[1062，1063]分析了PSM，TSM和LiSPSM之间的等效性，从而很好地概述了这些方法的采样和混叠问题。当光线的方向垂直于视图的方向（例如头顶）时，这些方案最有效，因为可以移动透视变换以使更多的样本更靠近眼睛。

矩阵变形技术无法提供帮助的一种照明情况是，当光线在相机前面并指向相机时。这种情况被称为决斗挫折，或更通俗地说是“前灯鹿”。靠近眼睛需要更多阴影贴图样本，但是线性扭曲只会使情况变得更糟[1555]。诸如此类的问题和其他问题，例如质量的突然变化[430]和相机运动过程中产生的阴影的“紧张”，不稳定的质量[484，1227]，使这些方法不受欢迎。在查看器所在的位置添加更多样本的想法是一个好主意，导致产生针对给定视图生成多个阴影图的算法。当Carmack在2004年Quakecon上的主题演讲中描述了这一想法时，它首先产生了明显的影响。Blow独立地实现了这种系统[174]。这个想法很简单：生成一组固定的阴影贴图（可能以不同的分辨率），覆盖场景的不同区域。在Blow的方案中，在观察者周围嵌套了四个阴影贴图。这样，高分辨率地图可用于附近的对象，而远处的那些对象的分辨率会下降。 Forsyth [483，486]提出了一个相关的想法，为不同的可见对象集生成不同的阴影贴图。在他的设置中避免了如何处理跨越两个阴影贴图之间的边界的对象的过渡问题，因为每个对象都有一个且只有一个与之关联的阴影贴图。旗舰工作室开发了一个融合了这两个想法的系统。一个阴影贴图用于附近的动态对象，另一个阴影贴图用于在查看器附近的静态对象的网格部分，第三阴影贴图用于整个场景中的静态对象。每帧生成第一个阴影图。由于光源和几何形状是静态的，因此其他两个只能生成一次。尽管所有这些特定系统现在都已经很老了，但是针对不同对象和情况的多个映射（有些是预先计算的，有些是动态的）的想法是此后开发的算法中的一个共同主题。

2006年，恩格尔[430]，劳埃德（Lloyd）等人[1062，1063]和张等人（1962，1963）独立研究了相同的基本思想。 想法是将视锥的体积平行于视线方向切成几部分。 见图7.18。 随着深度的增加，每个连续的体积的深度范围大约是先前体积的深度范围的两到三倍[430，1962]。 对于每个视图体积，光源都可以形成一个将视锥体紧紧包围的视锥，然后生成一个阴影贴图。 通过使用纹理地图集或阵列，可以将不同的阴影贴图视为一个大的纹理对象，从而最大程度地减少了缓存访问延迟。 图7.19显示了获得的质量改进的比较。 恩格尔（Engel）将此算法称为级联阴影贴图（CSM），比张的术语平行分割阴影贴图更常用，但两者都出现在文献中并且实际上是相同的[1964]。

这种类型的算法易于实现，可以覆盖巨大的场景区域并具有合理的结果，并且很健壮。可以通过以更接近眼睛的较高速率进行采样来解决决斗的视锥问题，并且没有严重的最坏情况问题。由于这些优点，在​​许多应用程序中使用了级联阴影映射。尽管可以使用透视变形将更多样本打包到单个阴影贴图的细分区域中[1783]，但规范是为每个级联使用单独的阴影贴图。如图7.18所示，图7.20从查看者的角度显示，每张地图所覆盖的区域可以变化。较近的阴影贴图的较小视图体积在需要的地方提供了更多样本。确定如何在地图之间划分z深度的范围（称为z分区的任务）可能非常简单，也可能涉及[412，991，1791]。一种方法是对数划分[1062]，其中每个级联图的远近平面距离之比都相同：

其中n和f是整个场景的近平面和远平面，c是地图的数量，r是结果比率。例如，如果场景的最近对象是1米远，最大距离是1000米，并且我们有三个级联贴图，则r = p3 1000/1 =10。最接近视图的近平面和远平面距离将是1和10，则下一个间隔为10到100以保持该比率，最后一个间隔为100到1000米。初始接近深度对该分区影响很大。如果近深度只有0.1米，则10000的立方根为21.54，这是一个相当高的比率，例如0.1到2.154到46.42到1000。这意味着生成的每个阴影贴图必须覆盖更大的区域，从而降低了其精度。 。在实践中，这样的划分为接近近平面的区域提供了相当大的分辨率，如果该区域中没有任何对象，这将被浪费。避免这种不匹配的一种方法是将分区距离设置为对数分布和等距分布的加权混合[1962，1963]，但是如果我们可以确定场景的紧密视域，则更好。

