基于GPU多视图渲染

的多路空间复用裸眼立体显示技术

摘要：多路空间复用裸眼显示系统通过显示由多张图合成的图像来为多个用户提供足够的可视区域。传统显示方式需要在多个通道中渲染。在本文中，我们为多路空间复用裸眼立体显示提出一种基于GPU的多视图渲染（GBMVR）的方式。这种方式在一个通道的几何渲染阶段从多个视图中分别生成多张图片，在另一个通道的表面渲染阶段将多张图片合成为一张图片。其次，还实现了Google Chromium浏览器的兼容，因此例如雷神之锤3的很多传统基于OpenGL的应用程序不用更改任何源码，也不用重新编译就可以直接在经过GBMVR增强的Chromium浏览器上运行。通过配置并行渲染结构，这种方式也适用于高分辨率多路渲染裸眼立体显示平面投影仪。

摘要：立体显示；GPU；插件；

1．简介

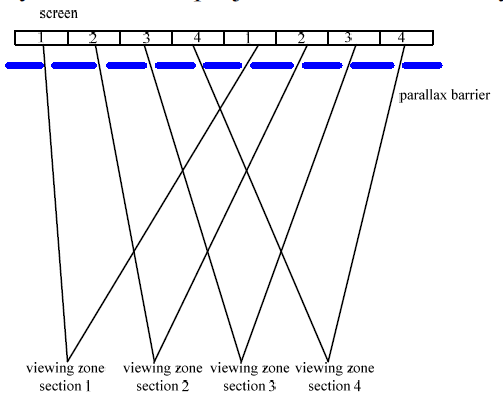
裸眼立体显示提供了一种沉浸式的3D显示体验，这种方式不需要要眼镜或者其他特殊的设备。裸眼立体显示有很多应用场景，比如娱乐（游戏和电影）、模拟系统和虚拟现实。最新的多路空间复用裸眼立体显示系统可以显示多个视图，因此可以为多个观看用户提供足够的观看区域。这些视图都是被一组相机所看到的或者通过计算机图形学的方式生成的。这些视图有一些数据冗余，在多视图渲染的时候这些冗余会影响渲染效率。

在本文中，我们提出一种基于GPU的多视图渲染方法，这种方法只在一个渲染通道中运行多路裸眼显示几何渲染Shader。为了让我们的方法具有更广泛的适用性，我们还实现了Chromium浏览器的兼容，通过在Chromium中拦截并修改运行在Chromium中的程序发给OpenGL图形库的命令。因此，GBMVR方法可以支持只有单一视窗输出的应用程序。

2．前期工作

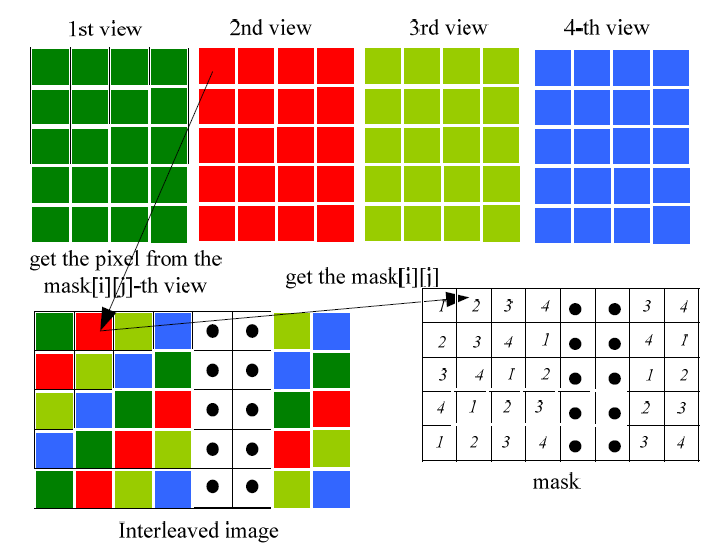
2.1 裸眼立体显示

在第20世纪内，裸眼立体显示的研究有了很大的进展。目前几乎所有的裸眼立体显示技术都可主要归为两大类：空间多路复用显示和多投影机投影显示。空间多路复用裸眼立体显示系统是显示由多个视图合成的图像。而多投影机投影立体显示系统是每个视图都是由同时显示一个或者多个摄像机拍摄的图片。



