基于 GPU 的动态头发渲染

毛 伟,陈利学
(西南石油大学 计算机科学学院,四川 成都 610500)

摘 要:运用模拟头发运动的系统计算头发阴影的阴影生成算法和一个通过每一串头发来模拟光线散射的发射模型,就可以创建出极其真实的头发影像。渲染结果表明,利用以上方法可以渲染出极其逼真的头发。

关键词:GPU;发射模型;头发影像
中图分类号:TP399 **文献标识码**:A **文章编号**:1674-7720(2012)03-0037-03

Dynamic hair rendering based on GPU

Mao Wei, Chen Lixue
(School of Computer Science, Southwest Petroleum University, Chengdu 610500, China)

Abstract: In this paper, it includes a system for simulating the hair's movement, a shadowing algorithm to compute hair self-shadowing, and a reflectance model to simulate light scattering through individual strands of hair. When combined, these elements produce the extremely realistic images of hair in real time. Rendering results show that the above method can render very realistic hair.

Key words: GPU; emission model; hair image

头发渲染在图形图像渲染中越来越受到人们的关注。在当今人体虚拟和仿真技术里,头发可以给人体模型一种真实感,而且在电影和动画游戏领域里都有广泛的应用。动态三维头发渲染一直是计算机图形学的一大难题,头发的最终形态、光泽度、颜色以及种类数量都是非常难以控制的。本文讨论了基于 GPU 编程技术的高质量动态头发实时渲染,其中主要讨论和解决了动态头发的建模、动力学和碰撞以及最终的着色渲染几个关键问题。

1 动态头发的建模

1.1 控制头发

受控发丝的结构用于粗略地描绘整个发型。从 Maya 内建的表示“头皮”的专用几何体(渲染时不可见)“生长”出受控发丝。受控发丝从头皮的每个顶点沿着法线生长出来。为了程序上能让头发运动,一旦有了一组受控发丝,就让它们服从物理、动力学和碰撞的计算。在模拟头发运动的系统里,运动完全依赖于动力学,它是一个人工控制系统,需要能“假造”或者“修正”头发的行为。

1.2 数据流

头发几乎每一帧都有动作和变化,所以需要在每一帧中重建最终渲染头发的集合。要得到平滑曲线,首先将动态受控的受控发丝转换成贝塞尔曲线,并镶嵌成平滑线条,然后通过插值来增加头发的密度,插值后的头发集合被送到引擎来做最终帧的渲染。其中使用了一个动态的顶点缓冲区来容纳这些顶点数据。

1.3 镶嵌和插值

1.3.1 镶嵌

头发的镶嵌处理是通过在每根受控发丝上添加顶点以平滑化受控发丝来完成的。这会增加 5 倍以上的顶点,从 7 个顶点增加到 36 个顶点。为了计算新顶点的位置,计算切线并使用这些切线来计算贝塞尔控制点,将受控发丝转换成贝塞尔曲线。通过贝塞尔曲线,计算额外顶点位置,从而使受控发丝更加平滑。经过平滑处理的受控发丝通过插值来复制,以生成一把稠密的头发,为最终渲染作好准备。

1.3.2 插值

插值的头发是由头皮网格拓扑来生成的,如图 1 所示。每个三角形的末端都会有 3 条平滑的受控发丝,要把三角形内部的表面用头发丝来填充,所以把受控发丝的坐标每 3 个一组地进行插值,以构造新的平滑头发。

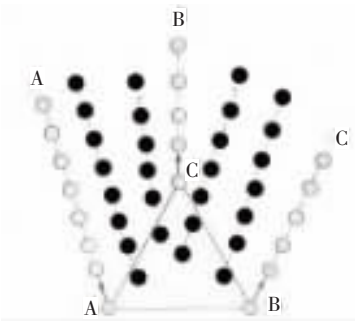

图1 生成插值的头发

平滑的受控发丝和插值的头发具有同样数量的顶点。利用重心坐标来生成新的插值发丝，填满每一个三角形。如插值头发 K 是基于3个重心系数 $(X1,X2,X3)$ 来计算的，其中，$X1+X2+X3=1$，且 $K=A\times X1+B\times X2+C\times X3$.