挑战在于设置近平面。如果距离眼睛太远，则该平面可能会夹住物体，这是非常糟糕的伪影。对于剪辑场景，艺术家可以预先精确设置此值[1590]，但是对于交互式环境，此问题更具挑战性。 Lauritzen等人[991，1403]提出了样本分布阴影图（SDSM），该图使用前一帧的z深度值通过两种方法之一确定更好的分区。第一种方法是浏览z深度的最小值和最大值，并使用这些深度设置近平面和远平面。这是通过在GPU上执行所谓的reduce操作来执行的，在该操作中，计算或其他着色器将分析一系列越来越小的缓冲区，并将输出缓冲区作为输入进行反馈，直到剩下1×1缓冲区为止。通常，将这些值推出一点以调整场景中对象移动的速度。除非采取纠正措施，否则从屏幕边缘进入的附近物体可能仍会导致帧问题，尽管下一帧将很快得到纠正。第二种方法还分析了深度缓冲区的值，制作了一个称为直方图的图表，该图表记录了z深度在范围内的分布。除了查找紧密的近平面和远平面之外，在根本没有对象的位置，图形中可能还会有间隙。通常添加到该区域的任何分区平面都可以捕捉到实际存在对象的位置，从而为级联映射集提供更高的z深度精度。

在实践中，第一种方法是通用的，快速的（通常在每帧1 ms范围内），并且效果良好，因此已在多种应用中被采用[1405，1811]。 见图7.21。 与单个阴影贴图一样，由于光样本逐帧移动而造成的闪烁伪影是一个问题，并且随着对象在级联之间移动而变得更糟。 可以使用多种方法来维护世界空间中的稳定采样点，每种方法都有自己的优势[41，865，1381，1403，1678，1679，1810]。 当对象跨越两个阴影贴图之间的边界时，阴影的质量可能会突然发生变化。 一种解决方案是使视图体积稍微重叠。 在这些重叠区域中获取的样本从两个相邻的阴影贴图中收集结果并进行混合[1791]。 或者，可以通过使用抖动[1381]在这样的区域中采集单个样本。

由于它的流行，人们已经投入了大量的精力来提高效率和质量[1791，1964]。如果阴影贴图的视锥内部没有任何变化，则无需重新计算该阴影贴图。对于每种光线，可以通过找到哪些对象对光线可见以及其中哪些可以在接收器上投射阴影来预先计算阴影投射器的列表[1405]。由于很难理解阴影是否正确，因此可以采用一些适用于级联和其他算法的快捷方式。一种技术是使用低级别的细节模型作为代理来实际投射阴影[652，1812]。另一个方法是从考虑中移除小型堵塞器[1381，1811]。从理论上讲，这样的阴影不太重要，距离较远的阴影贴图的更新频率可能比一帧更新一次的频率低。这个想法冒着由大型运动物体引起的伪影的风险，因此需要谨慎使用[865，1389，1391，1678，1679]。 Day [329]提出了在帧与帧之间“滚动”远距离贴图的想法，即每个静态阴影贴图的大部分都是可在帧与帧之间重用的，并且只有条纹可能会改变，因此需要渲染。 DOOM（2016）等游戏维护着大量的阴影贴图集，仅重新生成对象已移动的那些贴图[294]。可以将更远的级联地图设置为完全忽略动态对象，因为这样的阴影可能对场景的贡献很小。在某些环境中，可以使用高分辨率的静态阴影贴图代替这些更远的级联，从而可以大大减少工作量[415，1590]。稀疏纹理系统（第19.10.1节）可用于单个静态阴影贴图非常巨大的世界[241、625、1253]。级联阴影贴图可以与烘焙的光照贴图纹理或其他更适合特定情况的阴影技术结合使用[652]。 Valient的演讲[1811]值得注意，因为它描述了针对各种视频游戏的不同阴影系统自定义设置和技术。第11.5.1节详细讨论了预先计算的光和阴影算法。创建多个单独的阴影贴图意味着要遍历每个几何贴图集。在一次通过将遮挡物渲染到一组阴影贴图的想法的基础上，已经建立了许多提高效率的方法。几何着色器可用于复制对象数据并将其发送到多个视图[41]。实例化的几何着色器允许将一个对象输出到多达32个深度纹理[1456]。多视口扩展可以执行诸如将对象渲染到特定纹理阵列切片[41、154、530]之类的操作。第21.3.1节在将它们用于虚拟现实的背景下，将对它们进行更详细的讨论。视口共享技术的一个可能的缺点是，生成的所有阴影贴图的遮挡物必须沿着管道发送，而不是与每个阴影贴图相关的集合[1791、1810]。

您自己当前正处在全球数十亿光源的阴影中。光仅从其中的一些到达您。在实时渲染中，如果所有光源始终处于活动状态，则包含多个光源的大型场景可能会被计算淹没。如果视锥内部有一定空间但眼睛看不到，则不需要评估遮挡此接收器空间的对象[625，1137]。 Bittner等人[152]使用眼睛的遮挡剔除（第19.7节）找到所有可见的阴影接收器，然后从光的角度将所有潜在的阴影接收器渲染到模板缓冲蒙版。此蒙版编码从光中看到哪些可见的阴影接收器。为了生成阴影贴图，它们使用遮挡剔除从灯光中渲染对象，并使用遮罩将没有接收器的对象剔除。各种剔除策略也可以用于照明。由于辐照度与距离的平方成正比，因此一种常见的技术是在一定的阈值距离之后剔除光源。例如，第19.5节中的门户网站剔除技术可以发现哪些光线会影响哪些单元格。这是一个活跃的研究领域，因为其性能优势可观[1330，1604]。

