
CGDC
中国游戏开发者大会
CHINA GAME DEVELOPERS CONFERENCE
2014

Hair FX - 游戏里的真实毛发技术

Monier Maher
GameWorks产品经理

王晓峰
高级技术美术

目录

- 简介
- 美术制作
- 毛发运行库流程
- 游戏集成经验分享
- 未来的工作

游戏里的毛发

- 计算机来模拟毛发
  - 毛发数量太多
  - 直到最近GPU的处理能力才打到标准
  - 业内最为期待的次世代特效之一
- NVIDIA HairWorks库
  - 很多NVIDIA工程师不断努力的结晶
  - 目前是一个完整的解决方案了（梳理、模拟和渲染）

《巫师3》E3 演示

《使命召唤10：幽灵》

目录

- 简介
- 美术制作
- 毛发运行库流程
- 游戏集成经验分享
- 未来的工作

美术制作

- 交由我们的技术美术王晓峰同志来演讲

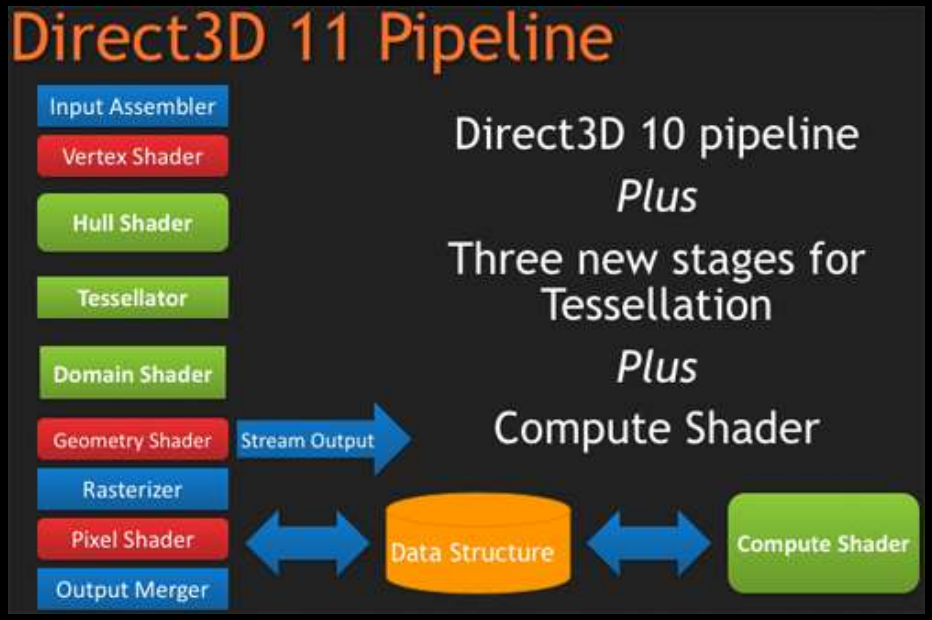
目录

- 简介
- 美术制作
- 毛发运行库流程
- 游戏集成经验分享
- 未来的工作

运行库渲染流程

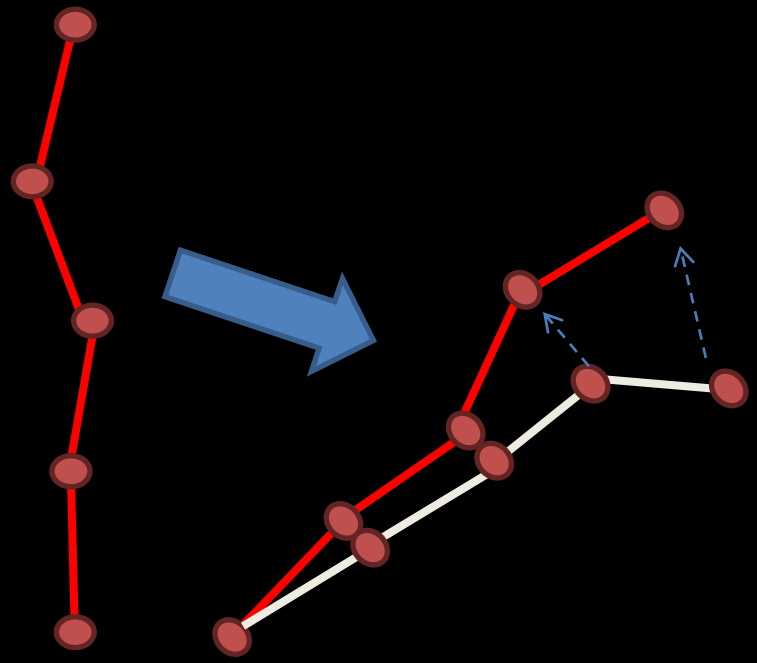
1. 更新growth mesh的蒙皮动画信息，同时更新了毛发的根节点位置
2. 根据蒙皮动画信息，计算、更新毛发曲线的“目标（动画）位置”
3. 对毛发引导曲线(Guide Curve)进行模拟
4. 把引导曲线插值成更精细的样条曲线(spline curve)
5. 在每一个三角形上用tessellation的方式来生成更多的渲染毛发,(DX11 Tessellation)
6. 对所有渲染毛发做最后的阴影和颜色渲染

