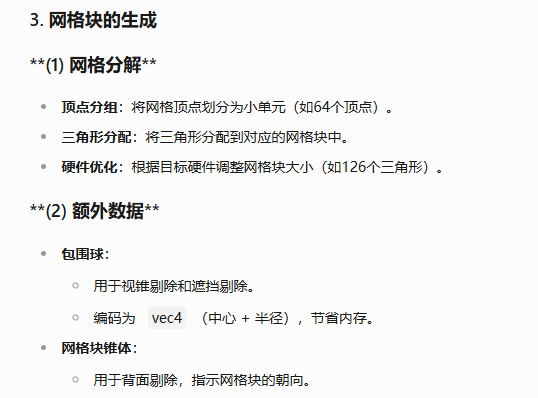
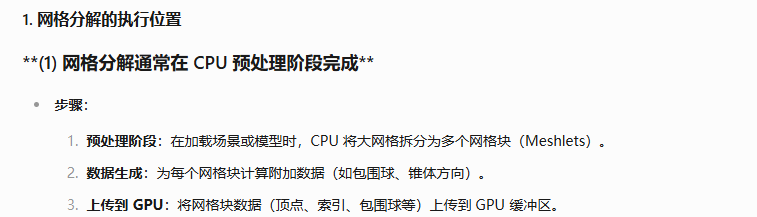


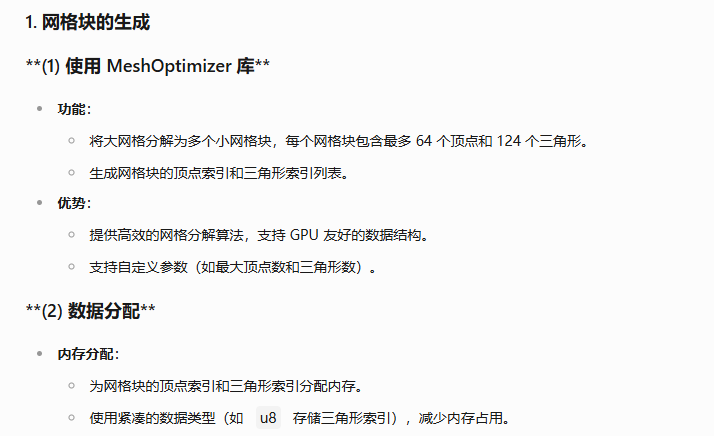
**注意**：  
当我们提到图形管线的几何阶段时，我们并不是指几何着色器。管线的几何阶段包括输入装配（IA）、顶点处理（Vertex Processing）和图元装配（Primitive Assembly）。顶点处理可以运行以下一种或多种着色器：顶点着色器、几何着色器、曲面细分着色器、任务着色器和网格着色器。