7.5 PCF 2021年4月2日18点40分

阴影贴图技术的简单扩展可以提供伪软阴影。此方法还可以帮助解决分辨率问题，当单个光样本单元覆盖许多屏幕像素时，这些问题会导致阴影看起来很块状。该解决方案类似于纹理放大（第6.2.1节）。取而代之的是获取四个最接近的样本，而不是从阴影贴图中获取一个样本。该技术不会在深度本身之间进行插值，而是将其与表面深度进行比较的结果。也就是说，将表面的深度分别与四个texel深度进行比较，然后针对每个阴影贴图样本确定该点在光还是阴影下。然后对这些结果（即阴影的0和光的1）进行双线性插值，以计算实际上对表面位置有多少贡献的光。这种过滤会产生人为的柔和阴影。这些半影的变化取决于阴影图的分辨率，相机位置和其他因素。例如，较高的分辨率会使边缘的柔化范围更窄。但是，半影和平滑总比没有好。从阴影贴图中检索多个样本并混合结果的这种想法称为百分比更接近过滤（PCF）[1475]。区域光产生柔和的阴影。到达表面上某个位置的光量取决于从该位置可见的光区域的比例。 PCF试图通过逆转过程来近似准时（或定向）光的柔和阴影。它不是从某个表面位置找到该光的可见区域，而是从原始位置附近的一组表面位置中找到了点光源的可见性。见图7.22。名称“更紧密过滤”是指最终目标，即查找可见光所取样品的百分比。该百分比是然后用于遮蔽表面的光量。在PCF中，会在大约相同的深度但在阴影贴图上的不同纹理位置上在曲面位置附近生成位置。检查每个位置的可见性，然后将这些生成的布尔值（点亮或熄灭）混合在一起，以获得柔和的阴影。请注意，此过程是非物理的：此过程不是直接对光源进行采样，而是依赖于对表面本身进行采样的想法。到遮挡物的距离不会影响结果，因此阴影具有类似大小的半影。尽管如此，该方法在许多情况下仍可提供合理的近似值。

一旦确定了要采样区域的宽度，以避免混叠伪影的方式进行采样就很重要。如何对附近的阴影贴图位置进行采样和过滤有多种变体。变量包括要采样的区域的范围，要使用的样本数，采样模式以及如何加权结果。使用功能较弱的API，可以通过类似于双线性插值的特殊纹理采样模式加速采样过程，该模式访问四个相邻位置。而不是混合结果，而是将四个样本中的每个样本与给定值进行比较，然后返回通过测试的比率[175]。然而，以规则的网格图案执行最近的邻居采样会产生明显的伪像。使用使结果模糊但尊重对象边缘的联合双边滤波器可以提高质量，同时避免阴影泄漏到其他表面上[1343]。有关此过滤技术的更多信息，请参见第12.1.1节。 DirectX 10引入了对PCF的单指令双线性过滤支持，从而提供了更为平滑的结果[53、412、1709、1790]。与最近的邻居采样相比，这提供了可观的视觉改进，但是来自常规采样的伪像仍然是一个问题。最小化网格图案的一种解决方案是使用预先计算的泊松分布图案对区域进行采样，如图7.23所示。这种分布会将样本散布开，以使它们既不靠近也不规则。众所周知，对于每个像素使用相同的采样位置，无论其分布如何，都可能会产生图案[288]。可以通过围绕样本中心随机旋转样本分布来避免此类伪影，从而将混叠化为噪声。 Castano [235]发现，泊松采样产生的噪声因其平滑，风格化的内容而特别引人注目。他提出了一种基于双线性采样的高效高斯加权采样方案。

自遮蔽问题和漏光（例如痤疮和Peter Panning）在PCF中可能会变得更糟。斜率比例偏差仅基于样本与阴影的角度，仅根据其与光线的角度将表面推离光线。通过从表面上的单个位置在更大的区域中采样，某些测试样本可能会被真实的表面阻塞。已经发明了一些不同的附加偏见因素，并成功地使用了这些因素来减少自我遮蔽的风险。 Burley [212]描述了偏心锥，其中每个样本都朝着与光到原始样本的距离成正比的方向移动。 Burley建议坡度为2.0，同时具有较小的恒定偏差。见图7.24。 Schuler [1585]，Isidoro [804]和Tuft [1790]提出了基于观察到的技术，即接收器本身的斜率应用于调整其余样本的深度。在这三种方法中，塔夫特的公式[1790]最容易应用于级联阴影贴图。 Dou等人[373]进一步完善和扩展了这一概念，考虑了z深度如何以非线性方式变化。这些方法假定附近的样本位置在由三角形形成的同一平面上。称为接收器平面深度偏差或其他类似术语，由于在此假想平面上的位置确实在表面上，或者如果模型是凸面，则在该表面的前面，此技术在很多情况下都可以非常精确。如图7.24所示，凹面附近的样本可能会被隐藏。常量，斜率比例尺，接收器平面，视线偏移和法线偏移的组合已用于解决自阴影问题，尽管仍然需要针对每个环境进行手动调整[235，1391，1403]。 PCF的一个问题是，由于采样区域的宽度保持不变，因此阴影将显示出均匀柔和的阴影，且阴影的宽度相同。在某些情况下这是可以接受的，但在封堵器和接收器之间有地面接触的地方看来是不正确的。见图7.25。