DX11毛发着色流水线



- 除VS被GS替代以外，我们用到了DX11的所有渲染阶段
- 多遍(Multi-pass)渲染方法
  - 蒙皮和模拟(CS)
  - 样条曲线生成(GS)
  - 毛发插值(HS-DS-GS)
  - 阴影渲染(PS)
  - 最终着色渲染(PS)

毛发的模拟

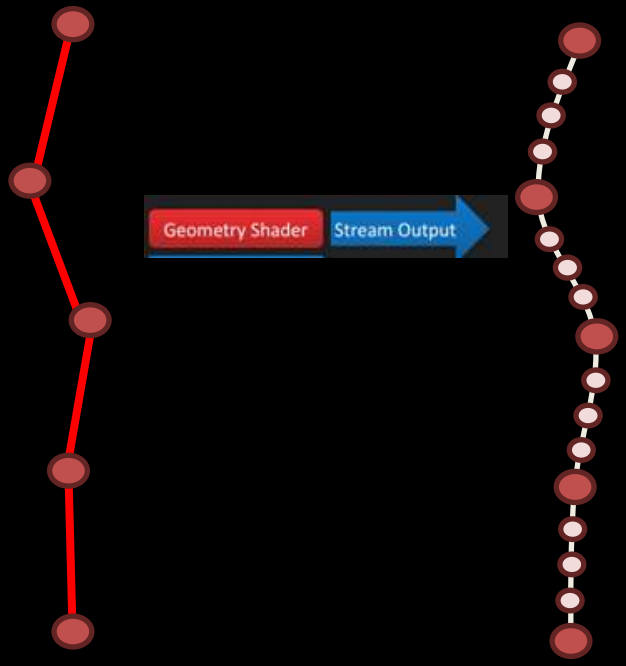


- 使用Compute Shader (CS)
- 根据输入的所有引导毛发曲线（Guide Hair)上的节点
  - 从根节点开始对每个节点做蒙皮
  - 对没有受约束的节点做模拟
  - 每个节点都有可能受到各种约束：形状约束，距离约束，防止拉伸约束等
  - 模拟结果存到一个结构化的存储空间以备其他着色器使用

简化的毛发模拟模型

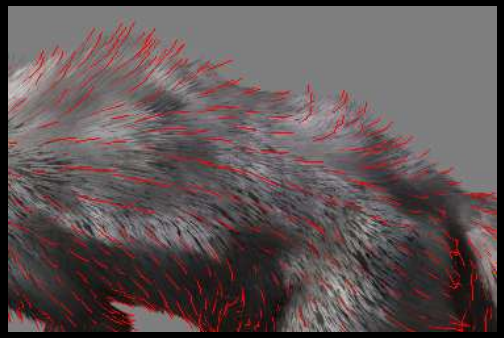
- 毛发节点之间简单的距离约束
  - GPU实现时做红/白式的隔点更新提高并行度
- 防止毛发拉伸的LRA (Long Range Attachment) 算法
  - http://www.matthiasmueller.info/publications/sca2012cloth.pdf
- 保持头发梳理形状的约束
  - 每一个毛发节点都被约束在以蒙皮动画所得的点为圆心的一个球体内
  - 球体半径从根节点起逐渐增大
- 动画蒙皮就是用Compute Shader实现的毛发模拟的一部分
  - 蒙皮支持线性的骨骼矩阵(Bone Matrix)或者八元数(Dual Quaternions)