在区间[0,1]里生成两个随机数，若它们的和大于1，则用1减去较大的数，然后用1减去这两个数得到第3个数，所以这3个数的和就为1，这样就可以确定生成密集头发的位置了。

2 动力学粒子系统

头发的动力学是基于粒子系统的，把每一根没有插值的受控发丝的顶点当成一个粒子运动。这些粒子不是均匀分布在一根头发上的。这些受控发丝上的片段随着和头颅距离的增大而增大。这样，不必添加太多的顶点就可以生长出更长的头发。对于这个粒子的运动，利用Verlet积分来计算，它简单而且稳定。在粒子不停移动的同时，受控发丝的长度也必须保持不变，以免拉伸。因此，在受控发丝的粒子之间使用一些约束条件。若粒子靠得太近，这些约束条件就会使它们相互排斥；当它们距离太远时，会使片段缩短。当然，当拉开一个粒子时，与之邻接片段的长度就会变得无效，所以这样的修改就会重复地使用。当多次迭代之后，这个系统就会向期望的结果收敛，最终保持发根长度为常数。为了保证头发看起来真实，最后就是头发的碰撞问题。在头发的碰撞检测中，受控发丝的碰撞数据中引入一种“珍珠结构”，利用球体来进行碰撞检测，每个球体会与另一个定位在粒子上的球体碰撞，而不是与一个点碰撞。

3 头发的着色

头发的着色问题可以分成头发的局部发射模型和计算头发之间的自阴影的方法两部分。

3.1 用于头发的实时发射模型

局部反射模型选用Marschner模型，它是一个全面的基于物理的头发发射表示。Marschner反射模型可以通过一个四维的双向散射函数来阐明：

$$S(\varphi_i,\theta_i,\varphi_o,\theta_o) \tag{1}$$

其中，$\theta_i\in[-\pi/2,\pi/2]$ 和 $\varphi_i\in[0,2\pi]$ 是在极坐标中的输入方向，$\varphi_o\in[0,2\pi]$ 和 $\theta_o\in[-\pi/2,\pi/2]$ 是极坐标中的光线方向。

函数 S 完整地描述了一根毛发纤维是如何散射和反射光线的。计算这个函数，就能计算出在任何光源位置的表面着色。计算 S 的开销较高，为了避免对每个像素都进行计算，把 S 的结果保存在查找表里，在运行时读取，这个查找表就可以编译成一个纹理，并可以在Pixel shader中访问。但函数 S 有4个参数，而GPU本身是不支持四维纹理的。把1个四维函数编码成二维纹理，小心地处理查找表，就能仅用1个较小的二维映射来对四维函数进行编码。

Marschner模型把每根独立的毛发纤维当作半透明的圆柱体，并考虑可能穿透头发的光线路径。对于3种类型的轨迹，每种都按轨迹符号给了不同的标记。每一道光都以一个字符串来代表其光线和表面的相互作用的类型。R轨迹表示从头发纤维表面反射出来的光向着观察者；TT轨迹表示光折射入头发，而且再次折射出的光向着观察者；TRT轨迹表示光折射入头发纤维，在内表面反射，再次折射出的光向着观察者。其中，“R”表示光线反射，“T”表示光线穿过表面的折射。图2展示了这3种反射轨迹的头发外观。

