
NVIDIA演示团队的奥秘
GeForce 6800
NVIDIA演示团队

美人鱼“Nalu”的制作过程

Hubert Nguyen
William Donnelly
NVIDIA 公司

金黄长发的渲染

金黄长发的渲染

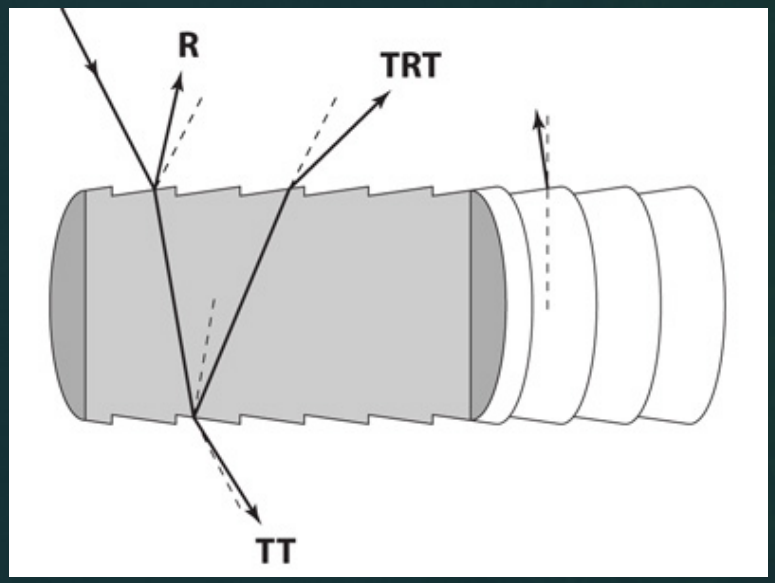
- 长发
  - 要求有流畅的动画效果
    - 因此光照效果不能固定
  - 要求有大量的光影变化
    - 因此着色处理必须很快
- 金发
  - 需要三个可见的高亮区，而黑色头发只需要一个就够了
  - 阴影更加明显

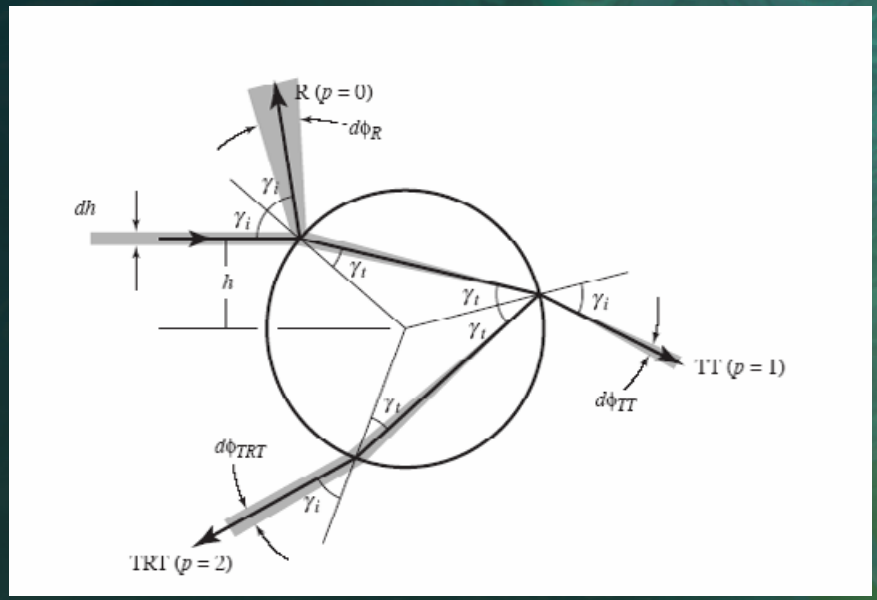
鸣谢

- “如同真人头发丝上散射出来的柔和光线”
- 制作者：Steve Marschner、Henrik Wann Jensen、Mike Cammarano、Steve Worley和Pat Hanrahan
- SIGGRAPH 2003

纸模型，
三个明显的高亮区

- 仅考虑3个最重要的条件
  - R, TT, TRT





- 使用路径符号
  - R表示反射
  - T表示传输

TT高亮

- TT – 强烈的前向散射元素
  - 对表现水下的头发效果非常重要

反射系数模型

- 头发模型是一个4维函数
  - 两个光线角度 + 两个眼睛视角
- 分解为低维表达式
  - ```
    M_R (thetaH) * N_R  (thetaD, phiD)
    + M_TT (thetaH) * N_TT (thetaD, phiD)
    + M_TRT(thetaH) * N_TRT(thetaD, phiD)
    ```
- 纹理编码使用了二维贴图
  - 纹理贴图要比大量计算要快
  - 使用mip贴图消除了“着色锯齿”现象
    ```

阴影效果

- 基于“不透明阴影贴图”（**OSM**）技术
