影院特效II:
复仇烈火

Kevin Bjorke, NVIDIA公司
2004年9月

概要

- 电影与游戏：
  相异之处与相似之处
  - 视觉品质与 “视效开发”
  - 制作规模
- 来自影院的灵感得到实现
  - 新的工具、着色器、灵感
  - 鲜活的实例
  - 融入游戏引擎
  - 融入艺术管道
- 源代码!
  - 所有实例的源代码都可以在
    http://developer.nvidia.com上找到。

“MRT” visualization
of texture coordinates

“复仇烈火”？？？

- *迄今为止……**
  - 可编程着色处理是当前功能最强大的游戏美工工具，它可以使游戏画面呈现影院级精美效果
  - 但这是一项艰苦的工作
    - 难以实施和实验
    - 难以嵌入游戏引擎
    - 更难纠错
- 回放时间。

“Thad” from Animatrix – Character © Silver Pictures

* Part I available at http://developer.nvidia.com/

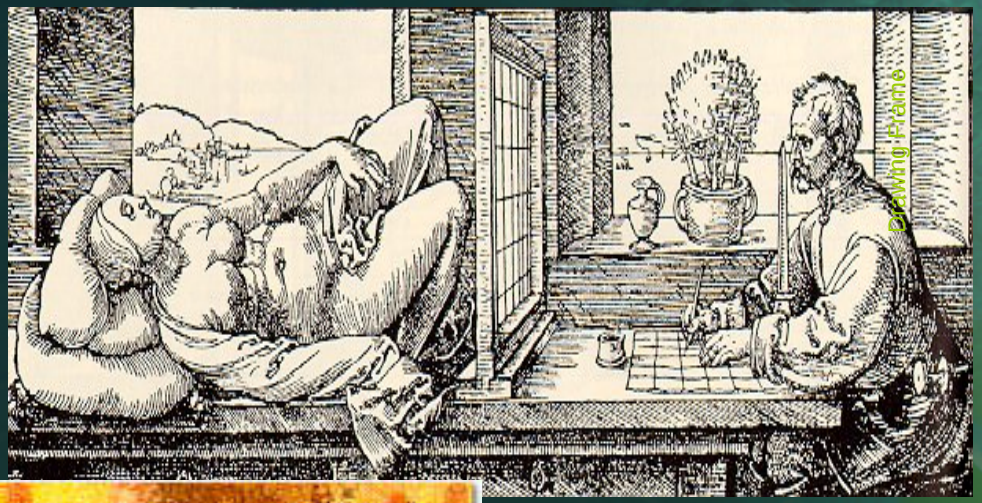
视觉艺术与游戏

- 两者总是密不可分
- 世界上最古老的游戏艺术可能是 “High Score”!

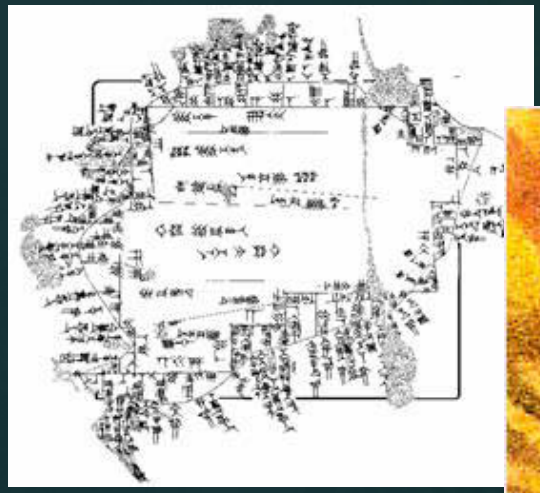
Chauvet Cave – maybe 25,000 B.C.

几何图形与光照

- 电脑图形是最新的科技发展
- 影院与图片
- 光学与几何图形
- “规则的图像”



Albrecht Durer



Mesopotamian Survey Map, ca. 2500 B.C.

Chauvet Cave – maybe 25,000 B.C.

电影与现实主义

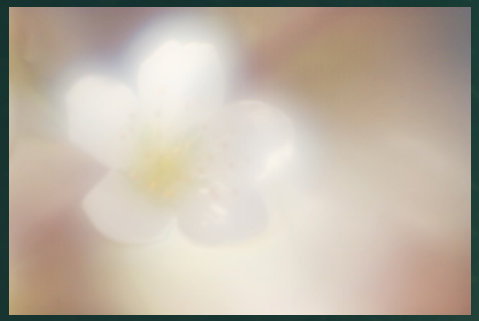
- 电影不是纪录片
- 电影是生动形象、风格鲜明的展示艺术
  - 属于主观创作的范畴，而非客观存在的现实
  - “它要高于生活”
- “纪录片风格”仅仅是*一种风格*而已
  - 所谓 “现实电视” 实际上是偷偷编排好的
  - 英国广播公司（BBC）的 “办公室” 节目明显是事先编排好的

Sylvia ©2003 Focus Features

CGI与“照相现实主义”

- “照相现实主义”只是另一种风格
- 照片可以高度抽象！
- 大自然包罗万象，丰富多彩，仅凭几个方程式根本无法重现
- 电影从早期媒体借鉴了很多图像制作灵感

Vogue cover, 1950 - Penn



抽象与精确的结合

- 贴图、透视图（甚至文献）都源于....会计学！
- 准确的描述固然非常重要，但却比不上“所有权”和“税务”这类抽象事务。
- “艺术省略”的重要性早已为人们所知。

Mesopotamian Survey Map, ca. 2500 B.C.