图2 反射轨迹

因此，这个反射模型的形式为：

$$S=S_R+S_{TT}+S_{TRT} \tag{2}$$

每个 S_P 项可以进一步分解为两个函数的积：函数 M_P 描述了 θ 角在反射中的影响，函数 N_P 捕获了 θ 方向上的反射。若有一根完美的圆柱形头发纤维，就能把 M 和 N 写成更小的一组角。若定义辅助角 $\theta_d=\frac{1}{2}(\theta_i-\theta_o)$ 和 $\varphi_d=\varphi_i-\varphi_o$，那么式(2)的每一项就可以写成：

$$S_P=M_P(\theta_i,\theta_o)\times N_P(\theta_d,\varphi_d) \tag{3}$$

其中，P=R、TT、TRT。

在这种形式里，只有函数 M 和 N 且它们都只有两个参数。这意味着能对每个函数构造查找表，并将其编码成二维纹理。尽管存储了6个函数，但大多数都是单通道的，可以存储在相同纹理中。M_R、M_{TT} 和 M_{TRT} 每个只占一个通道，所以打包入第一个查找纹理中。N_R 是单通道的，但 N_{TT} 和 N_{TRT} 每个都占3个通道。把 N_{TT} 和 N_{TRT} 同放在第二个查找纹理中。为了改进性能和减少纹理使用，做一个简单的假设：$M_{TT}(\theta_i,\theta_o)=M_{TRT}(\theta_i,\theta_o)$。这就允许把 N_{TT} 和 N_{TRT} 保存在相同的纹理中，并把纹理的数目从3减少到2。虽然这个模型是以角度来描述的，从向量计算出角度需要用反三角函数，这样开销很大，所以不把 θ_i 和 θ_o 从第一次查找传下来，而是计算正弦值：

$$\sin\theta_i=(light\cdot\tan\ gent),\quad \sin\theta_o=(eye\cdot\tan\ gent). \tag{4}$$

这能把 M 写成 $\sin\theta_i$ 和 $\sin\theta_o$ 的函数，节省了着色处

理的计算，加上点工作，也可以计算 $\cos\varphi_d$。首先把视线和光线向量正交投射到头发上：

$$lightPerp=light-(light\cdot\tan\ gent)\times\tan\ gent$$
$$eyePerp=eye-(eye\cdot\tan\ gent)\times\tan\ gent \quad (5)$$

然后从下式：

$$(lightPerp\cdot eyePerp)=\|lightPerp\|\times\|eyePerp\|\times\cos\varphi_d \quad (6)$$

就能计算出 $\cos\varphi_d$：

$$\cos\varphi_d=(eyePerp\cdot lightPerp)\times((eyePerp\cdot eyePerp)\times(lightPerp\cdot lightPerp))^{-0.5} \quad (7)$$

这就剩下 θ_d 还未计算了，把 θ_d 定义成 θ_i 和 θ_o 的函数。因为 θ_i 和 θ_o 来编址的查找表已经准备好了，可以向这个查找表加上一个额外的通道来存储 θ_d。查找表是 CPU 计算的，使用每分量 8 bit 的 128×128 纹理。8 bit 格式需要把数值调整到区间[0，1]，还必须在着色器中添加一个额外的调整因子来抵消相关的饱和度。虽然可以把毛发发射模型用于渲染以带线条表示的头发上，但也可以扩展成实体几何来表示，使用表面主切线中的一个。最后考虑表面的自遮挡，这可以通过乘以一个额外的项 $(wrap+dot(N,L))/(1+wrap)$ 得到，其中 $dot(N,L)$ 是光和法线的点积，$wrap$ 的范围在 0~1 之间，控制了光允许环绕模型多远，简单地近似模拟了毛发之间的光照混合。

3.2 头发中实时的体化阴影

实时应用程序里的阴影通常有模板阴体和阴影图两种计算。不过它们对于高精度几何体的阴影图会出现严重失真。所以在计算时，用一种针对渲染头发阴影设计的近视阴影。非透明的阴影图扩展了一般的阴影图，可处理物体及反失真。不透明的阴影图允许分数的阴影值，它不是简单地进行二元遮挡测试，需要知道在给定像素上有多少光穿透到了深度 Z（在光空间）。这些通过式(8)给出：

$$T(x,y,z)=e^{-k\alpha(x,y,z)}$$
$$\alpha(x,y,z)=\int_0^z r(x,y,z')dz' \quad (8)$$