- 制作者：**Tae-Yong Kim**和**Ulrich Neumann SIGGRAPH 2001**

为什么使用“不透明阴影贴图”（OSM）技术

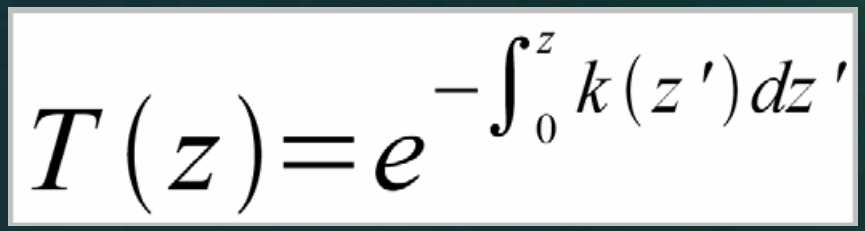
- “不透明阴影贴图”与“普通阴影贴图”的比较
  - “遮挡的光线百分比是多少？”
  - 相对于
  - “光线是否被遮挡？”
- 因此支持AA边线渲染和立体渲染
- 边线附近的正常阴影贴图锯齿
  - 头发全部为边线！

Kim与Neumann的测试结果

无阴影

15个切片

不透明度的计算公式



- **T(z): 渗透到深度Z的光线量**
- 对于离散图像（头发）：
  - 整体效果按照光照部分和阴影部分之间的头发丝之和来计算
- 通过相加混合来计算总和
  - “消光系数”K控制阴影的暗度

创建不透明贴图

- 在头发上选择16个切片位面
  - 根据头发的限定范围进行统一分配
- 相对于每个头发像素和每个位面而言
  - 头发像素是否比位面更靠近光源？
    - 是：添加头发到分摊部分（位面）
    - 否：不作任何处理

OSM创建

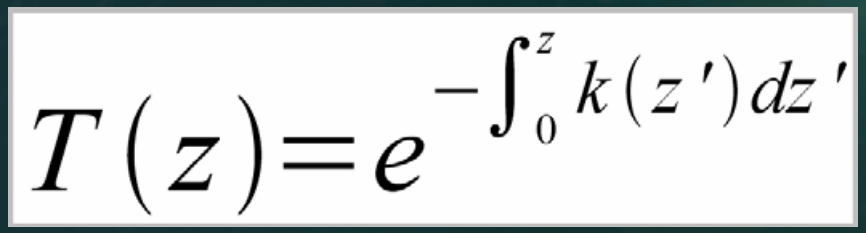
- 以16个切片渲染头发
  - 原始执行：16条渲染通道（RP）
- 可以使用较低的头发细节等级（LOD）

...达到16

顶点着色器的应用

- 计算（头发）片段的光照空间位置
  - 将z-偏量添加到反向限制的z-解析度中
- 光照空间的头发像素位置决定了：
  - 查看哪些不透明贴图(z)
  - 查看不透明贴图的哪些位置(x,y)

像素着色器的应用



我们知道每个平面上的整数值

- 通过线性内插法来计算中间值
- 内插值是一个平面值的线性组合

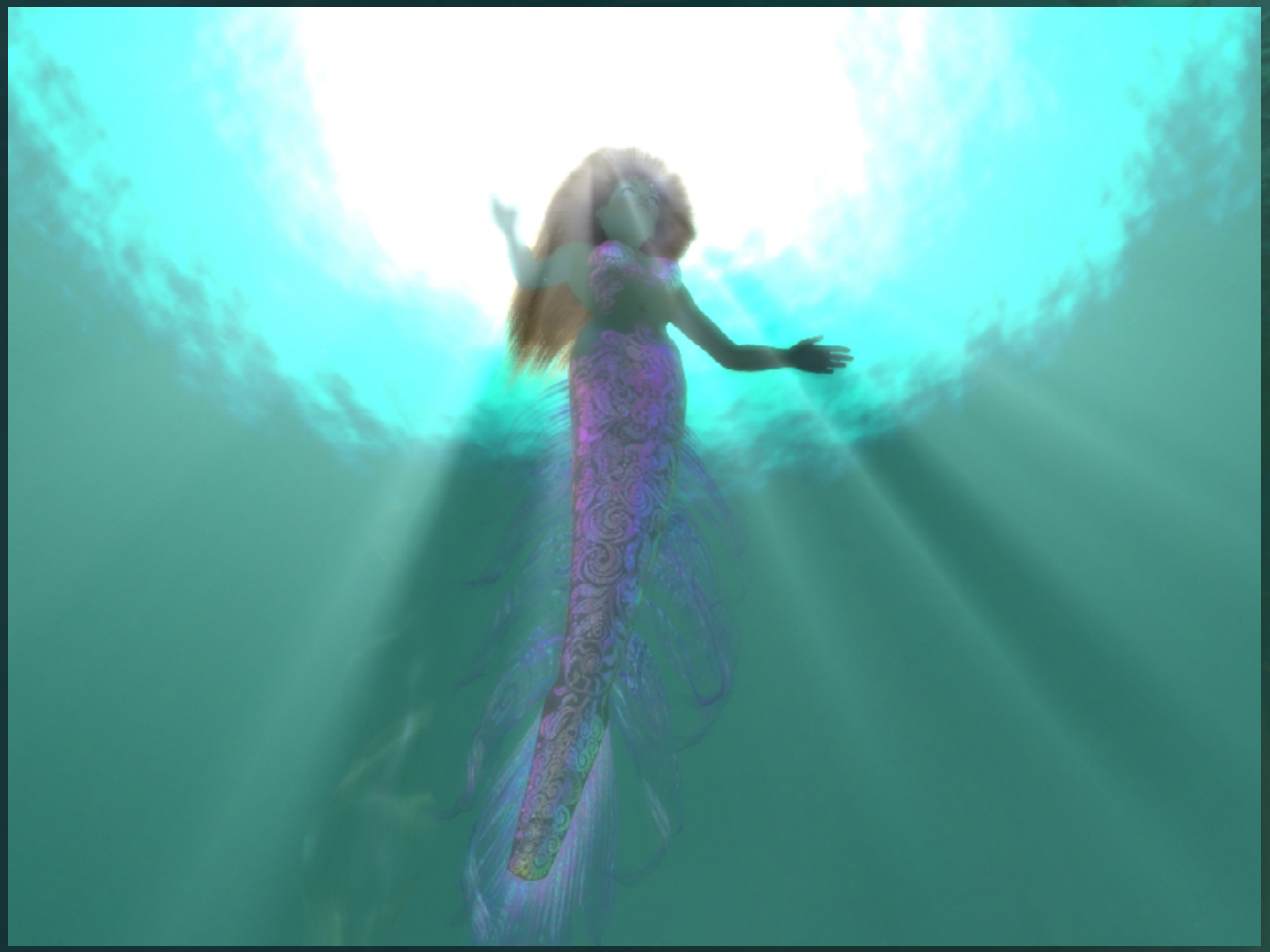
- 通过求幂计算不透明度

在演示中.....

