

The very basic Renderman (四)

Prman实际应用解决

任何软件无论多强大多厉害若不能在实际中得到应用则如同一件占硬盘空间的废物。下面我们就说说Prman在实际中如何应用。

一、为什么用Prman，什么场合用Prman

Prman的特点就是快、精确、细腻。所以在一些高精度的制作中适合用Prman。

Prman不附属于其他程序，使用独立的进程，独占内存块，有利于高速渲染，而且对SMP和Netrender支持十分好，

当使用ShadingRate 0.8时Prman的精度已经几乎无人能及，其他渲染器就必须使用更高的Resolution渲染然后缩小才可以达到这样的精确度。

但Prman没有光线跟踪，没有GI，不适合做效果图、建筑设计，也不太适合做电视片头。

所以Prman主要应用在电影特技、角色动画、视觉效果设计、艺术设计等纯视觉应用领域上。这也是为什么几乎每年奥斯卡最佳视觉效果奖的电影都应用了Prman渲染。

二、Renderman疑难解决

很多人反映Renderman很难，但这是制作观念没有改变，因为Renderman的制作观念和其他的渲染器不一样，制作思路要适应所以很多人觉得难。

真正掌握了Renderman的制作思路后就不会觉得难了，反而会觉得十分科学，而且越修炼下去就越能掌握提高渲染速度的技巧和Shader的制作技巧。

1.加速渲染

有的朋友加速渲染只着眼于机器的速度，不想自己动动脑子想想。但这是治标不治本的，因为大家的银子并不多，而且大型的场 景加上ShadingRate 0.8左右的参数会大量消耗机器的运算量！特别是内存，很快就不够用了。渲染时间在制作上是很宝贵的，特别是在时间就是金钱的今天。Prman制霸电影界的其中一个原因也是它的渲染速度快。商业制作中大部分都是制作动画而不是静帧。

-1- 渲染分类

严格区分Preview和Produce的ShadingRate。很多人喜欢ShadingRate低的渲染效果，因为他细腻、漂亮。但未做成成品前一定要严格地分开ShadingRate设置。经验丰富的Renderman技术人员在Shader的Preview里就可以看到真正成品的结果，因为在Icon的旁边我们可以设置Shader Preview的ShadingRate。

-2- 不要盲目追求低ShadingRate

其实Renderman的ShadingRate也是有很大的弹性的，设置不同的ShadingRate不但只控制细腻与否，还可以控制Shader的表现效果的不同。比如Displacement里加了Fractal，如果使用太小的ShadingRate，纹理就会显得太粗糙，不好看。若ShadingRate控制在1.0左右，纹理就比较光滑，容易表现石头、水泥地等质感。

-3- Map的使用

RAT里的Map是可以重用的，所以可以用一个笨办法来加速Map的渲染，就是先用比较高的ShadingRate渲染Map，然后用比较低的ShadingRate渲染画面。比较聪明的办法就是加入RIB box。RIB box用于

在渲染时区分特定的场合在RIB文件里加入特定的语句。

例如：

```
[if {$ELEMENTTYPE == "shadow"} { return "ShadingRate 5" }][if {$ELEMENTTYPE == "shadow"} { return "PixelSamples 1" }]
```

这样在渲染Shadow的时候系统会在RIB文件里把Shadingrate改成5、把PixelSamples 改成1 X 1。这样在渲染Shadow的时候就可以加快速度。当然，这种加速在单帧渲染时不会有太大的体现，我尝试过渲

染一幅比较简单的图，用了SoftShadow，在没有加RIB box的情况下用了2m 34s，用了RIB box的情况下2m 27s，速度只快了几秒，但如果是渲染动画，这几秒在整个动画渲染中看来就十分重要。一帧省了7秒，十帧就省了70秒，如果是一分钟的动画就省了165分钟的渲染时间啊！！