**图2-1 光屏障式立体显示可视区域**

多路空间传输显示依靠视线遮挡例如在光屏障式形成观察区域中的视差遮挡板。视差挡板是由一列窄缝组成的光栅片，如图2-1，在每一个可视区域内，只可以看到窄缝后的一部分图像的像素。显示的图像是通过一个遮罩从多个视图中隔行提取合成的。遮罩是一个同最终合成的图像相同大小的二维数组，设mask[i][j]为第行第j个元素，那么mask[i][j]储存着视图的下标。如图2-2，最终合成的图像的第i行第j个像素就是根据mask[i][j]储存的下标，从第mask[i][j]个视图取得的像素。



**图2-2 由四个视图隔行扫描合成的图像**

2.2 GPU编程

最新的英伟达显卡提供了三种可编程模块：顶点着色器，几何着色器和表面着色器。几何着色器同顶点着色器类似都是执行数学运算。不同在于，几何照射器由于可以生成新的图元所以可以输出多个图元。因此可以处理以一个输入而需要多个输出的问题。

GPU广泛的运用于为程序的耗时数学计算部分加速。例如，几何着色器被用来分割投影四面体的4D展示。Sorbier等人提出一种通过在单个几何着色器的单个通道中将多个视图分别生成多张图片的方法。但是他们的方法没有兼容性，并且不能支持高分辨空间复用裸眼立体显示。Kooima等人提出了一种基于GPU的图像交织算法来交错多个视图的方法，这种方法是通过片段着色器实现的。他们的方法需要在在两个通道内为左右两个视图分别渲染。

2.3 Chromium浏览器

Chromium浏览器是开源并且兼容性很强的系统，它是为工作站集群的交互渲染而设计的。Chromium可以截取应用程序节点调用的OpenGL命令。这些被截获的OpenGL命令由SPU处理（流处理单元）。通过SPU，开发者可以修改或者替换任意由应用程序调用的图形命令。此外，Chromium提供了为高分辨率显示的sort-first渲染架构。

3．多视图渲染

3.1 基于GPU的多视图渲染

基于GPU的多视图渲染方式在两个渲染通道中交织图像，在第一个渲染通道里，GPU将多个视图生成到一张图片中。在第二个渲染通道里，GPU在片段着色器中用第一个通道中生成的图片交织出最后的图片。

1. GPU将多个视图生成到一张图片里

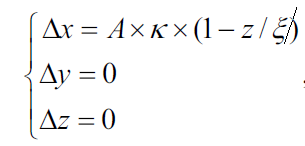
在我们的方法中，当应用程序只为一个视图计算图元时，GPU在几何着色器中为所有视图生成图元。

1. 视差

其他视图的图元是通过GPU从几何着色器的输入图元几何变换生成的。该变换由以下步骤组成：

步骤1.视图变换，沿着X轴移动图元，通过一个可以配置的参数K，K为视点之间的距离，K是一个用来控制变换的值。

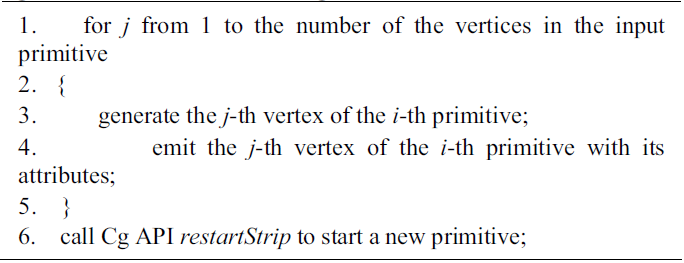
步骤2. 视差角度分布变换，将准备好的图片通过变换投影矩阵沿着X轴调整视差分布。为了控制变换值，我们定义了一个可配置参数ξ，表示焦平面的深度。实际上，这里在两个视图中只有水平视差，根据下面的变换公式，我们可以得到一个顶点的视差ΔH(Δx, Δy, Δz)。



在A =(2×Zn)/(r-l)公式中，r和l是左右顶点裁剪上的坐标，Zn是近深度裁剪面的距离，z是顶点在三维空间的深度。

1. 通过几何着色器的多视图图元生成

和Sorbier的方法不同，我们的方法渲染多个视图到一张图片上，由于这些渲染视图在纹理的内存中一个挨着一个对齐排列，视口沿着X轴被分为多个子窗口。我们假设输入图元输入第一个视图。如图2-3所示，可以根据输入的图元，生成其他视图的图元。