7.6 PCSS

在2005年，Fernando [212，467，1252]发表了一种有影响力的方法，称为“百分比封闭器软阴影（PCSS）”。 它通过在阴影图上搜索附近区域以找到所有可能的遮挡物来尝试解决。 这些封堵器到该位置的平均距离用于确定样本区域的宽度：

其中，dr是接收器与光线的距离，并且是平均遮挡物距离。 换句话说，样品的表面积的宽度随着平均遮挡物离接收器的距离越来越远而更接近光而增加。 检查图7.22，并考虑移动封堵器的效果，以了解如何发生。 图7.2（第224页），7.25和7.26显示了示例。

如果没有找到遮挡物，则该位置已完全点亮，无需进一步处理。类似地，如果该位置被完全遮挡，则处理可以结束。否则，将对感兴趣的区域进行采样，并计算出光线的近似贡献。为了节省处理成本，可以使用样品区域的宽度来改变采集的样品数量。可以实现其他技术，例如，对于不太可能重要的遥远的软阴影使用较低的采样率。这种方法的一个缺点是需要对阴影图的适当大小的区域进行采样才能找到遮挡物。使用旋转的泊松圆盘图案可以帮助隐藏欠采样伪像[865，1590]。 Jimenez [832]注意到泊松采样在运动中可能会不稳定，并发现通过使用抖动和随机之间的中间函数形成的螺旋模式在帧与帧之间提供了更好的结果。 Sikachev等。 [1641]详细讨论了使用SM 5.0中功能的PCSS的更快实现，该功能由AMD引入，并经常以其名称来称呼，即接触硬化阴影（CHS）。这个新版本还解决了基本PCSS的另一个问题：半影的大小受阴影贴图分辨率的影响。见图7.25。通过首先生成阴影贴图的mipmap，然后选择最接近用户定义的世界空间内核大小的mip级别，可以最小化此问题。采样一个8×8的区域以查找平均阻止程序深度，仅需要16个GatherRed（）纹理调用。一旦找到半影估计，则将较高分辨率的Mip级别用于阴影的锐利区域，而将较低分辨率的Mip级别用于较柔和的区域。

CHS已用于许多视频游戏中[1351、1590、1641、1678、1679]，并且研究仍在继续。例如，Buades等人[206]提出了可分离的软阴影映射（SSSM），其中对网格进行采样的PCSS过程被分为可分离的部分，并且各个像素之间尽可能地重用了元素。事实证明，对于加速每个像素需要多个样本的算法有帮助的概念是分层的最小/最大阴影图。虽然通常无法对阴影贴图深度进行平均，但每个​​mipmap级别的最小值和最大值可能会很有用。也就是说，可以形成两个mipmap，一个保存在每个区域中找到的最大zdepth（有时称为HiZ），另一个保存最小的zdepth。给定要采样的纹理像素位置，深度和面积，可以使用mipmap快速确定完全照明和完全阴影的条件。例如，如果纹理像素的z深度大于为mipmap的相应区域存储的最大z深度，则该纹理像素必须处于阴影中-无需进一步的采样。这种类型的阴影图使确定光可见性的任务更加有效[357、415、610、680、1064、1811]。 PCF之类的方法通过对附近的接收器位置进行采样来工作。 PCSS通过查找附近阻塞器的平均深度来工作。这些算法不直接考虑光源的面积，而是对附近的表面进行采样，并且受阴影图分辨率的影响。 PCSS背后的一个主要假设是，平均阻断剂是对半影大小的合理估计。当两个遮挡物（例如路灯和远处的山脉）部分遮挡同一像素的同一表面时，此假设将被破坏，并可能导致伪像。理想情况下，我们要确定从单个接收器位置可见多少面光源。几位研究人员已经探索了使用GPU进行反投影的方法。想法是将每个接收器的位置视为一个视点，并将区域光源视为一个视平面，并将遮挡物投射到该平面上。 Schwarz和Stamminger [1593]以及Guennebaud等[617]都总结了以前的工作并提出了自己的改进。 Bavoil等人[116]采用了另一种方法，即使用深度剥离来创建多层阴影贴图。反投影算法可以提供出色的结果，但是（到目前为止）每像素高昂的成本意味着它们尚未在交互式应用程序中得到采用。

7.7 过滤阴影映射 2021年4月5日10点48分

一种可以过滤生成的阴影图的算法是Donnelly和Lauritzen的方差阴影图（VSM）[368]。 该算法在一张地图中存储深度，在另一张地图中存储深度平方。 生成图时可以使用MSAA或其他抗锯齿方案。 可以对这些地图进行模糊处理，映射，放置在总面积表中[988]或任何其他方法。 将这些贴图视为可过滤纹理的能力是一个巨大的优势，因为从它们中检索数据时，可以使用整个采样和过滤技术阵列。

