2.6 管线综述

点、线和三角形是构建模型或对象的渲染图元。假设该应用程序是一个交互式计算机辅助设计 (CAD) 应用程序，并且用户正在检查华夫饼制造商的设计。在这里，我们将在整个图形渲染管线中遵循这个模型，包括四个主要阶段：应用程序、几何、光栅化和像素处理。场景以透视图渲染到屏幕上的窗口中。在这个简单的示例中，华夫饼机模型包括线条（以显示零件的边缘）和三角形（以显示表面）。华夫饼机有一个可以打开的盖子。一些三角形由带有制造商标志的二维图像构成。对于这个例子，表面着色完全在几何阶段计算，除了纹理的应用，它发生在光栅化阶段。

2.6.1 应用程序阶段

CAD应用程序允许用户选择和移动模型的各个部分。例如，用户可能会选择盖子，然后移动鼠标将其打开。应用阶段必须将鼠标移动转换为相应的旋转矩阵，然后确保该矩阵在渲染时正确应用于盖子。另一个示例：播放的动画沿着预定义的路径移动相机以从不同视图显示华夫饼机。然后，应用程序必须根据时间更新相机参数，例如位置和视图方向。对于要渲染的每一帧，应用程序阶段将模型的相机位置、光照和图元提供给管道中的下一个主要阶段——几何阶段。

2.6.2 几何处理

对于透视视图，我们假设应用程序提供了一个投影矩阵。此外，对于每个对象，应用程序都计算了一个矩阵，该矩阵描述了视图变换以及对象本身的位置和方向。在我们的例子中，华夫饼机的底座有一个矩阵，盖子是另一个。在几何阶段，对象的顶点和法线使用该矩阵进行变换，将对象放入视图空间。然后可以使用材质和光源属性计算顶点处的着色或其他计算。然后使用单独的用户提供的投影矩阵执行投影，将对象转换为代表眼睛所见的单位立方体空间。立方体外的所有基元都将被丢弃。与这个单位立方体相交的所有图元都被裁剪在立方体上，以获得一组完全位于单位立方体内部的图元。然后将顶点映射到屏幕上的窗口中。在执行完所有这些每三角形和每顶点操作之后，结果数据将传递到光栅化阶段。

2.6.3 光栅化

然后将在前一阶段裁剪后幸存下来的所有图元进行光栅化，这意味着找到图元内的所有像素并将其进一步发送到管线中进行像素处理。

2.6.4 像素处理

这里的目标是计算每个可见图元的每个像素的颜色。那些与任何纹理（图像）相关联的三角形将根据需要使用这些图像进行渲染。可见性通过z缓冲区算法以及可选的丢弃和模板测试来解决。依次处理每个对象，然后将最终画面显示在屏幕上。

2.6.5 总结

这条管线源于数十年针对实时渲染应用程序的API和图形硬件演变。需要注意的是，这并不是唯一可能的渲染管道；离线渲染管道经历了不同的进化路径。电影制作的渲染通常使用微多边形管道[289, 1734]完成，但最近已经普遍开始使用光线追踪和路径追踪了。第11.2.2节中介绍的这些技术也可用于架构和设计视觉化。

多年来，应用程序开发人员使用此处描述的过程的唯一方法是通过使用中的图形API定义的固定功能管线。固定功能管线之所以如此命名，是因为实现它的图形硬件由无法以灵活方式编程的元素组成。主要固定功能管线的机器的最后一个例子是2006年推出的任天堂Wii。另一方面，可编程GPU可以准确地确定在整个管线的各个子阶段应用哪些操作。对于本书的第四版，我们假设所有开发都是使用可编程GPU完成的。