3.7 几何着色器

几何着色器可以将图元转换为其他图元，这是曲面细分阶段无法做到的。例如，可以通过让每个三角形创建线边将三角形网格转换为线框视图。或者，线条可以被面向观察者的四边形替换，因此制作具有较厚边缘的线框渲染[1492]。随着DirectX 10的发布，几何着色器在2006年末随着DirectX 10的发布被添加到硬件加速图形管线中。它位于管道中的曲面细分着色器之后，它的使用是可选的。虽然它是Shader Model 4.0的必需部分，但在早期的着色器模型中并未使用。OpenGL 3.2和OpenGL ES 3.2也支持这种类型的着色器。

几何着色器的输入是单个对象及其关联的顶点。对象通常由带状三角形、线段或简单的点组成。几何着色器可以定义和处理扩展图元。特别是，可以传入三角形外的三个附加顶点，可以使用折线上的两个相邻顶点。见图3.12。使用DirectX 11和Shader Model 5.0，您可以传入更精细的面片，最多有32个控制点。也就是说，曲面细分阶段对于面片生成更有效[175]。

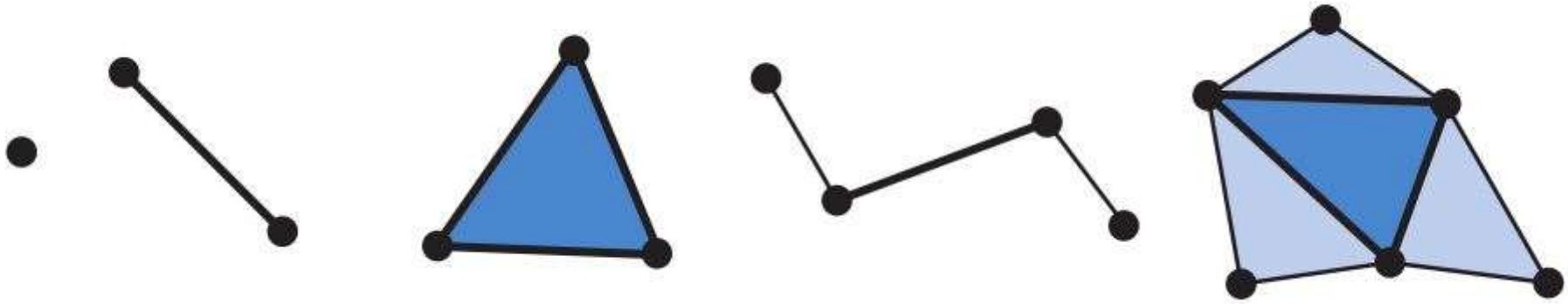

图3.12. 几何着色器程序的几何着色器输入是某种单一类型：点、线段、三角形。最右边的两个图元包括与线和三角形对象相邻的顶点。更复杂的面片类型是可能的。

几何着色器处理该图元并输出零个或多个顶点，这些顶点被视为点、折线或三角形条带。请注意，几何着色器根本无法生成任何输出。通过这种方式，可以通过编辑顶点、添加新图元和删除其他图元来有选择地修改网格。

几何着色器旨在修改传入数据或制作有限数量的副本。例如，一种用途是生成六个转换后的数据副本，以同时渲染立方体贴图的六个面；详见第10.4.3节。 它还可以用于高效地创建级联阴影贴图以生成高质量的阴影。利用几何着色器的其他算法包括从点数据创建可变大小的粒子、沿着轮廓挤

出翅片以进行毛发渲染，以及找出物体边缘的阴影算法。更多示例请参见图3.13。本书的其余部分将讨论这些和其他用途。

图3.13. 几何着色器(GS)的一些用途。在左侧，使用GS即时执行元球等值面细分。在中间，使用GS完成线段的分形细分并输出，并且由GS生成广告牌以显示闪电。在右侧，布料模拟是通过使用带有流输出的顶点和几何着色器来执行的。（来自 NVIDIA SDK 10 [1300]示例的图像，由NVIDIA Corporation提供。）

DirectX 11添加了几何着色器使用实例化的能力，其中几何着色器可以在任何给定的图元上运行一定次数[530, 1971]。在OpenGL 4.0中，这是用调用计数指定的。几何着色器还可以输出最多四个流。一个流可以向下发送到渲染管线以进行进一步处理。所有这些流都可以选择发送到流输出渲染目标。

几何着色器保证以与输入相同的顺序输出图元的结果。这会影响性能，因为如果多个着色器内核并行运行，则必须保存和排序结果。这个和其他因素不利于几何着色器用于在单个调用中复制或创建大量几何体[175, 530]。

在发出绘制调用后，管线中只有三个地方可以在GPU上创建工作：光栅化、细分阶段和几何着色器。其中，考虑到所需的资源和内存，几何着色器的行为是最不可预测的，因为它是完全可编程的。在实践中，几何着色器通常用处不大，因为它不能很好地映射到GPU的优势。在某些移动设备上，它是用软件实现的，因此在移动端不鼓励使用它[69]。

3.7.1 流输出

GPU管线的标准用途是通过顶点着色器发送数据，然后光栅化生成的三角形并在像素着色器中处理它们。过去数据总是通过管线传递，无法访问中间结果。流输出的概念是在Shader Model 4.0中引入的。在顶点着色器（以及可选的曲面细分和几何着色器）处理顶点之后，除了被发送到光栅化阶段之外，它们还可以以流的形式输出，即有序数组。事实上，可以完全关闭光栅化，然后将管线纯粹用作非图形流处理器。以这种方式处理的数据可以通过管线发回，从而允许迭代处理。这种类型的操作对于模拟流动的水或其他粒子效应非常有用，如第13.8节所述。它还可以用于为模型蒙皮，然后让这些顶点可重复使用（第4.4节）。

流输出仅以浮点数的形式返回数据，因此它可能具有显着的内存成本。流输出适用于图元，而不是直接适用于顶点。如果网格沿着管线发送，每个三角形都会生成自己的一组三个输出顶点。原始网格中的任何顶点共享都将丢失。出于这个原因，一个更典型的用途是将顶点作为点集图元通过管线发送。在OpenGL中，流输出阶段称为变换反馈，因为它的大部分使用重点是变换顶点并返回它们以供进一步处理。图元保证按照它们输入的顺序发送到流输出目标，这意味着顶点顺序将被保持[530]。