我们将在这里更深入地描述VSM，以了解该过程的工作方式。 同样，此类算法对所有方法都使用相同类型的测试。 有兴趣学习更多有关该领域的读者，请访问相关参考书，我们还建议使用Eisemann等人的书[412]，该书为该主题提供了更大的篇幅。 首先，对于VSM，在接收器的位置对深度图进行采样（仅一次），以返回最近的光遮挡器的平均深度。 当该平均深度M1（称为第一矩）大于阴影接收器t上的深度时，该接收器被视为完全处于光线下。 当平均深度小于接收者的深度时，将使用以下公式：

其中pmax是光中样本的最大百分比，σ2是方差，t是接收器深度，M1是阴影图中的平均预期深度。 深度平方阴影图的样本M2（称为第二矩）用于计算方差：

值pmax是接收机可见度百分比的上限。实际照度百分比p不能大于此值。这个上限来自切比雪夫不等式的单面变体。该方程试图使用概率论来估计遮挡物在表面位置的分布超出表面到光的距离的程度。 Donnelly和Lauritzen表明，对于固定深度处的平面遮挡物和平面接收器，p = pmax，因此公式7.7可作为许多真实阴影情况的良好近似。 Myers [1251]建立了关于这种方法为何起作用的直觉。在阴影边缘，面积上的差异会增加。深度差异越大，差异越大。因此，（t-M1）2项是可见性百分比的重要决定因素。如果该值仅略高于零，则意味着平均遮挡器深度比接收器更接近光线，然后pmax接近1（完全点亮）。这将沿着半影的完全照亮的边缘发生。进入半影，平均遮挡物深度接近光，因此该项变大并且pmax下降。同时，方差本身在半影内变化，从沿边缘的几乎为零到最大的方差，在此方格的深度不同且面积相等。这些术语保持平衡以在整个半影上给出线性变化的阴影。与其他算法的比较请参见图7.26。

方差阴影贴图的一个重要特征是，它可以以优雅的方式处理由于几何形状而引起的表面偏差问题。 Lauritzen [988]推导了如何使用表面的坡度修改第二弯矩的值。偏差和数值稳定性带来的其他问题可能是方差映射的问题。例如，公式7.8从另一个相似的值中减去一个大的值。这种类型的计算往往会放大底层数字表示形式的准确性不足。使用浮点纹理有助于避免此问题。由于GPU的优化纹理功能得到有效利用，因此整体VSM可以显着提高处理时间，从而提高质量。虽然PCF需要更多的样本，因此需要更多的时间来避免在生成较柔和的阴影时产生噪声，但VSM只能使用一个高质量的样本来确定整个区域的效果并产生平滑的半影。这种能力意味着在算法的限制内，可以在不增加成本的情况下任意使阴影变得柔和。

与PCF一样，过滤内核的宽度决定了半影的宽度。通过找到接收器和最近的遮挡物之间的距离，可以改变内核宽度，从而产生令人信服的柔和阴影。 Mipmapped样本对于半暗带的宽度缓慢增加的覆盖率估计不佳，从而产生了盒状伪像。 Lauritzen [988]详细介绍了如何使用求和面积表来给出更好的阴影。图7.27给出了一个示例。当两个或多个封堵器覆盖接收器且一个封堵器靠近接收器时，沿半影区域的一个位置方差阴影映射将分解。来自概率理论的切比雪夫不等式将产生与正确的光照百分比无关的最大光照值。最近的遮挡物（仅部分遮挡了光线）使方程的近似值偏离了目标。这会导致漏光（又称漏光），而被完全遮挡的区域仍会接收光。见图7.28。通过在较小的区域上获取更多的样本，可以解决此问题，将方差阴影映射转换为PCF形式。与PCF一样，速度和性能也要取舍，但对于阴影深度复杂度较低的场景，方差映射效果很好。 Lauritzen [988]提出了一种由艺术家控制的方法来改善该问题，该方法是将低百分比视为完全阴影，并将其余百分比范围重新映射为0％至100％。这种方法使流血变暗，但总体上使半影变窄。尽管轻度出血是一个严重的局限性，但VSM可以很好地从地形中产生阴影，因为此类阴影很少涉及多个遮挡物[1227]。

能够使用过滤技术快速产生平滑阴影的希望在过滤阴影映射中引起了极大的兴趣。主要挑战是解决各种出血问题。 Annen等人[55]介绍了卷积阴影图。扩展了Soler和Sillion的平面接收器算法[1673]背后的思想，该思想是通过傅立叶展开对阴影深度进行编码。与方差阴影映射一样，可以对此类映射进行过滤。该方法收敛于正确答案，因此减轻了漏光的问题。卷积阴影映射的一个缺点是需要计算和访问多个项，从而大大增加了执行和存储成本[56，117]。 Salvi [1529，1530]和Annen等[56]同时并独立地想到了基于指数函数使用单个项的想法。称为指数阴影图（ESM）或指数方差阴影图（EVSM），此方法将深度的指数及其第二矩保存到两个缓冲区中。指数函数更接近于阴影贴图执行的阶跃函数（即是否在光线下），因此可以显着减少出血伪影。它避免了卷影阴影贴图引起的另一个问题，即振铃，在那儿，在超过原始遮挡器深度的特定深度处可能会发生轻微的光泄漏。