其中，$T(x,y,z)$ 是光在像素位置 (x,y) 上得到深度 Z 的部分，α 被称为不透明度，$r(x,y,z)$ 则是用来描述在每一个点 (x,y,z) 上的单元距离内被吸收的百分比，值 k 是当 $\alpha=1$ 时选择的一个常量，T 近似于 0（在数值精度上）。这就允许忽略范围在[0，1]之外的 α 值。不透明阴影图的思想是，在一个离散的 z 值集合 $z_0,\cdots,z_1$ 中计算 α 值，然后通过在两个近似的值之间插值来确定 α 值，方程如下：

$$\alpha(z)=\frac{\alpha(z_i)(z-z_i)+\alpha(z_{i+1})(z_{i+1}-z)}{(z_{i+1}-z)} \quad (9)$$

其中，$z_i<z<z_{i+1}$。

这是一个合理的近似，因为 α 是 z 的一个严格递增函数，取 $n=6$，z_0 是光空间中最近平面，而 z_{15} 是光空间中的最远平面，其他平面均匀地分布，因此 $z_i=z_0+id_z$。当 $d_z=(z_{15}-z_0)/16$ 时，$r=0$ 时，表示在头发外面，不管 x 和 y 取什么值，$\alpha(x,y,z)=0$，因此只需在 $z=z_1,\cdots,z_{15}$ 时保存 α 的值。

对于特定的不透明阴影图，在给定不透明阴影图的分片后，必须在点 (x,y,z) 上计算 T 的值。通过在两个相邻的分片上线性插值 α 的值来完成。

$\alpha(x,y,z)$ 的值是 $\alpha(x,y,z_0),\cdots,\alpha(x,y,z_n)$ 的线性合并。

$$\alpha(x,y,z)=\sum_{i=0}^{n-1}(\alpha(x,y,z_i)\times\max(0,1-|z-z_i|/d_z)) \quad (10)$$

可以为所有 16 个数值在顶点着色器里计算 $w_i=|z-z_i|/d_z$。为了提高效率，在顶点着色器中计算这些权重，把它们尽快直接传递到片段着色器中，一旦计算出了权重，和的计算为 $\sum_{i=0}^{15}w_i\alpha(x,y,z_i)$，数据是对齐的，所以单个点积计算为 $\sum_{i=0}^{3}w_i\alpha(x,y,z_i)$。

最后，从光学密度上计算传递，得到一个在 0~1 之间的值来表示光源到达点 (x,y,z) 的光照分量。把着色值乘上这个值来得到头发的最终颜色。

本文演示了从动力学来渲染和着色头发，不透明阴影图除了对渲染头发非常有用，还能用在深度图失效的情况。随着 GPU 越来越灵活，它不仅能承担典型的并行任务（如镶嵌和插值），也包括 CPU 涉及的碰撞检测和物理设置，从而达到较为逼真的渲染效果。最后希望在下一代程序里可以看到更逼真的头发。

参考文献

[1] GREEN S. Real-time approximations to subsurface scattering[A]. In GPU Gems, edited by Randima Fernando, 2004:263-278.

[2] KIM T Y, NEUMANN U. Opacity shadow maps [C]. Proceedings of SIGGRAPH2001, 2001:177-182.

[3] MARSCHNER S R, JENSEN H W, CAMMARANO M, et al. Light scattering from human hair fibers [C]. ACM Transactions on Graphics -Proceedings of SIGGRAPH 2003,2003,22(3):780-791.

[4] MCCOOL M D. Homomorphic factorizations of BRDFs for high-performance rendering [C]. Proceedings of SIGGRAPH 2001, 2001:171-178

[5] 唐勇，刘强，吕梦雅.一种头发动态模拟方法[J].计算机仿真，2006，27(7)，211-213.

（收稿日期：2011-11-18）

作者简介：

毛伟，男，1984 年生，硕士研究生，主要研究方向：计算机应用技术，图形图像与虚拟现实。

陈利学，男，1955 年生，博士生导师，教授，主要研究方向：图形图像，嵌入式开发。