无阴影的头发

有阴影的头发

光轴：“上帝之光”



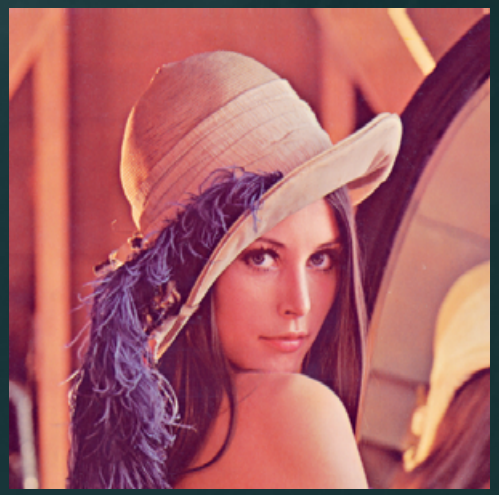
光轴基于光线模糊

辐射状模糊效果

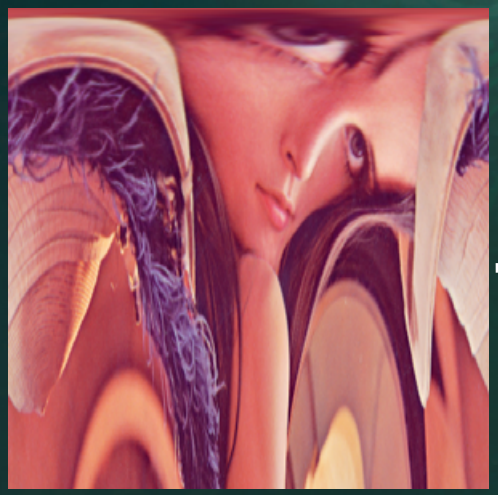
- 转换为极坐标(x,y)->(r,theta)
  - 使用网格线，位置 = (r,theta)，纹理坐标 = (x,y)
- 在辐射方向上模糊
- 转化回笛卡尔坐标
  - 使用同样的几何变形，但位置和纹理坐标被翻转了

从笛卡儿坐标到极坐标

辐射状模糊效果：视觉



从矩形的到
极性的

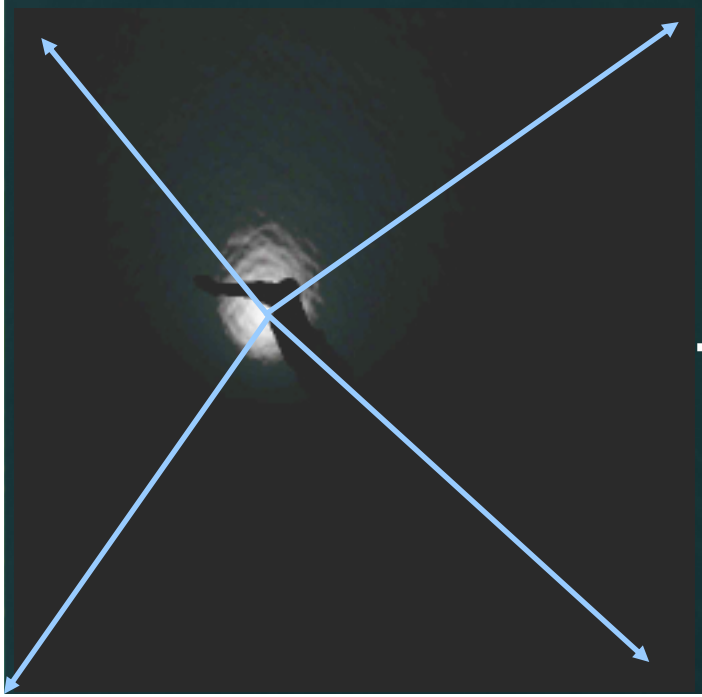


垂直模糊

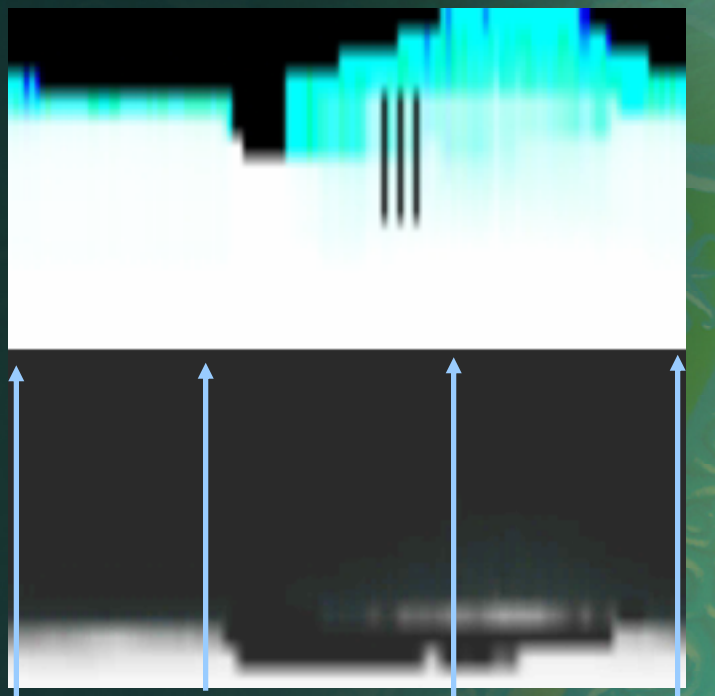
从极性的到
矩形的

辐射状模糊效果：在演示中

- 渲染明亮区域
- 在alpha通道中渲染阴影投射器
- 在辐射方向上模糊，减去遮蔽因素（从alpha通道内）



到极坐标



演示中的“上帝之光”

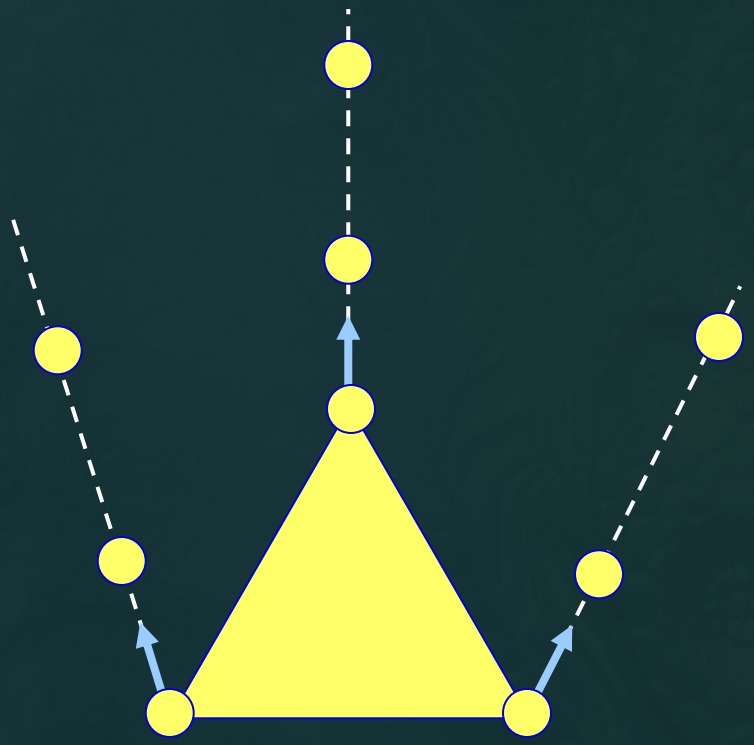
头发的几何与动态表现

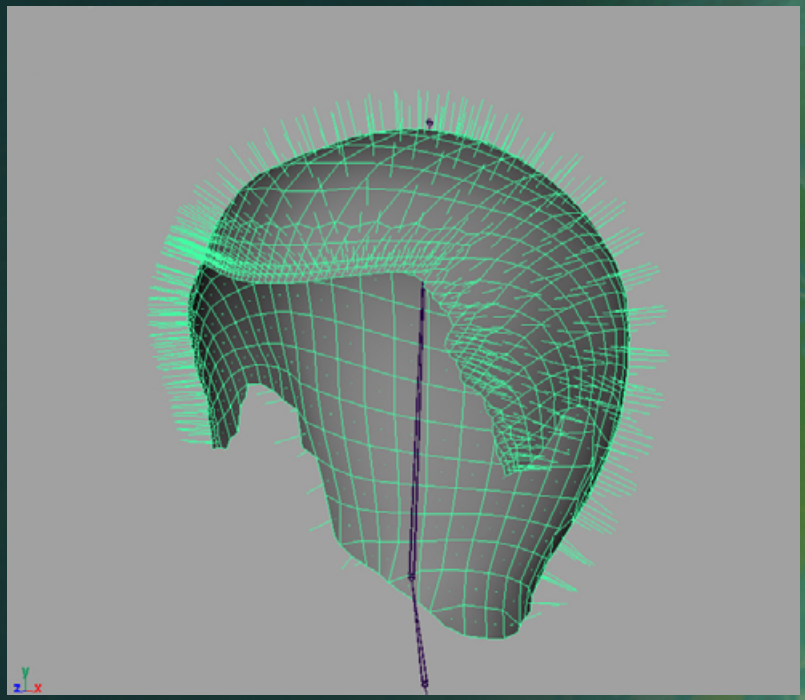
头发的几何表现：概要

- 762“控制头发”功能驱动4095根单独的头发
- “控制头发”功能
  - 真正有动力/碰撞驱动的头发组
  - 基于微粒体系，每个微粒之间由距离约束条件来连接
  - 通过参照系进行处理
- “精细头发”的几何坐标是通过柔化和修改“控制头发”来创建的

头发几何坐标：分布与增长

- “控制头发”通过专用几何坐标来增长



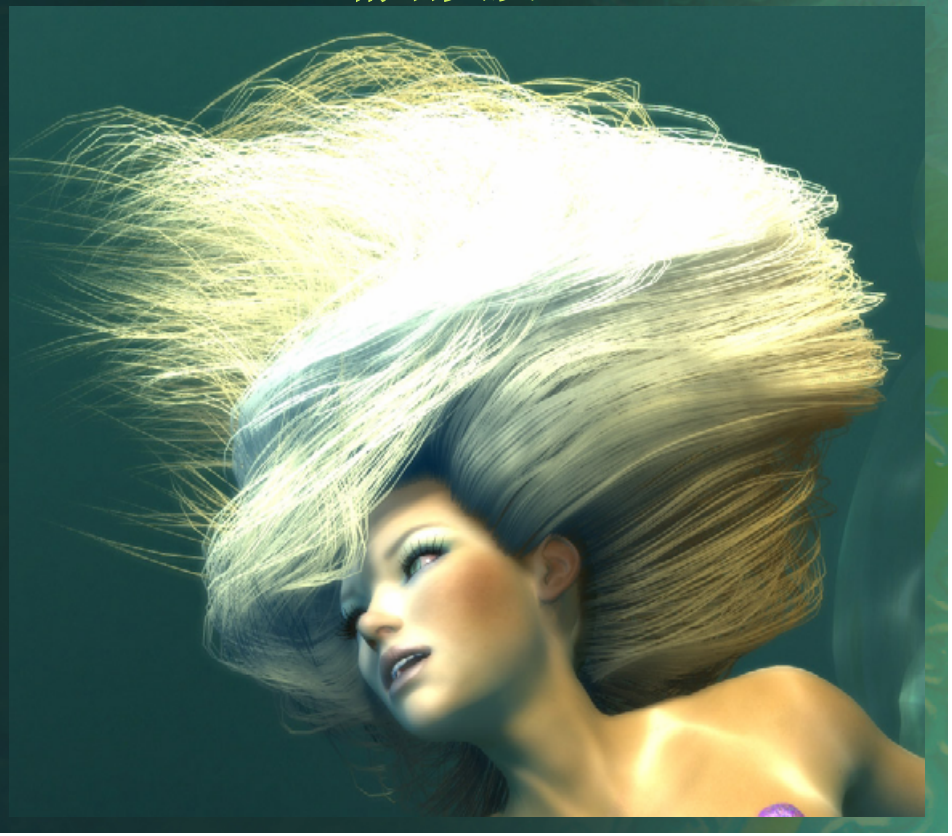


头发几何坐标：控制头发 （左图）

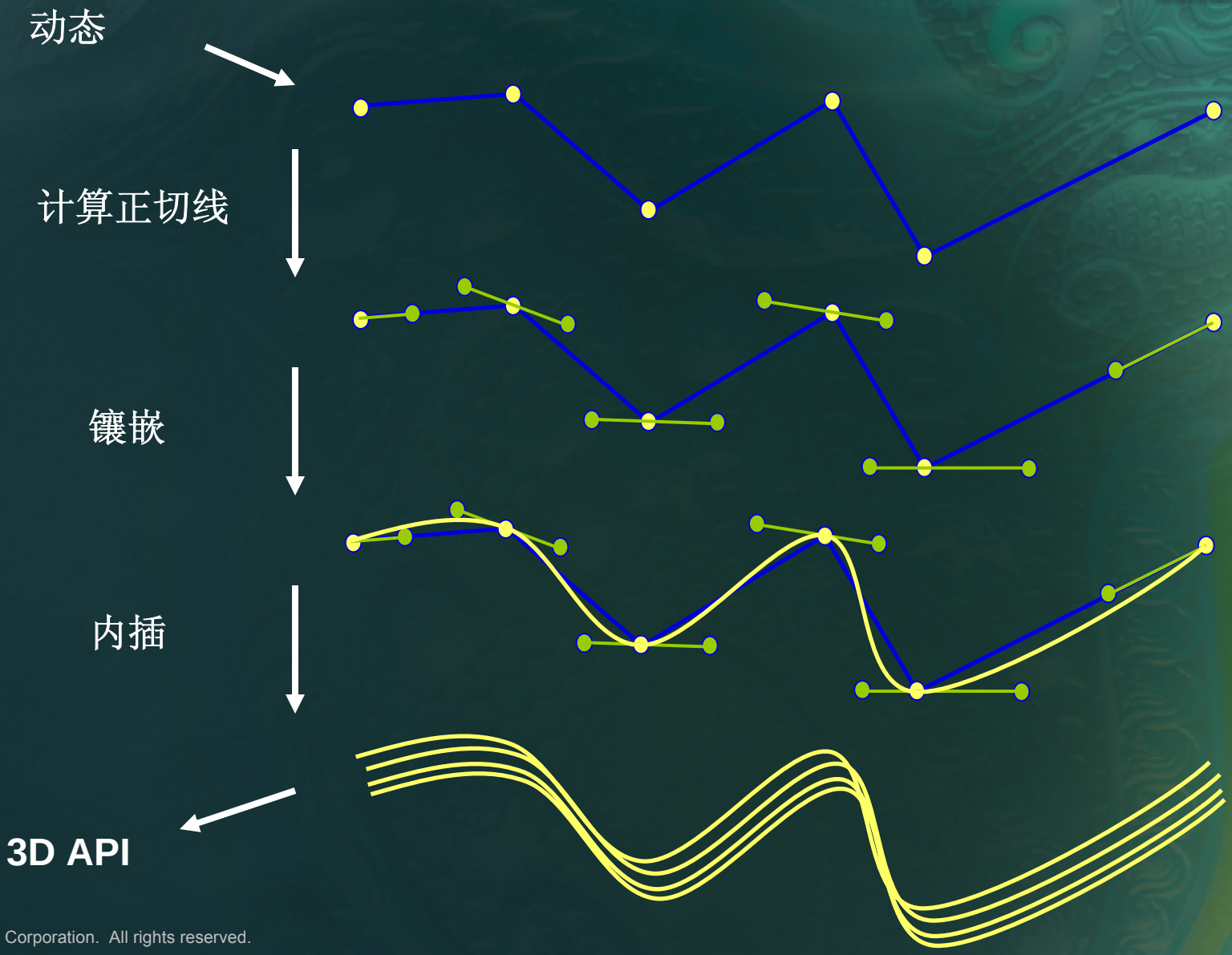
- 头发的物理性质/动态/碰撞是通过“控制头发”功能来表现的

控制头发

精细头发



头发几何坐标：生命周期（每帧）



头发的动态

- 基于微粒体系
- 使用“Verlet”整合
  - 利用上一帧位置来计算速率

$$\mathbf{x}' = 2\mathbf{x} - \mathbf{x}^* + \mathbf{a} \cdot \Delta t^2$$

$$\mathbf{x}^* = \mathbf{x}$$

参考：“高级的角色动作”