何况在商业制作中，场景中往往使用了多个光源，而且还有多张环境贴图，这样就可以把渲染速度大大地提高很多倍。

-4- 分层渲染

分层渲染是最近很热门的一个课题，因为现在人们制作的动画越来越复杂，复杂程度大大超过硬件渲染可以承受的程度。象Renderman这样的纯视觉渲染器对分层渲染的支持是十分好的（分层渲染在GI中是不适用的）。分层渲染是十分有用的，可以更大地加快渲染速度，比上面介绍的RIB box更快。渲染的时候渲染器会先读入场景的数据资料，然后放在内存里。Renderman的Reyes算法会把多边形剖分成很小的微多边形，直到渲染完才把内存回收。但如果场景十分复杂，Reyes算法需要的内存就成倍地增加，特别是在ShadingRate低的场合，一般的工作站很快就被吃光了内存。而且内存不足的时候系统会使用虚拟内存，使用虚拟内存会大大减慢渲染速度，在渲染动画的时候这种减速是绝对致命的！分层渲染的好处是在保证质量的同时加快渲染速度。分层渲染一般把场景分成几部分，如背景、静物和角色，分开渲染，记录必要的通道，然后再在合成软件里合成。分层渲染要求把背景和角色分开渲染，这也是为什么Prman不加入光线跟踪的原因之一。Prman支持自己写DisplayServer，也就是自己指定的输出程序，prman会自己把相应的信息输出到指定的程序模块里。RAT已经提供了十分足够的渲染输出支持，包括P——渲染的点信息、N——渲染的法线信息、stuv——贴图和曲面的坐标信息、Cs Os Ci Oi——颜色信息等等。基本的输出支持rgbaz的输出。关于输出和合成的知识是所有软件都基本通用的，在这里不做详细的叙述了。

-5- 尽量使用Nurbs

Prman的Nurbs渲染速度是世界第一的，但它渲染多边形就很慢了。因为大型的动画公司都有Nurbs的转换软件和三维扫描软件，所以在制作中可以几乎全用Nurbs。不过按照SIGGRAPH中SQUARE USE的技术人员的描述，他们也大量地使用贴图，而且不担心UV的问题，也不使用DeepPaint之类的软件。他们解决贴图其实是用了Project技术来解决的。RAT里有magic surface这一神奇的Project解决方法，它可以用mtorCoordSys来生成贴图面，把贴图投影到物体表面上，就像用贴纸贴上去一样，而且还可以Sticky，和物体表面结合，物体变形而贴图的形状跟随变形，不会出现普通的投影贴图不跟随变形的情况。

2. 没有光线跟踪的日子

在Prman里，没有Rayserver就没有光线跟踪，没有光线跟踪就很难做出反射/折射的效果。但很难不代表不行，而且Prman制作金子的漂亮程度绝对不亚于光线跟踪渲染器。当然，这必须花一点苦功才行，要修炼出漂亮的金子效果，就要把EnvMap做熟。在此我要卖个关子，因为Prman是个需要用户自己解决能力很高的软件，要学好最根本的还是要培养自己的解决能力。缺少了开发Prman比任何一个渲染器都差劲。提示一下，要做反射折射就要涉及到Slim里的几个与EnvMap有关的Shader。ratEnvironment有bug，不要用。

3. SemiTransparecy

一道难题，十分难。在透明物体的背后会留下半透明的阴影，当然还有聚焦。这就是半透明效果。在Prman里要做出这种效果牵连甚广，必须模拟，不能直接实现（除了Rayserver）。首先，分析一下这种效果。这是一个半透明的影子在物体的背光处。半透明的影子.....其实就是一张贴图，一张从光源望向物体的贴图。怎么做呢？.....Slim里有一种Shader叫MapGen，下面的是ShadowMap、SoftShadow、Reflection、EnvironmentMap、Reference.....这些都是用来做特殊贴图的Shader，连接到物体上就会制作成贴图，.....有了贴图怎么贴上物体上呢？RAT里有一种十分厉害的Shader叫Layer，可以把很多Shader象PS的Layer一样叠加，这样就可以把贴图叠在物体的表面上了。那UV怎么办呢？上文提到过一个叫magic surface的东西，在Slim里就是magic color，可以把贴图投影到指定的表面上（magic color可以在Pixar的网站上下载）。有了这些思路就可以制作出基本的效果，想要更真实的就要自己再研究一下了

4. 大型场景

这样的大型场景是怎么做的呢？

再给出一个思路：用Fur。

分层渲染的时候可以给山川的表面Attach一层Fur，最好是UltraFur，因为可以在Maya里用Paint修饰。修饰完毕后就可以只渲染Fur，输出成tif或者iff文件。在后期合成软件里可以用Glow做出发光的效果，再和渲染出来的场景叠加——OK! Fur还可以用来渲染草和树木等东西，可以自己发挥想象。

三、扩展RAT

Prman最重要的一点是可扩展性。它的扩展可以有：

1.用ShadingLanguage写Shader。真正的写Shader其实不难，反而是最容易的，只要有一点语言的基础就可以写出自己想要的效果

2.用RAT API写RIB Gen。这是DII文件，也就是可以和Prman互交的Shader。严格地说，这不是Shader这么简单，而是自己定义的可以扩展RIB场景的模块。mtorFur和UltraFur就是这种Shader。

。