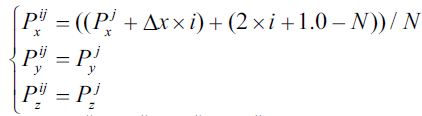


**图3-3 生成第i个视图的第i个图元的伪代码**

为了从输入图元Pj的第j个顶点生成第i个图元的第j个顶点图元Pij，GPU需要执行以下步骤：

步骤1. 计算Pij的坐标。

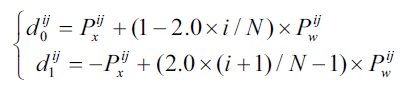
Pij是Pj和Pj的视角的和，为了将第i个视图渲染到第i个子窗口，Pij只能在x轴做变换，因此，计算公式如下：



在图中，(,,,)是的齐次坐标，(,,,)是的齐次坐标。

步骤2. 计算的裁剪距离

由于所有视图共用一个由OpenGL程序从单个视图中生成的视椎体，如果不裁剪一些图元可能会被画到隔壁的子窗口中。因此，每个顶点都需要计算并设置顶点同子窗口左右边缘的裁剪距离。在光栅化阶段，如果启用了用户自定义裁剪面，图形管线会根据裁剪距离裁剪图元。两个计算顶点裁剪距离的公式如下所示：

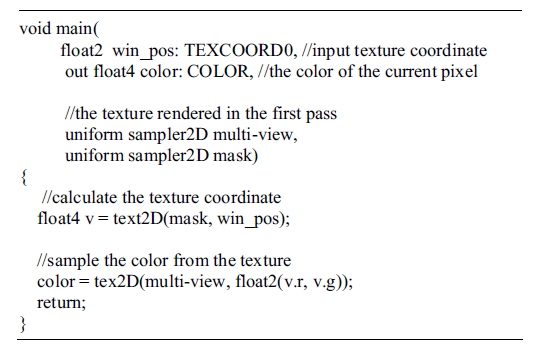


图中，和是从到其子窗口左右边界的裁剪距离。

1. 将多个视图渲染到一张图片上

通过几何着色器，GPU渲染多个视图到一个帧缓存中，在调用OpenGL的API：glSwapBuffers前，应用程序将多个视图复制到规定的纹理内存中。

1. 在片段着色器的第二个通道中根据图像合成图片。



**图3-4 合成算法伪代码**

为了提高运算速度，在遮罩图片中储存了纹理坐标的采样值，因此，可根据遮罩图片的像素颜色进行采样纹理坐标。片段着色器的交织处理如图2-4。

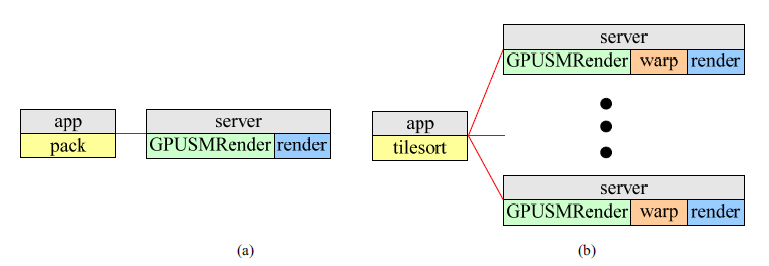
3.2 基于GPU的多视图渲染

通过Chromium的基于GPU的多视图渲染方法，GBMVR由5个模块组成：GPU渲染模块，曲面模块，标准打包模块，标准排序模块和标准渲染模块。GPU渲染模块封装了基于GPU的多视图渲染方法，校正模块并行处理几何校正和亮度校正，当标准排序模块传输几何数据到下游服务节点时打包模块复杂传输。渲染模块发送所有几何信息到系统的OpenGL库，调用OpenGL渲染。

1. GBMVR配置

GBMVR可以通过配置来支持在pc上裸眼立体显示，如图3-5（a）。在图3-5（a）中有一个应用程序节点和一个服务器节点。应用程序节点拦截几何信息并且将信息传输到服务器节点。服务器节点通过基于GPU的多视图渲染方法交织出图像。

此外，GBMVR在计算机集群还可以被配置为支持高分辨率空间多路复用裸眼立体显示，如图3-5（b），在图中有一个应用程序节点和多个服务器节点。应用程序节点拦截几何信息。标准排序模块对几何数据进行快速排序划分，然后将数据发送到多个下游服务器节点，每一个服务器节点渲染整个巨大的合成图像的一部分。校正模块确保让所有图片显示在一起无缝并且亮度相同。