Thomas Jakobsen, IO Interactive, 丹麦

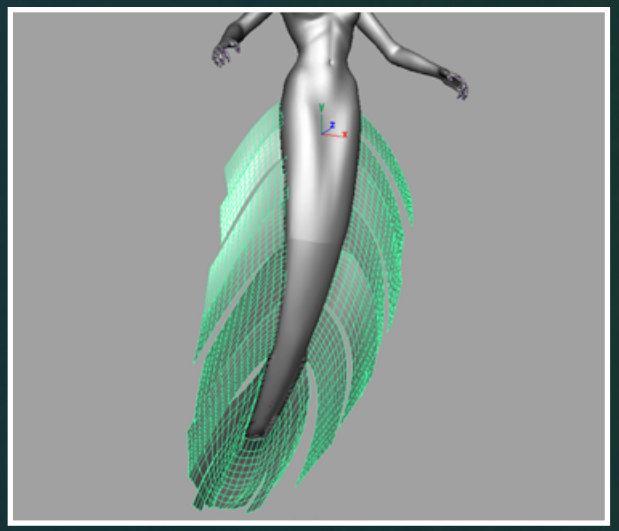
头发动态：约束条件

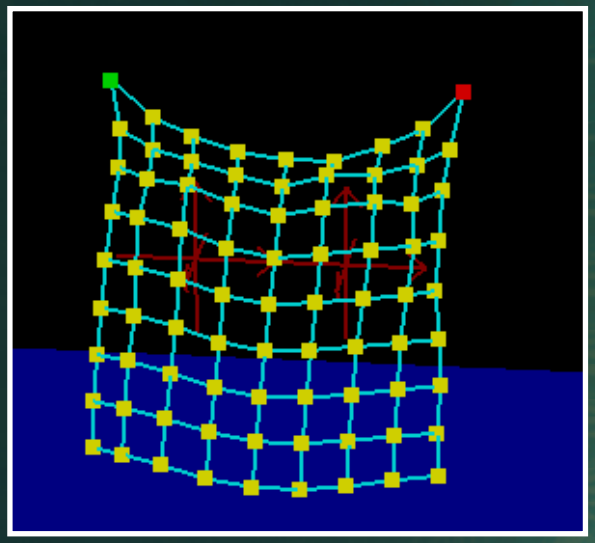
- 有两种约束条件：
  - 无限质量：“发根”部分应用无限质量，允许头部“拖动”头发。
  - 距离约束条件：规定“控制头发”片段的长度不变

- 如果我们反复使用这些约束条件，这些微粒将全部集中到所需位置

美人鱼鱼鳍

- 鱼鳍是织物的模拟物
  - 将三角边线作为约束条件，任何网状物都可以转化为织物





柔和阴影

- 基于
  “纹理空间漫射”参见“GPU几何”
- 进行定期的阴影映射计算
  - 但是在纹理空间内渲染
    - 将UV坐标作为顶点着色器输出位置
  - 模糊化纹理坐标黑白阴影效果
  - 渲染人物时，利用模糊阴影效果取代阴影比较

柔和阴影：可视化处理

UV 空间渲染

柔和阴影：挑战

- 在UV空间内展开人物
- 可见的接缝是UV不连续

未来工作

- 将更多工作转移到GPU上
  - 物理性质
  - 碰撞（截屏图 – Simon的织物效果演示）
  - 曲线镶嵌
  - 普通/切线计算
  - 头发修改
  - 所有想得到的工作，真的！☺

“Timbury”的制作过程

制作者：**Ryan Geiss**

NVIDIA股份有限公司

全名：**Timbury Entonin Mudgett**

出生：**1896**年，英国，克利夫兰

职业：昆虫学家

他的兴趣：搞科学研究

所用技术与效果

- 高动态范围（HDR）光影效果
  - 可进行高精度的光影效果计算
  - 自动获得控制：镜头对亮光做出反应
    - 观察亮光-> 普通物体变成逆光轮廓
    - “*普通*”的白色调（比如白色衣物）变得暗了，但是饱和白色（比如太阳）仍然很明亮。
- 环境照明采用FP16精度的环境反射
- 图像的“柔化”后处理
- 动画：皮肤和混合造型处理
- 适应性的细分表面
- “软”（多抽头）阴影
- 折射眼镜醋

帧渲染

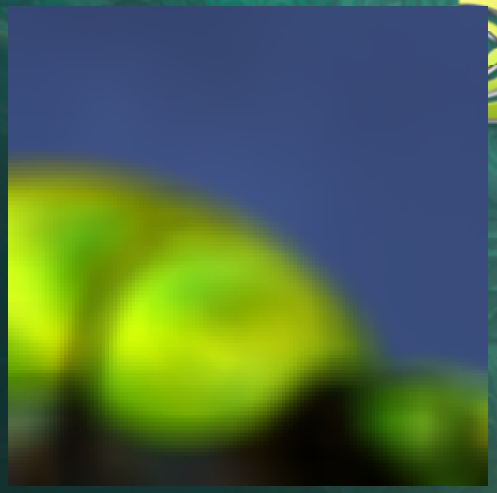
- 将场景渲染到纹理上
- 分析其亮度
- 创建副本，充分地模糊化
- 将清晰的图像和模糊的图像混合起来（以柔化图像）
- 使图像变暗（基于光照分析）；
- 在屏幕上绘制最终图像。

渲染（更多细节）

- 首先，将整个场景渲染为一个大的fp16精度的纹理
- 将其渲染为256x128纹理，每个（目的地）像素取样3x3源纹理元素
