**5 渲染流水线**

1，模型的表示

实体3D对象是用三角形网格来近似表示的，三角形是3D物体建模的基石。

除了三角形，点和线也有其用武之地。

2，计算机色彩

颜色向量也有自己专属的颜色运算，即分量式乘法:对应分量相乘。

128位颜色用XMVECTOR类型来描述。

32位颜色用XMCOLOR类型来描述，每个占用8位字节的颜色分量就可以分别描述256种不同的颜色强度。存储在后台缓冲区中的像素颜色数据，往往都是以32位颜色值来表示。

3，渲染流水线概述

渲染流水线是以摄像机位观察视角而生成2D图像的一系列完整步骤。

输入装配器阶段，顶点着色器阶段，外壳着色器阶段，曲面细分阶段，域着色器阶段，几何着色器阶段，（流输出阶段），光栅化阶段，像素着色器阶段，输出合并阶段。

4，输入装配器阶段

输入装配器阶段会从显存中读取几何数据（顶点和索引），再将它们装配为几何图元（如三角形和线条等）。

4.1，图元拓扑

通过顶点缓冲区和图元拓扑来告知Direct3D如何用顶点数据来表示图元。

除了少数情况外，本书中大多使用三角形列表。

1，点列表

2，线条带

n+1个顶点就会生成n条线段。

3，线列表

2n个顶点就会生成n条线段。

4，三角形带（图见书本）

n个顶点可生成n-2个三角形。

5，三角形列表

每3n个顶点会生成n个三角形。

6，具有邻接数据的图元拓扑

对于存有邻接数据的三角形列表而言，每个三角形都有3个与之相邻的邻接三角形。

7，控制点面片列表

将顶点数据解释为N个控制点的面片列表，此图元常用于渲染流水线的曲面细分阶段。

4.2 索引

四边形的顶点列表如下：

Vertex v[4] = {v0,v1,v2,v3} UINT indexList[6] = {0,1,2, 0,2,3}

八边形的索引列表：

UINT indexList[24] = {

0,1,2, 0,2,3, 0,3,4, 0,4,5, 0,5,6, 0,6,7, 0,7,8, 0,8,1}

5，顶点着色器阶段

我们可以把顶点着色器看作是一种输入与输出数据皆为单个顶点的函数。

for(UINT i = 0; i < numVertices; i++)

outputVertex[i] = VertexShader(inputVertex[i])

5.1 局部空间和世界空间

局部坐标系：以目标中心为原点，并且坐标轴与该物体对齐的简易便用坐标系。

将局部坐标转换为世界坐标的过程叫做世界变换，变换矩阵叫世界矩阵。

方法一：根据物体的位置与朝向，指定其局部空间坐标系的原点和诸坐标轴相对于全局场景坐标系的坐标，再运用坐标变换即可。

方法二：定义一系列的变换组合W，即W=SRT。

首先，缩放矩阵S将物体缩放到世界空间，其次旋转矩阵R用来定义局部空间相对于世界空间的朝向，最后平移矩阵T定义的是局部空间的原点相对于世界空间的位置。

5.2 观察空间

为摄像机赋予一个局部坐标系，也被称作观察空间，视觉空间或摄像机空间。

在此坐标系中，该虚拟摄像机位于原点并沿z轴的正方向观察，x轴指向摄像机的右侧，y轴则指向摄像机的上方。

由世界空间至观察空间的坐标变换称为取景变换，此变换所用的矩阵称为观察矩阵。

只要给定摄像机的位置，观察目标点以及世界空间中“向上”方向的向量，我们就能构建出对应的摄像机局部坐标系，并推导出相应的观察矩阵。

XMMATRIX XM\_CALLCONV XMMatrixLookAtLH(

FXMVECTOR EyePosition,//虚拟摄像机位置

FXMVECTOR FocusPosition,//观察目标点

FXMVECTOR UpDirection//世界空间中向上方向的向量)