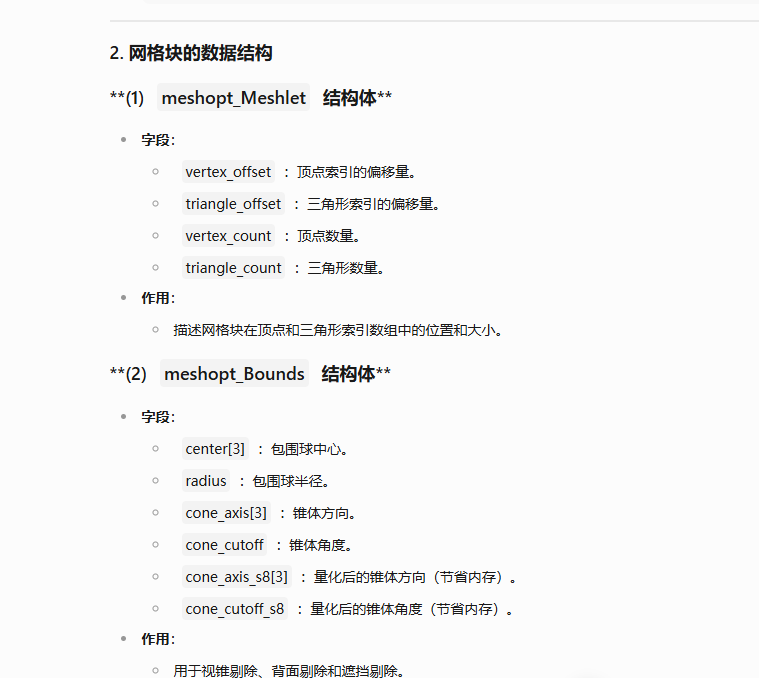




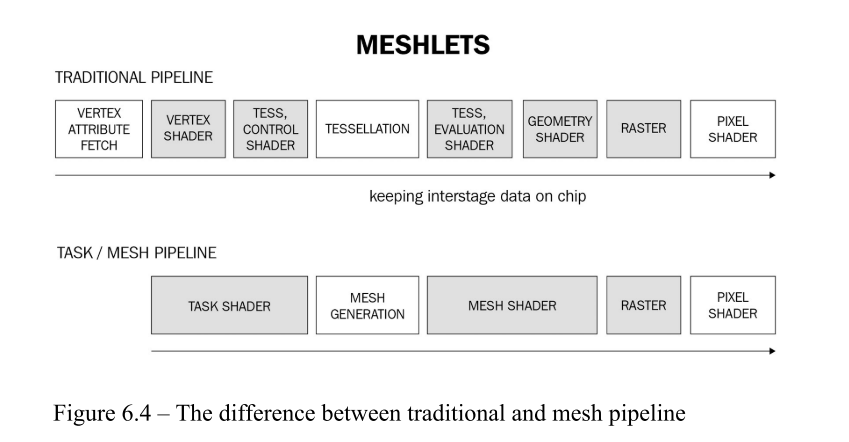












在开始之前，我们应该提到，​**网格着色器（Mesh Shaders）​** 可以单独使用，而无需 ​**任务着色器（Task Shaders）​**。例如，如果你想在 CPU 上对网格块执行剔除或其他预处理步骤，完全可以这样做。  
另外，请注意，任务和网格着色器在图形管线中取代了 ​**顶点着色器（Vertex Shaders）​**。网格着色器的输出将直接由 ​**片段着色器（Fragment Shader）​** 消费。

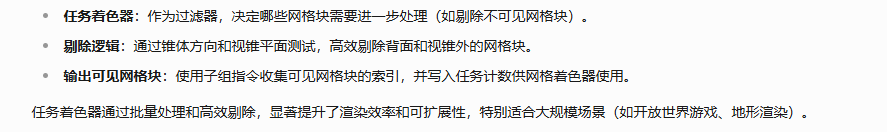
**任务和网格着色器的引入\*\***

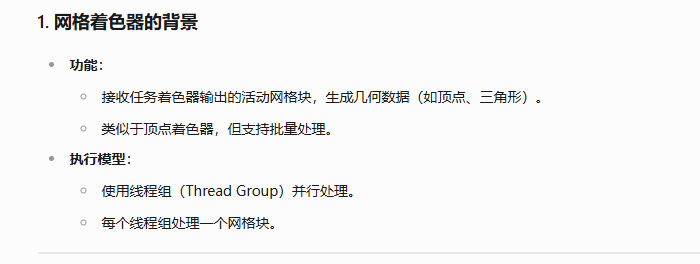
* ​**网格着色器**：取代顶点着色器，直接生成几何数据（如顶点、三角形）。
* ​**任务着色器**：作为过滤器，决定哪些网格块需要进一步处理（如剔除不可见网格块）。













**使用计算着色器进行 GPU 剔除**  
在上一节中，我们演示了如何在网格块上执行背面剔除和视锥剔除。在本节中，我们将使用计算着色器实现视锥剔除和遮挡剔除。  
根据渲染管线的不同，遮挡剔除通常通过深度预渲染（Depth Pre-Pass）完成，在该阶段我们只写入深度缓冲区。然后，深度缓冲区可以在 G-Buffer 阶段使用，以避免着色已知被遮挡的片元。  
这种方法的缺点是我们必须绘制场景两次，并且除非有其他工作可以与深度预渲染重叠，否则必须等待深度预渲染完成后才能进行下一步。

**深度金字塔生成**  
在描述遮挡算法时，我们提到了深度缓冲区的使用。然而，我们并没有直接使用深度缓冲区，而是使用了称为 ​**深度金字塔** 的结构。你可以将其视为深度缓冲区的 Mipmap。  
与传统 Mipmap 的主要区别在于，我们不能使用双线性插值来计算较低级别。如果我们使用常规插值，将计算出场景中不存在的深度值。