- 另外四个流程，都是256x128 fp16精度的纹理：
  - x模糊
  - y模糊
  - x模糊
  - y模糊
- （创建一个精细的近高斯模糊图像）（但是很快！）
- 进行最后的渲染流程时，将模糊图像与原来的高清晰度图像进行平均化处理，从而获得一个精细柔和的电影级效果。

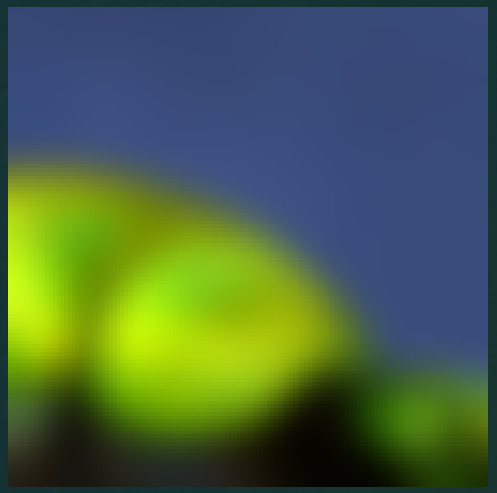
•原始

模糊/x

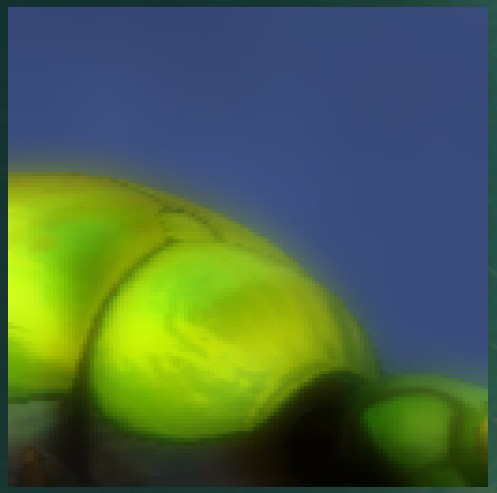


模糊/y

•模糊/x(2)



模糊/y(2)



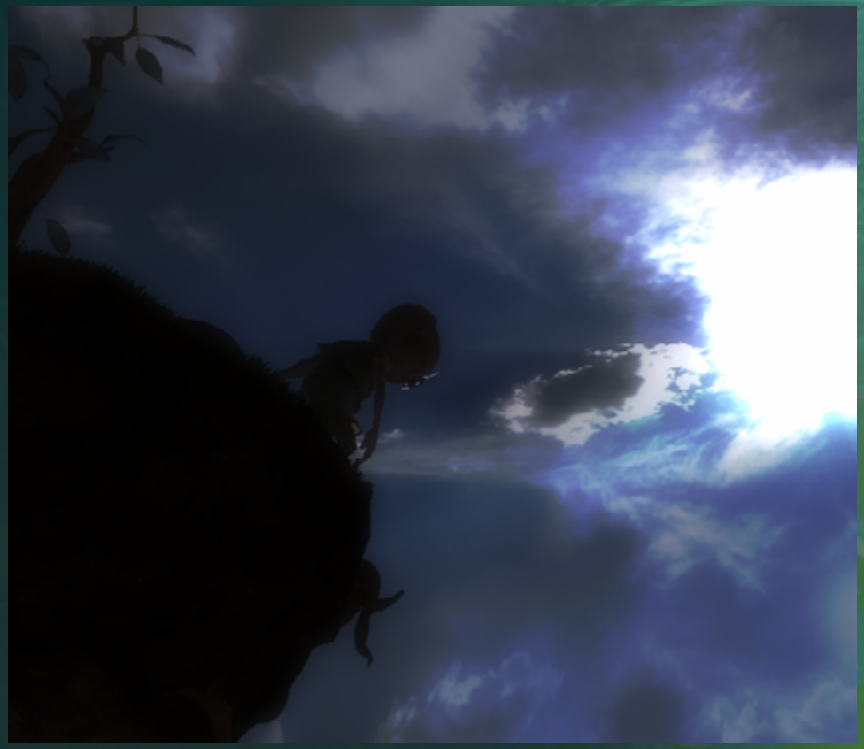
混合

自动获取控制（AGC）

- 模仿镜头或人眼面对强光时的反应：收缩入光孔
- 场景中真正的强光是太阳
  - 大约比场景中大多数其他颜色的亮度要高40倍
- 浏览场景，您能看到入光孔是如何对强光做出反应的。[演示]
- 直视太阳→入光孔收缩，使大部分物体变成逆光轮廓

AGC实例

正常光照



直视太阳

使用fp16 HDR纹理

- fp16源纹理包括：
  - 天空环境反射
  - 环境光照运用了所有的环境反射（1个漫射，8个镜面反射）
- fp16渲染对象包括：
  - 基本渲染对象
  - 所有的后处理渲染对象
- fp16源纹理以Industrial Light & Magic的**OpenEXR**文件格式存在磁盘里。
  - 免费源代码：**http://www.openexr.org**
- 优势
  - 高质量光影计算和后处理效果
  - 启用AGC效果

渲染流程：添加AGC

- 确定进入镜头的光线量
- 将[经过柔化处理过后的]模糊图像的采样缩减为8x8图像
- 再次缩减为1x1图像
  - 综合片段着色器以1像素运行
  - 取16个采样，带双线性过滤，运用所有64个源纹理元素
  - 靠近中心的采样量略大
  - 每个样例的亮度为：lum = dot(float3(0.3,0.48,0.22))
  - 这一帧的平均亮度水平为：L = sum(lum values) / sum(weights)
  - 写入浮点3（1/L, 1/L, 1/L）作为着色器的输出信息

渲染流程：添加AGC（续）

- 最终合成流程的着色器
  - 混合清晰图像和模糊图像（~50/50）
  - 在1x1纹理上从任何地方的样本上扩展该渲染结果（保持1/L）。

线性与非线性色调映射