存储指数值的一个限制是，第二矩值可能会变得非常大，因此使用浮点数会超出范围。为了提高精度并允许指数函数更加陡峭地下降，可以生成z深度以使其线性[117，258]。由于与卷积图相比，VSM具有更高的质量，更低的存储量和更好的性能，因此，指数阴影图方法引发了三种过滤方法中的最大兴趣。 Pettineo [1405]指出了其他一些改进，例如使用MSAA改善结果并获得有限的透明度的能力，并描述了如何使用计算着色器改善过滤性能。最近，Peters和Klein [1398]引入了矩阴影映射。它提供了更好的质量，但是以使用四个或更多时间为代价，从而增加了存储成本。可以通过使用16位整数存储力矩来降低此成本。 Pettineo [1404]实现了该新方法并将其与ESM进行了比较，从而提供了探索多种变体的代码库。

级联阴影贴图技术可以应用于滤波后的贴图，以提高精度[989]。 与标准级联图相比，级联ESM的一个优势是可以为所有级联设置单个偏差因子[1405]。 Chen和Tatarchuk [258]详细介绍了级联ESM遇到的各种漏光问题和其他伪像，并提出了一些解决方案。 可以将过滤后的地图视为PCF的一种廉价形式，只需少量样本即可。 像PCF一样，此类阴影具有恒定的宽度。 这些过滤的方法都可以与PCSS结合使用，以提供可变宽度的半影[57，1620，1943]。 矩阴影映射的扩展还包括提供光散射和透明效果的功能[1399]。

7.8 体积阴影技术 2021年4月5日11点00分

透明物体会衰减并改变光的颜色。对于某些透明对象集，可以使用类似于5.5节中讨论的技术来模拟这种效果。例如，在某些情况下，可以生成第二类型的阴影图。透明对象将呈现给它，并存储最接近的深度和颜色或Alpha覆盖范围。如果接收器没有被不透明的阴影图遮挡，则将测试透明深度图，如果遮挡了，则根据需要检索颜色或覆盖范围[471，1678，1679]。这个想法使人想起了第7.2节中的阴影和光投影，所存储的深度避免了投影到透明物体和光之间的接收器上。此类技术无法应用于透明对象本身。自阴影对于真实渲染对象（例如头发或云朵）非常重要，在这些对象中，对象很小或半透明。单深度阴影贴图不适用于这些情况。 Lokovic和Veach [1066]首先提出了深阴影贴图的概念，其中每个阴影贴图的纹理像素都存储了光如何随深度下降的功能。通常通过一系列不同深度的样本来近似此功能，每个样本都具有不透明度值。地图中包含给定位置深度的两个样本用于查找阴影的效果。 GPU面临的挑战是有效地生成和评估此类功能。这些算法使用类似的方法，并且遇到一些与顺序无关的透明性算法（第5.5节）所遇到的类似挑战，例如，如实存储表示每个功能所需的数据的紧凑存储。

Kim和Neumann [894]率先提出了基于GPU的方法，他们将其称为不透明阴影贴图。仅存储不透明度的地图是在一组固定的深度下生成的。 Nguyen和Donnelly [1274]提供了此方法的更新版本，生成的图像如图719页上的图17.2所示。但是，深度切片都是平行且均匀的，因此，由于以下原因，需要很多切片来隐藏切片之间的不透明度伪像线性插值。 Yuksel和Keyser [1953]通过创建更紧密地遵循模型形状的不透明度贴图来提高效率和质量。这样做可以减少所需的层数，因为对最终图像的每一层评估都更为重要。为了避免必须依赖固定的切片设置，已经提出了更多的自适应技术。 Salvi等人[1531]引入了自适应体积阴影贴图，其中每个阴影贴图纹理像素都存储了不透明度和层深度。像素着色器操作用于在栅格化数据流（表面不透明度）时有损压缩​​。这避免了需要无限制的内存量来收集所有样本并在一组中进行处理。该技术类似于深阴影贴图[1066]，但压缩步骤是在像素着色器中动态完成的。将函数表示形式限制为固定的少量不透明度/深度对，可使GPU上的压缩和检索效率更高[1531]。由于需要读取，更新和写回曲线，因此其成本比简单的混合要高，并且取决于用于表示曲线的点数。在这种情况下，该技术还需要支持UAV和ROV功能的最新硬件（第3.8节的内容）。有关示例，请参见图7.29。

自适应体积阴影映射方法用于GRID2游戏中的逼真的烟雾渲染，平均成本低于2毫秒/帧[886]。 Furst等人[509]描述并提供了用于实现视频游戏深阴影图的代码。 他们使用链接列表存储深度和Alpha，并使用指数阴影映射在光照和阴影区域之间提供软过渡。

