概述

为了加快渲染速度，动态场景中的有效遮挡剔除是游戏和实时图形社区非常重要的主题。 我们提出了一种新颖的算法，该算法的灵感来自图形硬件深度剔除的最新进展，但针对具有SIMD功能的CPU进行了调整和优化。我们的算法具有非常低的内存开销，比以前的工作快3倍，而使用全分辨率深度缓冲方法可以剔除98％的三角形。它支持交错遮挡物栅格化和遮挡查询，而不会带来损失，从而使其易于在场景图形遍历或渲染代码中使用。

1 介绍

现代游戏和实时图形应用程序中的环境正逐渐变得更加动态，并且允许用户有更大的自由与虚拟世界进行交互。尽管这增加了沉浸感，并使虚拟环境更像真实世界，但它对数据结构和剔除算法的要求也比以往更高。

结果，越来越多的游戏引擎较少关注传统的预先计算的可见性确定算法，例如潜在的可见集合[ARB90，KCCO01，NBG02]，而倾向于将重要遮挡物栅格化为层次深度缓冲区的算法[GKM93]。最近的一些例子是Frostbite引擎[And09，Col11]，Umbra遮挡剔除引擎（http://umbra3d.com/）和英特尔的软件遮挡剔除框架[CMK \* 16]，它们都依赖于软件光栅化来创建 层次深度缓冲区。然后，此缓冲区用于在将绘制命令发送到GPU之前在CPU上执行遮挡查询。Haar和Aaltonen [HA15]提出了一个具有相同组件的相似系统，但完全在GPU上运行。

尽管需要精确的遮挡查询，但是大多数系统都被迫在性能和准确性之间取得平衡。例如，Andersson [And09]和Collin [Col11]的工作使用了非常低分辨率的深度缓冲区，并通过使遮蔽网格成为内部保守来进行补偿。但是，创建内部保守的网格是一项艰巨的任务。网格应缩小一个像素的面积，这又取决于投影，并且可能是无边界的。这可能导致假阴性或错误的剔除，并对建模遮挡网格的美术师提出了很高的要求。相比之下，英特尔的遮挡剔除演示使用了全分辨率深度缓冲区，因此不会遭受相同的缺点。但是，性能可能会降低。

与以前的方法相比，我们的主要贡献是可以有效分离深度和覆盖范围数据的分层深度表示。这是性能的关键，因为我们可以非常快速地计算出图块的准确覆盖率，从本质上讲，使图块成为我们最小的处理单元，而不是像以前的工作中那样使像素成为像素。这使我们能够保留使用高分辨率深度缓冲区的好处，同时达到与低分辨率[And09，Col11]甚至GPU加速方法[HA15]类似的性能。

我们的工作受到Andersson等人[AHAM15]有关图形硬件的蒙版深度剔除工作的启发。我们建议对其算法进行以下四个扩展，以使其更适合于软件实现。

仅使用一些SIMD指令即可并行生成32×8像素图块的覆盖掩码的算法。

* 一种新颖的深度更新启发式方法，类似于Andersson等人的方法。 [AHAM15]，但为了性能而牺牲准确性。
* SIMD友好的分层深度缓冲区表示形式，具有低内存开销，专为遮挡剔除而设计。
* 低成本遮挡物渲染和遮挡查询交织，可实现简单高效的场景图遍历算法。

此外，我们提供了算法的优化实现。在图1中，我们显示了一个算法示例，该算法应用于具有复杂遮挡的场景。