- 我们的AGC执行过程可找出所有像素的平均光照度，然后通过其反面来扩展场景的光照度——就这么简单。
- 一种更高级的色调映射方式是计算照明值的log2值，然后通过一个特殊方式进行扩展：
  - AvgLumLog = $\sum \log_2$ luminance
  - AvgLum = $2^{\text{AvgLumLog}}$
  - 在最后流程里：颜色' = 灰色 * 色彩/（1 + 色彩
    - 灰色为"中度灰色"值(~0.5)。
  - [展示：利用 't' 来激活色调映射……]

*来自*数码图像图像的色调复制*——制作者：Erik Reinhard, Mike Stark, Peter Shirley和Jim Ferwerda。

记录总数与色调映射：

较好的亮度分布

线性总数和简单缩放：

较差的亮度分布

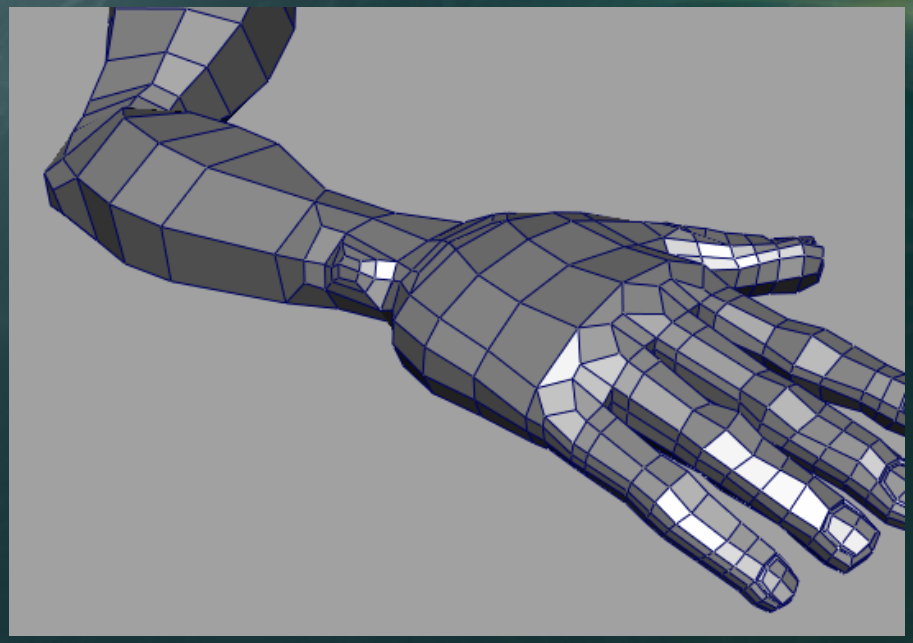
较差的对比度

较好的对比度

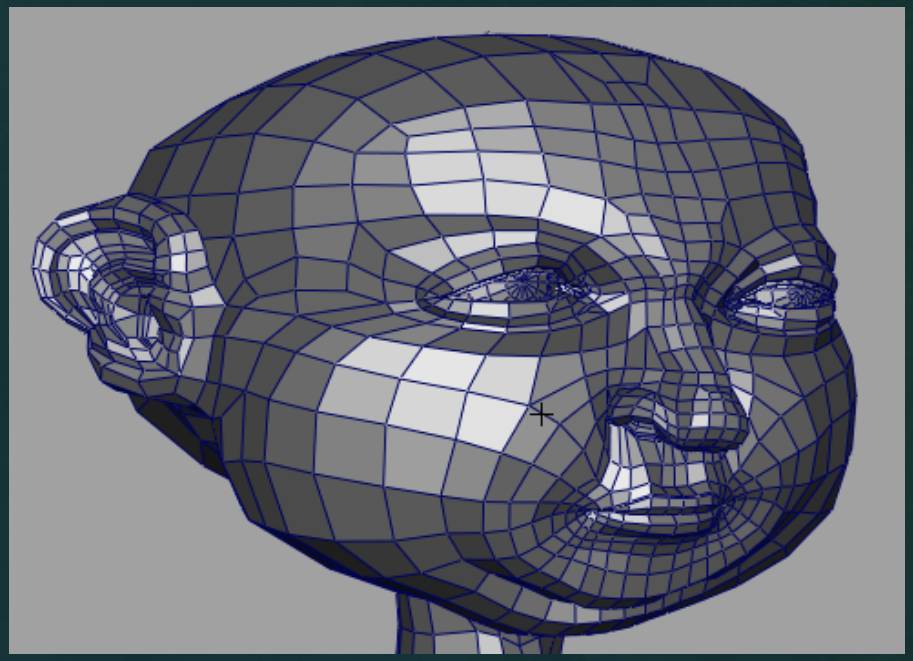
适应性细分表面

- 制作者：Michael Bunnell，Nvidia公司
  - 目标：适应性地细分多边形，以保持网格显得更加精细，使画面绘制效率更高。
  - 放大时添加多边形，就像肘弯一样（增加曲率）等。

  [……演示]
- 原始网格：一般为方形，不过三角形也可以
  - *应该为低解析度，但清晰程度应足以表现模特儿的重要特征（……请看下一个幻灯片）*
  - 开始时，将其完全转换为Catmull-Clark补丁（方形）。
- 目标：每个帧，适应性地细分多边形，直到屏幕空间“错误”数小于或等1.0像素。



手臂的原始控制网格



头部的原始控制网格

适应性细分表面（2）

- “错误”表示边线离理想的平滑表面有多远。
  - 特别地，错误是指场景中每个边线的中心与理想弯曲边线中心之间的距离。
  - 将该距离值投射到屏幕空间内——这就是错误的像素值。
- 如果方形的两条相对边有很高的错误值，可以沿着这些边线进行镶嵌。方形的另两条边同理。
  - 如果是圆柱体或手臂，可在最需要的方向上进行镶嵌。

链接

http://www.nzone.com/object/nzone_timbury_home.html

http://developer.nvidia.com/

http://www.openexr.org

通风口

通风口

海盗船“Clear Sailing”的制作

Joe Demers

NVIDIA公司

海洋模拟与渲染