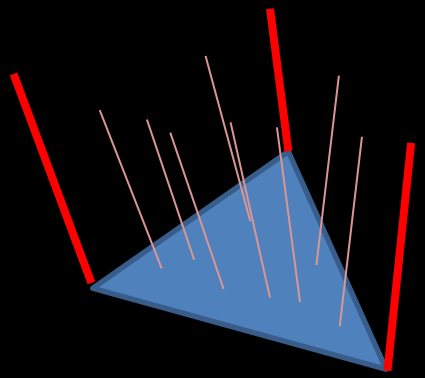
样条曲线(Spline Curve)生成



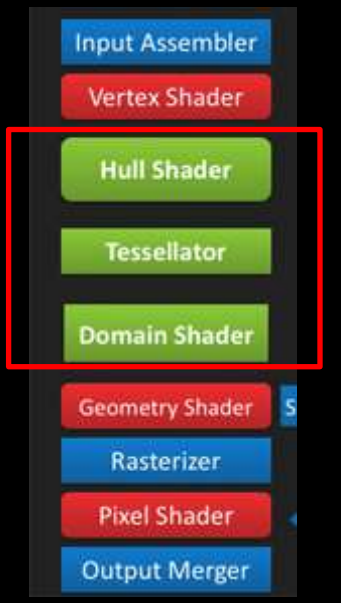
- 模拟的引导曲线节点数较少（如左图）
- 我们用标准的样条曲线插值法生成有更多节点的样条曲线（如右图）
- 新节点的其它属性（如切线等）也会通过插值生成
- 使用GS和SO来完成

用Tessellation引擎插值生成毛发



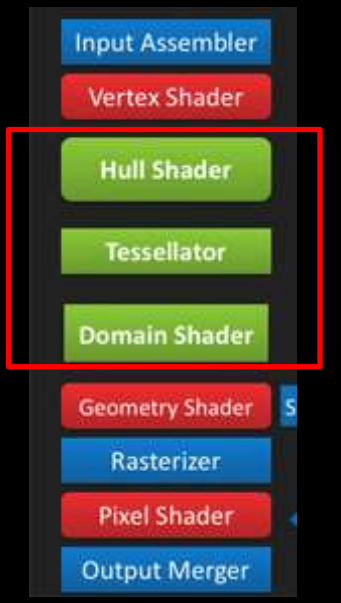


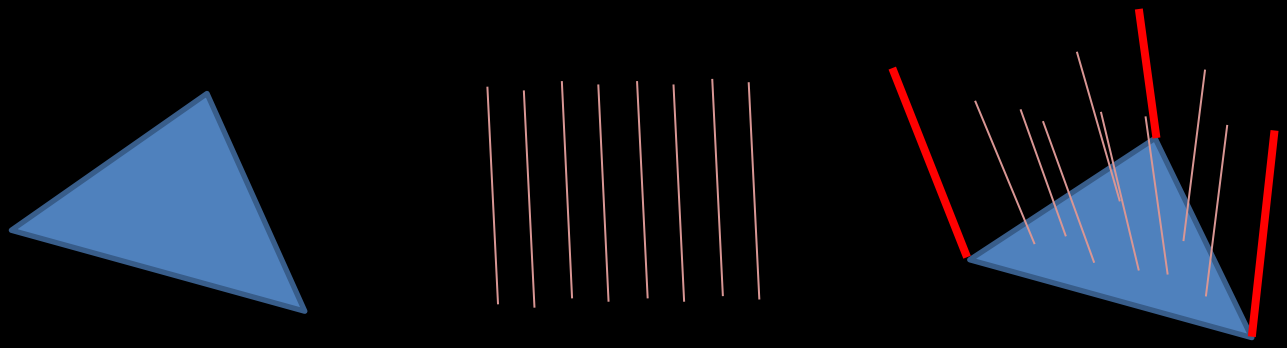
- 每一个‘长了毛’的三角形都会画到render target上
- Hull Shader (HS)决定了Tessellator在一个三角形上生成多少根毛发
- 每一个三角形顶点都会对应一个引导曲线(Guide Curve)
- 在Domain Shader (DS)里，我们对顶点位置和其它属性进行插值



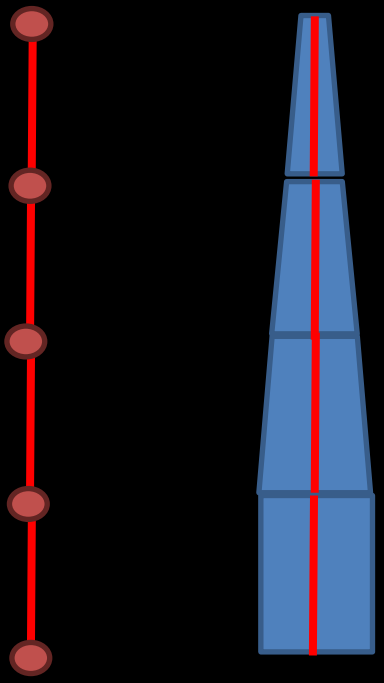
用Tessellation引擎插值生成毛发

- Hull Shader
  - 根据用户输入，如density map, LOD, 等，决定生成多少毛发
  - 决定每一根毛发有多少个线段
  - 输出是一系列isoline图元
- Domain Shader
  - 输入插值毛发的数据
  - 输出插值线段的位置信息和其它属性