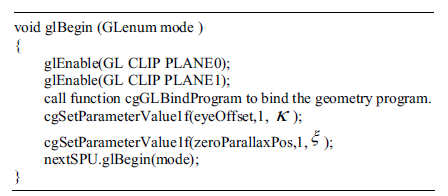
**图3-5 空间多路复用裸眼立体显示配置**

1. GBMVR模块

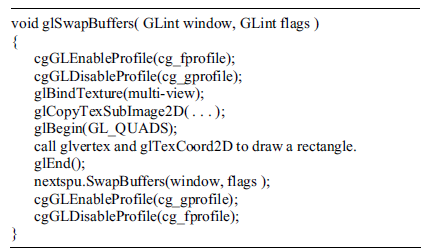
GBMVR模块被用来在Chromium中支持基于GPU的多视图渲染。该模块重新实现了glBegin 和glSwapBuffers方法。

如图3-6，为glBegin函数的伪代码。该函数首先打开了用户自定义裁剪面功能，让后绑定几何程序，接着调用CG API函数cgSetParameterValue传输参数K和ξ到几何程序，最后调用glBegin函数转到下一个模块。

如图3-7,，为glSwapBuffers函数的伪代码。首先，打开片段着色器并且关闭几何着色器。接着绑定纹理并且将多个视图的信息从帧缓存中拷贝到纹理中。接着，画一个矩形让表面着色器从纹理中交织图片。然后调用glSwapBuffers函数转到下一个模块。最后，打开几何着色器并关闭片段着色器开始下一帧渲染。



**图2-6 glBegin函数伪代码**



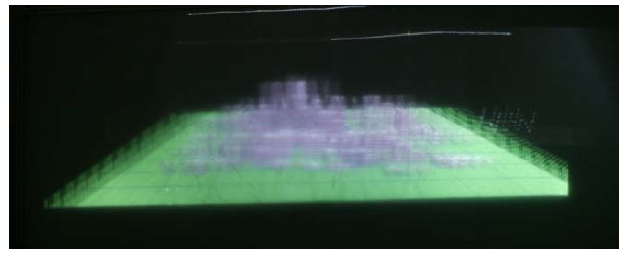
**图3-7 glSwapBuffers函数伪代码**

4．实验结果

如图4-8显示图片是从游戏雷神之锤3中通过多视图合成的。图4-9展示了一张高分辨率图片，这张图片显示在一块由12个平铺排列投影仪组合而成的3.6x1.6㎡的屏幕上。这个屏幕是为我们的多投影仪裸眼显示设计的。图像模糊是因为在屏幕上的透镜光栅模糊了图像。



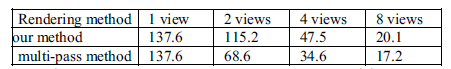
**图4-8 在PC上通过GBMVR多视图合成运行游戏雷神之锤3**



**图4-9 在7台PC上通过GBMVR合成4视图高分辨图像**

我们在pc上测试我们的方法，pc配置为Inter Core i7 920 2.67GHz Cpu，英伟达GeForce 9600GT GPU。排除Chromium传输的开销，我们在PC上比较了基于GPU的多视图渲染方法和传统多通道渲染方法的效率。用两种方式的不同视图个数渲染同一个3D模型沉思者（79万个面）。运行帧率列表见表4-1。显然，基于GPU的多视图渲染方式比其他方式更快。当视图数量为2时我们的方法极大的提高了运行效率。当视图数大于4时，两种方法没有明显的差距，这是受几何着色器的计算能力所制约。

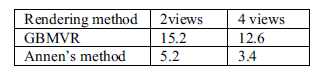
**表 4-1 两种方式渲染思考者模型时的帧率**



此外，我们在7台计算机上比较了GBMVR和Annen的方法。两种方法都以2个视图和4个视图分别运行游戏雷神之锤3。帧率列表如表4-2。显然，GBMVR方式比Annen提出的方法快得多。原因如下：

1. Annen的渲染方式在多个通道中渲染多张视图。
2. 为了渲染K个视图，Annen的方法需要在一帧从应用程序端传输K次指令到每一个服务器节点，然后我们的方法只有传输一次。

**表 4-2 两种方式运行雷神之锤3时的帧率**



5．结论

在本文中，我们提出一种基于GPU多视图渲染的多路空间复用裸眼立体显示技术，GBMVR在几何着色器中的一个通道中将多个视图渲染到一张纹理上，但是传统的方式在多个通道中渲染。很多传统的OpenGL程序（例如雷神之锤3）不用更改任何原代码或是重新编译就直接可以在我们的系统上运行。此外，GBMVR还支持高分辨率多路空间复用裸眼立体显示。