5.3 投影和齐次裁剪空间

下一个任务是将平截头体内的3D几何体投影到一个2D投影窗口之中。

1，平截头体

在观察空间中，我们可以通过近平面n，远平面f，垂直视场角a以及纵横比r这四个参数来定义一个：以原点作为投影中心，并沿z轴方向进行观察的平截头体。

可以通过垂直视场角a和纵横比来确定水平视场角β。

2，透视投影矩阵

若点(x,y,z)位于平截头体之中，当且仅当：

-1 <= x' / r <= 1

-1 < = y' <= 1

n <= z <= f

以NDC坐标来表示的x轴和y轴上的投影坐标：

x' = x / rztan(a/2)

y' = y / ztan(a/2)

透视投影矩阵公式见书本。

可以使用XMMatrixPerspectiveFovLH函数来构建透视投影矩阵

XMMATRIX XM\_CALLCONV XMMatrixPerspectiveFovLH(

float FovAngleY,//弧度制表示的垂直视场角

float Aspect,//纵横比 宽度/高度

float NearZ,//近平面距离

float FarZ

)

6，曲面细分阶段

曲面细分阶段是利用镶嵌化处理技术对网格中的三角形进行细分，以此来增加物体表面上的三角形数量。再将这些新增的三角形偏移到适当的位置，使网格表现出更加细腻的细节。

使用曲面细分的优点：

1，实现一种细节层次(LOD)机制，使离虚拟摄像机较近的三角形经镶嵌化处理得到更加丰富的细节，对距摄像机较远的三角形不进行任何更改；

2，我们在内存中仅维护简单的低模网格，再根据需求为它动态地增添额外的三角形，以此节省内存资源；

3，我们可以在处理动画和物理模拟时采用简单的低模网格，而仅在渲染的过程中使用经镶嵌化处理的高模网格。

曲面细分是一个可选的渲染阶段，可在用户需要之时才开启。

7，几何着色器阶段

几何着色器是一个可选渲染阶段。

几何着色器的主要优点是可以创建或销毁几何体。

我们可以利用几何着色器将输入的图元拓展为一个或多个其他图元，抑或根据某些条件而选择不输出某些图元。

顶点着色器与之相比，则不能创建顶点。几何着色器的常见拿手好戏是将一个点或一条线扩展为一个四边形。

8，裁剪

完全位于视椎体之外的几何体需要被丢弃，而处于平截头体交界以外的几何体部分也一定要接受被裁剪(clip)操作。

裁剪操作是由硬件来负责的，所以我们也就不再赘述其具体的实现细节。

9，光栅化阶段

光栅化阶段的主要任务是为投影至屏幕上的3D三角形计算出对应的像素颜色。

1，背面剔除

每个三角形都有两个面。

如果组成三角形的顶点顺序为v0,v1,v2，那么，我们通过下述方法来计算此三角形的法线n：

e0 = v1 - v0； e1 = v2 - v0; n = e0 x e1 / |e0 x e1|

法向量由正面射出，另一面则为背面。

根据观察者的视角看去，顶点绕序为顺时针方向的三角形为正面朝向。

背面朝向的三角形都被正面朝向的三角形所遮挡，所以绘制它们是没有意义的。

背面剔除就是用于将背面朝向的三角形从渲染流水线中除去的处理流程。这种操作将待处理的三角形重量削减一半。

2，顶点属性插值

从本质上来说，插值法即利用三角形顶点的属性值计算出其内部像素的属性值。

10，像素着色器阶段

我们编写的像素着色器是一种由GPU来执行的程序。它会针对每一个像素片段（片元）进行处理（即每处理一个像素就要执行一次像素着色器），并根据顶点的插值属性作为输入来计算出对应的像素颜色。

11，输出合并阶段

通过像素着色器生成的像素片段会被移送至渲染流水线的输出合并阶段。

在此阶段中，一些像素片段可能会被丢弃（例如，未通过深度缓冲区测试或模板缓冲区测试的像素片段）。而后，剩下的像素片段将会被写入后台缓冲区中。

此技术可令当前处理的像素与后台缓冲区中的对应像素相融合，而不仅是对后者进行完全的覆写。

一些透明这样的效果，也是由混合技术来实现的。