阴影算法的探索仍在继续，各种算法和技术的综合变得越来越普遍。 例如，Selgrad等人[1603]研究了存储带有链接列表的多个透明样本，并使用具有分散写入的计算着色器来构建地图。 他们的工作使用深阴影图概念以及过滤后的地图和其他元素，它们为提供高质量的软阴影提供了更通用的解决方案。

7.9 不规则Z-Buffer阴影

由于多种原因，各种阴影贴图方法很受欢迎.它们的成本是可预测的，并且可以很好地适应不断增加的场景大小，最坏的情况是与图元数量成线性关系.它们依靠光栅化来定期采样灯光的世界视野，从而将它们很好地映射到GPU上.但是，由于这种离散采样，出现了问题，因为眼睛看到的位置与光线看到的位置不是一一对应的.当光对表面的采样频率低于眼睛时，会出现各种混叠问题.即使采样率是可比较的，也存在偏差问题，因为在与眼睛所见位置稍有不同的位置采样了表面。

阴影体积提供了一种精确的分析解决方案，因为光线与表面的相互作用会产生一组三角形，这些三角形定义了给定位置是被照明还是处于阴影中。 当在GPU上实现时，算法的不可预测的成本是一个严重的缺点。近年来[1648]所探索的改进令人着迷，但尚未获得在商业应用中采用的“存在证据”。

从长远来看,另一种分析阴影测试方法可能具有潜力：光线跟踪。在11.2.2节中有详细描述，基本思想很简单，尤其是对于阴影。 光线从接收器位置射向光。 如果发现任何阻挡光线的物体，则接收器处于阴影中。 快速射线追踪器的大部分代码专用于生成和使用分层数据结构，以最大程度地减少每条射线所需的对象测试数量。 在动态场景的每个框架上构建和更新这些结构是数十年的历史，也是一个持续的研究领域.

另一种方法是使用GPU的光栅化硬件来查看场景，但不只是z深度，还存储了有关每个光网格单元中遮挡物边缘的附加信息[1003，1607]。例如，想象一下在每个阴影贴图纹理元素上存储与网格单元重叠的三角形列表。这样的列表可以通过保守光栅化来生成，如果三角形的任何部分与像素重叠，而不仅仅是像素的中心，则三角形会生成一个片段（第23.1.2节）。这种方案的一个问题是通常需要限制每个纹理像素的数据量，这反过来可能导致确定每个接收器位置的状态不准确。考虑到现代GPU的链表原理[1943]，每个像素存储更多数据当然是可能的。但是，除了物理内存限制外，每个纹理元素在列表中存储可变数量的数据的问题在于，GPU处理可能变得非常低效，因为单个扭曲可能具有一些片段线程，需要检索和处理许多项目，而其余的线程则处于空闲状态，无事可做。构建着色器以避免动态“if”语句和循环导致的线程发散对性能至关重要.

在阴影图中存储三角形或其他数据并针对它们测试接收器位置的另一种方法是翻转问题，存储接收器位置，然后针对每个三角形测试三角形。 这种保存接收器位置的概念最早由Johnson等人[839]和Aila和Laine [14]提出，被称为不规则z缓冲区（IZB）。 名称有点误导，因为缓冲区本身具有法线的阴影贴图，规则形状。相反，缓冲区的内容是不规则的，因为每个阴影贴图纹理像素将在其中存储一个或多个接收器位置，或者可能根本没有。参见图7.30。

使用Sintorn等人[1645]和Wyman等人[1930，1932]提出的方法，多遍算法创建了IZB并测试了其内容在光线下的可见性。 首先，从眼睛渲染场景，以找到从眼睛看到的表面的z深度。 这些点将转换为灯光的场景视图，并由此设置灯光视锥的紧密边界。 然后将这些点放置在灯光的IZB中，每个点都放在其对应纹理像素的列表中。 请注意，有些列表可能是空的，是灯光可以看到的空间，但是眼睛看不到任何表面。 遮罩保守地光栅化到灯光的IZB，以确定是否隐藏了任何点，等等。 保守栅格化可确保即使三角形不覆盖光纹素的中心，也将针对可能重叠的点进行测试。

可见性测试在像素着色器中进行。 测试本身可以可视化为射线追踪的一种形式。 从图像点的位置到光源会产生光线。如果一个点在三角形内并且比三角形平面更远，则该点被隐藏。光栅化所有遮挡物后，将可见性结果用于对表面进行着色。此测试也称为视锥跟踪，因为可以将三角形视为定义了视锥的视图，该视锥用于检查点是否包含在视锥中。

仔细的编码对于使此方法与GPU兼容至关重要。 Wyman等人[1930，1932]指出，它们的最终版本比初始原型快两个数量级。 这种性能提升的一部分是直接的算法改进，例如剔除表面法线背对光（因此始终不照明）的图像点，并避免为空纹理像素生成碎片。 其他性能提升来自改善GPU的数据结构，以及通过在每个texel中使用较短的相似长度的点列表来最小化线程差异。 图7.30显示了一个带有长列表的低分辨率阴影图，用于说明目的。 理想情况是每个列表一个图像点。 分辨率越高，列表越短，但封堵器生成的片段数量也越多。

