### 一、渲染流水线的三个概念阶段

图示

描述已自动生成

#### 1、应用阶段

这个阶段中开发者需要做的——准备好场景数据、粗粒度剔除、设置好每个模型的渲染状态。输出渲染图元交给几何阶段处理。在这个阶段开发者拥有绝对的控制权。通常由CPU负责实现。

粗粒度剔除：包括视锥体剔除和遮挡剔除两种。剔除不可见的物体，减少无效数据，移交给GPU的数据更精简。

渲染图元：渲染所需的几何信息。比如点、线、三角面等等。

#### 2、几何阶段

用于处理所有和要绘制的几何相关的事情。这一阶段将会输出屏幕空间的二维顶点坐标、每个顶点对应的深度值、着色等相关信息。并把这个信息传给光栅化阶段。在这个阶段开发者无法拥有绝对的控制权，通常在GPU上运行。

#### 3、光栅化阶段

使用几何阶段传来的数据产生屏幕上的像素渲染出图像。光栅化的主要任务是是决定每个渲染图元中哪些像素应该被绘制在屏幕上。在这个阶段开发者无法拥有绝对的控制权，这一阶段在GPU上运行。

### 二、渲染流水线

#### 1、CPU和GPU之间的通信（应用阶段）

渲染流水线的起点是CPU。CPU主要负责数据的处理，GPU负责渲染图形图像。当对画面的精度要求越来越高，如果用CPU进行渲染就会使得帧率大幅度下降（Catlike Coding Basic（五）是一个例子），因此需要ALU更多的GPU去代替CPU去进行这些浮点运算。GPU无法直接访问内存取得数据，需要CPU将数据传给GPU，在每次Draw Call之前CPU发送渲染所需的数据检测渲染状态、提交渲染所需要的数据、提交渲染所需要的状态。CPU通过应用程序向图形接口（例如OpenGL，DirectX）发送渲染命令Draw Call，图形接口再向显卡驱动发送渲染命令，显卡驱动把渲染命令翻译为GPU能理解的代码然后进行绘制。

CPU和GPU是通过一个命令缓冲区实现并行工作的从而完成渲染流水线。CPU向命令缓冲区中添加命令，GPU从中取出命令，添加和取出的过程互相独立。

1. 把数据加载到显存中

网格数据需要网格和纹理等数据需要被加载到显存中。因为显卡对显存访问速度更快，并且大多数显卡对于内存没有直接访问的权力。

1. 设置渲染状态

渲染状态定义了场景中的网格是如何被渲染的。包括顶点着色器、片元着色器、光源属性、材质等。

1. 调用Draw Call

Draw Call是一个命令，发出方是CPU，接收方是GPU。这个命令指向一个需要被渲染的图元。当GPU接受到Draw Call之后就会根据上一步确认的渲染状态和顶点数据进行计算，在屏幕上输出图像。

#### 2、GPU流水线（几何阶段、光栅化阶段）

图示

描述已自动生成

上图中绿色表示该流水线阶段是完全可编程控制的，黄色表示该流水线阶段是可配置但不可编程的，蓝色表示该流水线阶段由GPU固定实现（开发者不可配置、不可编程）。实线表示该Shader必须由开发者编程实现，虚线表示该Shader是可选的。

顶点着色器：通常用于实现顶点的空间变换、顶点着色等功能。顶点着色器的处理单位是顶点，即输入进来的每个顶点都会调用一次顶点着色器。GPU会利用本身的特性并行化去处理每个顶点。顶点着色器完成的主要工作是坐标变换和逐顶点光照。

曲面部分着色器：用于细分图元。

几何着色器：用于执行逐图元的着色操作，或者被用于生产更多的图元。（图元着色这个阶段做了什么？）

裁剪：将不在摄像机视野内的顶点裁剪掉，并剔除某些三角图元的面片。裁剪是为了不处理摄像机视野范围之外的物体。完全在视野之内的图元会继续传给下一个流水线阶段，完全不在视野内的图元不被渲染所以不会继续传给下一个流水线阶段，如果一个图元部分在摄像机视野内，就需要进行裁剪。以视野边界为界线，边界之外的是被裁剪掉的部分，处于视野边界上的点成为裁剪之后新的顶点。

屏幕映射：把每个图元的坐标转换到屏幕坐标系中。屏幕映射的任务是把每个图元的x和y坐标转换到屏幕坐标系下。屏幕坐标系是二维坐标系，因此屏幕映射不对输入的z坐标进行处理。输出网格顶点相关的信息给光栅化阶段。（片元和像素的区别？）

三角形设置：这个阶段会计算光栅化一个三角网格所需的信息。是计算三角形网格表示数据的过程。

三角形遍历（扫描变换）：检查每个像素是否被一个三角网格覆盖，若被覆盖则生成一个“片元”。这一步输出一个片元序列——一个用于计算每个像素最终颜色的状态集合。

（？）片元着色器：片元着色器的输入是上个阶段对定点信息进行插值得到的结果，输出一个或多个颜色值。

逐片元操作：1.这个阶段决定每个片元的可见性。2.如果一个片元通过了所有测试（模板测试、深度测试。模板测试通常用于限制渲染的区域，也可以渲染阴影或者渲染轮廓；深度测试和透明效果有密切关系。测试顺序不唯一，测试在流水线中片元着色器阶段之前。），就要把这个片元的颜色值和已经存储在缓冲区的颜色值进行合并或者更新（取决于是否打开混合选项）。