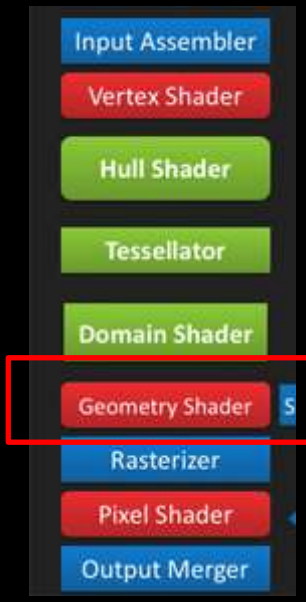




渲染毛发图元生成



- Domain Shader 输出一组线段
- Geometry Shader 把这些线段变成面向相机的多边形
  - 每一条线段 -> 一个四边形
  - 四边形的宽度是由输入的贴图决定的，从底部到尖部逐渐变窄



毛发插值的采样

- 为了属性的随机分布，我们生成了一个小的LUT表
  - 毛发根节点对应在毛发三角形里的重心坐标(Barycentric Coordinate)
  - 根据毛发ID来哈希查表
- 毛发形状控制(Shape Control)参数
  - 长度变化　Length variation
  - 波浪起伏　Waviness
  - 毛发成簇　Clumping
  - 毛发密度　Density
  - …

毛发渲染阶段

- 需要两遍渲染，一遍渲染阴影一遍最终效果渲染
- 每一遍渲染都要完整走过HS-DS-GS-PS管线
- 但是两遍渲染我们会使用不同的tessellation参数
  - 对于阴影渲染，可以使用更少的毛发
  - 最终效果渲染时做视觉裁剪（visibility culling）和LOD

阴影渲染 (2万根毛发，粗细 = 30)

最终效果渲染(20万根头发，粗细 = 3)

毛发渲染

- 阴影采样在DS中进行（逐顶点采样）以提高性能
- 其他的渲染都在Pixel Shader里进行
- Kajiya-Kay渲染主要的高光，同时做了假的次要高光(secondary highlight)处理
  - 一般而言，二次高光比较不明显，但是影响范围更大

只有主要高光的效果

加上次要高光的效果

毛发和皮肤的融合

- 两者使用相近的贴图→ 对于隐藏斑秃很有效
- Kajiya渲染模型对于Diffuse渲染效果不是很好
  - 收到光照时毛发和身体有显著不同
  - 把毛发的切线(Kajiya)和身体的法线混合起来做diffuse渲染

身体的Diffuse渲染

Diffuse Kajiya 渲染

Diffuse Lambert 渲染

性能分析

- 渲染是主要瓶颈所在
  - Tessellation（HS+DS)是最主要的瓶颈
- 模拟只占整个时间的10-20%
  - 较少的模拟的毛发节点数（如1万个点）vs 较多的渲染节点数(如 50万个点)
  - 对于长发会有所不同(更多的模拟计算，更少的tessellation)
- 渲染时的灵活性vs性能
  - 用户总希望给着色器更多的输入(毛发的：法线, 切线, 速度, …)
  - 对着色器的数据做了很多的压缩处理 (unit vector compression等)

目录

- 简介
- 美术制作
- 毛发运行库流程
- 游戏集成经验分享
- 未来的工作

游戏集成经验分享

- ‘Hair SDK’ 可以做的非常大而全，且有”侵略性”
- 大多数游戏引擎在渲染批次之间不会保存渲染状态(render state)
- 素材之间的缩放会很不一样(英尺, 厘米,..)
- 毛发部分的骨骼是整个骨骼模板的一部分，所以一般都要做骨骼的映射

目录

- 简介
- 美术制作
- 毛发运行库流程
- 游戏集成经验分享
- 未来的工作

未来的工作

- 更多的长发特性支持
  - 更好的长发模拟
  - 长发和身体的碰撞（胶囊体碰撞等）
  - 发丝间的相互影响
  - 渲染细节调节工具（Strand texture, Stray curves, volume control..)
- 适应各种引擎的需求
  - 延迟渲染的支持 (Deferred Shading)
  - 运动模糊 (Motion Blur)
  - 景深(Depth of Field)
  - 环境遮蔽(Ambient Occlusion)


let me share
2014
www.chinagdc.com.cn