从图7.30的左下方图像可以看出，由于透视效应，接地面上可见点的密度在左侧比右侧明显更高。使用级联阴影贴图可通过将更多的光照贴图分辨率聚焦在更靠近眼睛的位置来帮助减小这些区域中的列表大小。

这种方法避免了其他方法的采样和偏差问题，并提供了完全清晰的阴影。出于审美和感官上的原因，通常需要柔和的阴影，但附近的遮挡物（例如Peter Panning）可能会有偏斜问题。 Story and Wyman [1711，1712]探索混合阴影技术。核心思想是使用遮挡物距离混合IZB和PCSS阴影，当遮挡物靠近时使用硬阴影结果，而在更远的地方使用软阴影结果。参见图7.31。阴影质量通常对于附近的物体最为重要，因此，仅对选定的子集使用此技术，就可以降低IZB成本。 此解决方案已成功用于视频游戏中。本章从这样的图像开始，如第224页的图7.2所示。

7.10 其它应用

将阴影贴图视为定义一定的空间，将光与暗分离，也可以帮助确定对象的哪些部分需要阴影。 Gollent [555]介绍了CD Projekt的地形阴影系统如何为每个区域计算仍被遮挡的最大高度，然后可以将该最大高度用于阴影，不仅可以遮挡地形，还可以遮挡场景中的树木和其他元素。为了找到每个高度，为太阳绘制可见区域的阴影图。然后检查每个地形的高度场位置是否有阳光可见性。如果处于阴影中，则通过将世界高度增加一个固定的步长，直到看到太阳，然后执行二进制搜索，来估计太阳首次可见的高度。换句话说，我们沿着一条垂直线行进，并进行迭代以缩小与阴影图的表面相交的位置，从而将明暗分开。内插相邻的高度以在任何位置找到此遮挡高度。在图7.32中可以看到用于地形高度场的软阴影的此技术的示例。在第14章中，我们将看到更多使用光线行进穿过明暗区域的方法。

值得一提的最后一种方法是渲染屏幕空间阴影。由于分辨率有限，阴影贴图通常无法在小要素上产生准确的遮挡。当渲染人脸时，这尤其成问题，因为我们特别容易注意到它们上的任何视觉伪像。例如，渲染发光的鼻孔（当不需要时）看起来很刺耳。虽然使用分辨率更高的阴影图或仅针对目标区域的单独阴影图可以提供帮助，但另一种可能性是利用已经存在的数据。在大多数现代渲染引擎中，从摄像机角度来看的深度缓冲区（来自较早的预渲染）在渲染期间可用。存储在其中的数据可以视为高度场。通过迭代采样该深度缓冲区，我们可以执行光线行进过程（第6.8.1节），并检查朝向光的方向是否被遮挡。尽管代价高昂，因为它涉及到重复采样深度缓冲区，但这样做可以为剪切场景中的特写镜头提供高质量的结果，在剪切场景中，通常需要花费额外的毫秒时间。该方法是由Sousa等人[1678]提出的，如今已广泛用于许多游戏引擎[384，1802].

总结整个章节，到目前为止，某种形式的阴影映射是用于将阴影投射到任意表面形状上的最常用算法。 当在大面积（例如室外场景）中投射阴影时，级联阴影贴图可提高采样质量。 通过SDSM为近平面找到合适的最大距离可以进一步提高精度。 百分比更接近过滤（PCF）使阴影更柔和，百分比更接近软阴影（PCSS）及其变体使接触硬化，不规则的z缓冲区可提供精确的硬阴影。 过滤后的阴影图可提供快速的软阴影计算，并且在遮挡物远离接收器（如地形）时，效果特别好。 最后，尽管成本高昂，但屏幕空间技术仍可用于提高精度。

在本章中，我们重点介绍了应用程序中当前使用的关键概念和技术。每个都有自己的优势，选择取决于世界的大小，构图（静态内容还是动画），材质类型（不透明，透明，头发或烟雾）以及灯光的数量和类型（静态或动态；局部或远距离；点） ，斑点或面积），以及诸如基础纹理可以隐藏任何伪像的程度之类的因素。 GPU的功能不断发展和完善，因此我们希望在未来的几年中会继续出现与硬件良好匹配的新算法。例如，第19.10.1节中描述的稀疏纹理技术已应用于阴影贴图存储，以提高分辨率[241、625、1253]。 Sintorn，Kampe等人[850，1647]以一种创新的方法探索了将光的二维阴影贴图转换为三维三维像素集的想法（小盒子；请参见第13.10节）。使用体素的优点是可以将其分类为亮或暗，因此需要的存储量最少。高度压缩的稀疏体素八叉树表示法可存储大量灯光和静态遮挡物的阴影。 Scandolo等人[1546]将其压缩技术与使用双阴影图的基于间隔的方案相结合,从而提供了更高的压缩率.Kasyan [865]使用体素圆锥追踪（第13.10节）从区域光生成柔和阴影.有关示例，请参见图7.33.更多的圆锥形阴影显示在第585页的图13.33中。