- 我们的目标是创造出逼真的海洋景色，上面有起伏的波浪和行驶的船舶
- 我们需要：
  - 模拟起伏的波浪
  - 镶嵌海洋表面
  - 渲染海水（和泡沫）
  - 计算船在波浪上的动态表现
  - 渲染船激起的浪花

海洋模拟与渲染

海洋模拟

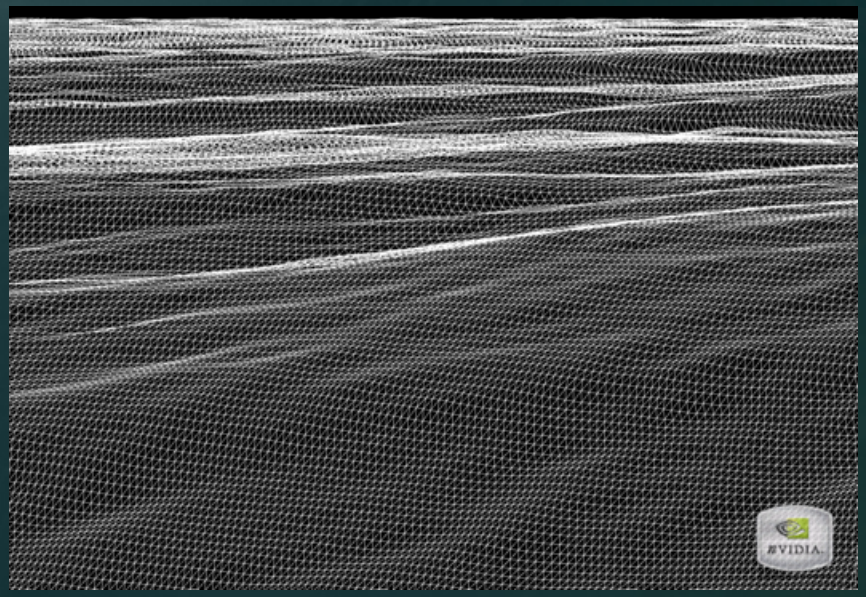
- 高端海洋模拟中最常用的两个模型是Gerstner海浪模型和基于FFT的统计模型
- 我们选择了Gerstner海浪模型，因为它更简单，没有周期性
- Gerstner海浪模型在海洋表面以平行于海浪的方向移动点，使海浪的尖端能够升高
- 我们发现，在150x150网格内使用45 Gerstner波浪能够为海盗船“Clear Sailing”演示带来最好的视觉效果。

海洋镶嵌

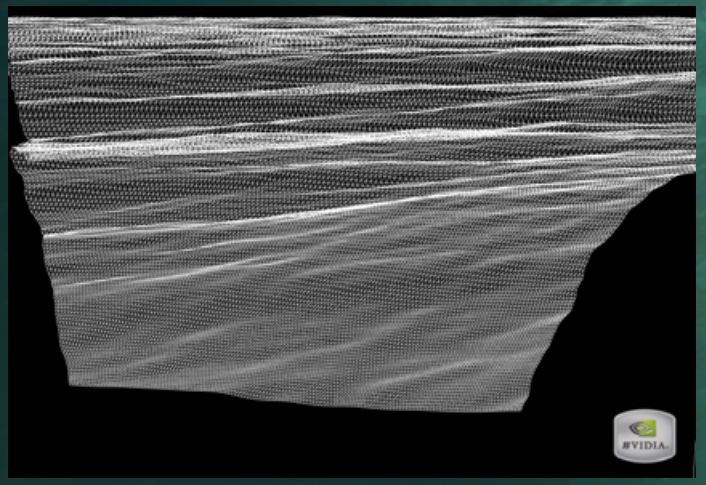
- 大部分以前的技术都是在场景内创建规则的顶点网格，然后填充这些网格，或者在网格上加上浓雾，或者同时使用两者
- 我们镶嵌眼部空间，将规则的网格映射到海平面与镜头视窗的交叉点
- 这样我们就可以仅模拟和渲染可见的地方，而且前景的镶嵌效果比背景的镶嵌效果更加精细

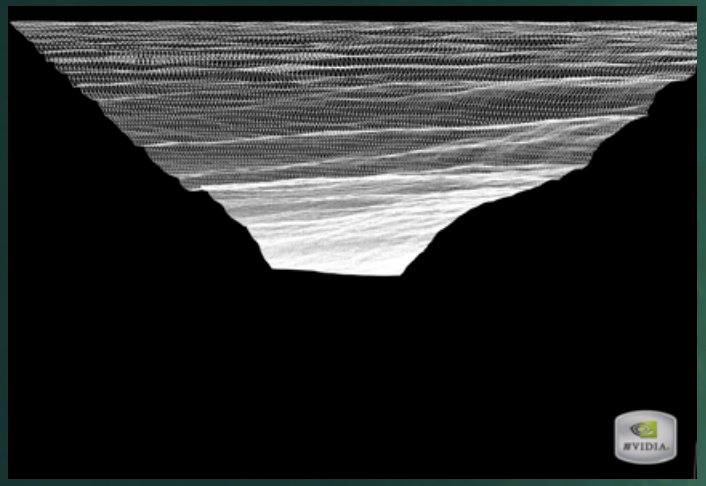
海洋镶嵌

满屏镶嵌



缩小来观看镶嵌





固定几何坐标，拉远镜头，我们可以看到实际绘制出来的几何坐标

海洋渲染

- 渲染深水需要多个光源效果
  - 水面反射的太阳光（方向性光源）
  - 水面反射的天空光线（环境纹理光）
  - 来自水面下的散射光（常量）
- 我们可以利用一个简单的菲涅耳函数来混合反射光和散射光
- 不过，菲涅耳指数虽然在平视海面时效果不错，但在垂直俯视海面时，容易使海面失色。所以，如果可能用到各种视角的话，低一些的指数的效果往往更好。

船的倒影

- 船用两种方式来遮蔽反射光
  - 使用z-缓存阴影来遮蔽太阳光
  - 通过将光线射入船的2D副本来遮蔽天空光
- 少量像素干扰会打破倒影，显得海浪频率更快

船的倒影

渲染泡沫

- 泡沫实际上就是水面上的一层半透明物
- 泡沫通常在海浪涌起的地方以及船尾航迹上的产生（即：泡沫层变得不透明）
- 我们使用渲染到纹理的方式和某些相加混合方式，使泡沫过一段时间自己消失

渲染泡沫

船的动态

- 船在波浪中起伏，但不影响海水
- 船上下起伏、倾斜、翻滚，但是不会在海平面上滑行或左右偏离
- 在水面上，船受到重力和风力的作用
- 在水面下，船受到摩擦力和海水浮力的作用
- 船的动态表现需要进行大量的调整/扭动等，不过，操作调整器也会给你带来许多乐趣
  - 如果你加快波浪的速度，降低动态计算（有效地缩放环境比率大小）的比率，船就会像一只玩具船一样移动

船的光照

- 船由4个光源照亮
  - 直接的阳光，通过z-缓存阴影功能来表现阴影
  - 天空光线，通过漫射和镜面反射表现
  - 周围环境的光线，天空固定的照明感
  - 反射光线，来自海水的微带蓝色的光线
- 漫射光贴图使船具有色彩
- 凹凸贴图和镜像贴图使船的轮廓更加清晰，使其比实际轮廓更具有几何学的复杂性

缆绳与风帆

- 风帆为两面体，当太阳在其背面时会微微发光
- 当水平面与太阳垂直时，向阳面和背阳面的着色器必须相符，否则你会看到接缝。
- 使用线条来画出缆绳，线条应根据观看者的距离而改变粗细
- 当线条变细到1个像素以下时，你可以通过降低透明度来获得理想的抗锯齿效果
- 使用alpha-to-coverage功能表示你不需要分类！

缆绳与风帆

弹着点与喷溅

- 弹着点
  - 当炮弹打到海中时，会溅出水花
  - 使用为数众多的水滴纹理来渲染大的水花
- 喷溅
  - 当船肋（或龙骨）移动到海面下时，在这些龙骨之间会溅出水花
  - 更快的动作使水花喷溅得更高更远

弹着点与喷溅

烟和碎片

- 烟
  - 当船大炮开火时，沿着开火路线喷出较大的带动态纹理（使用3D纹理）的火花
  - 烟的粒子开始飞快地移动，但速度迅速降下来，慢慢升高，消失
  - 烟的粒子下面部分显黑色，就像被太阳照亮一样——虽然简单，但看起来不错
- 碎片
  - 当大炮击中船舶时，产生许多小的三角形
  - 使用单一（但是很快）的动作移动并翻滚这些小三角形

烟和碎片

后期处理

- HDR风格的发光效果
  - 使用简单的8位单通道HDR效果
- 淡淡的雾气使水面、天空和船舶与整个场景融为一体。

后期处理

问题？

参考文献

- Jerry Tessendorf，“模拟海水”
  SIGGRAPH 2001 Course notes
  - course notes no longer online, here are the slides:
  - http://probe.ocn.fit.edu/slides2001.pdf
- Hinsinger, D.， Neyret, F.， 和Cani, M.P. “海浪的互动动画”，2002年电脑动画研讨会
  - http://www-imagis.imag.fr/Publications/2002/HNC02/index.gb.html