CGI，电影与绘画

- 电影从绘画等媒体中借鉴了光照和构图的概念
- 光照可以吸引注意力
- 光照可以确定情绪基调

Scott's *Blade Runner*

Rafael's *Transfiguration*

视觉效果开发

- 在着手某项工程（或部分工程）时，我们会确定哪些东西重要，哪些东西不重要，并确定构成风格的要素，这就是所谓的“视觉效果开发”。
- 在开发过程中，“视觉效果”确定得越早，效果越好（使用起来也更便宜）。

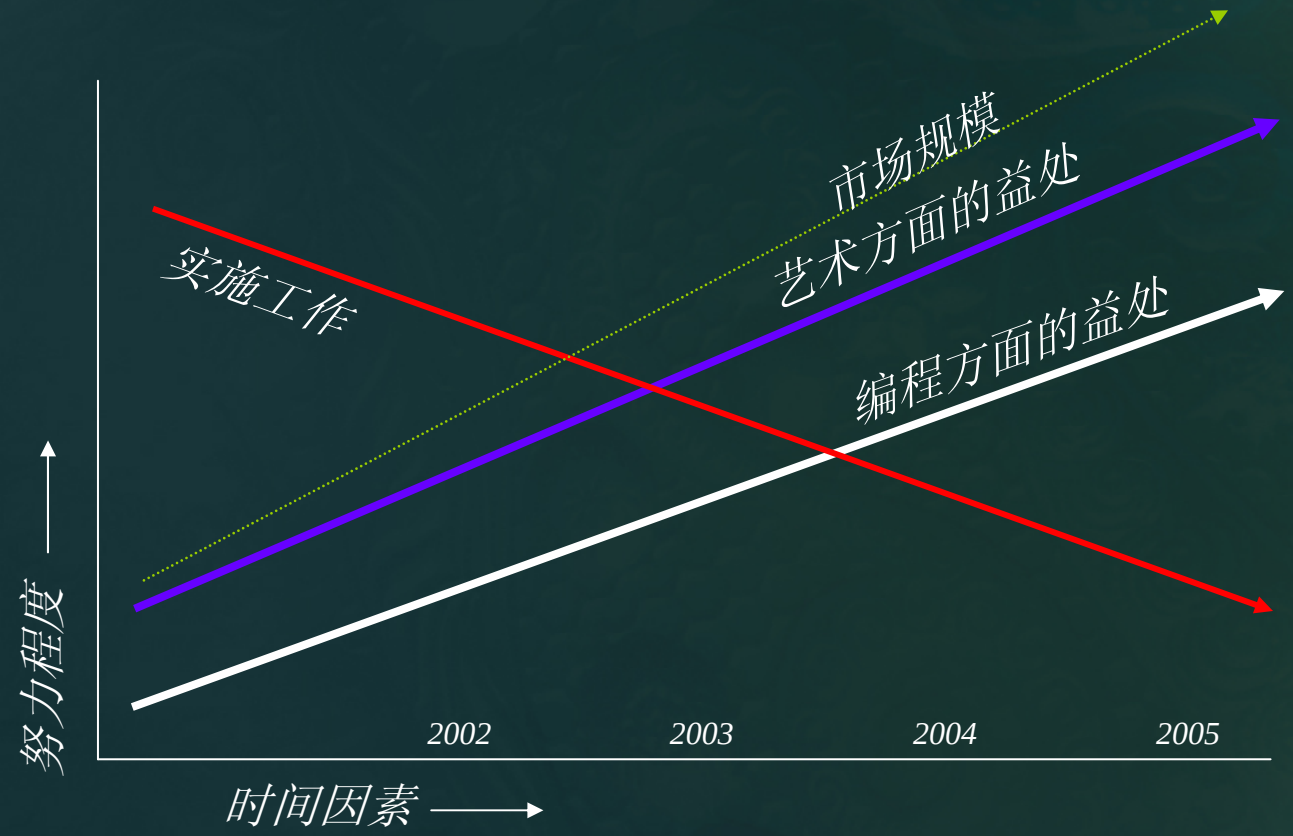
游戏中的视觉效果开发

- 在游戏中，视觉效果通常是引擎设计的副产品
  - 除了“最低公分母”之外，艺术家可能很难猜测任何其他事情了
  - 设计往往趋于保守和稳妥
  - 常常需要考虑技术条件的限制
- 在电影中，开始开发视觉效果时，很少过分关注像“预算”这样的“细节”。
  - 艺术家完全自由
  - 只需考虑故事情节

Test from *Animatrix* – © Silver Pictures

可编程着色：何时进行？

- 在工作流程中引入可编程着色，有利有弊
- 每个工作室都有其自己的“临界”点



开发着色器：编程人员

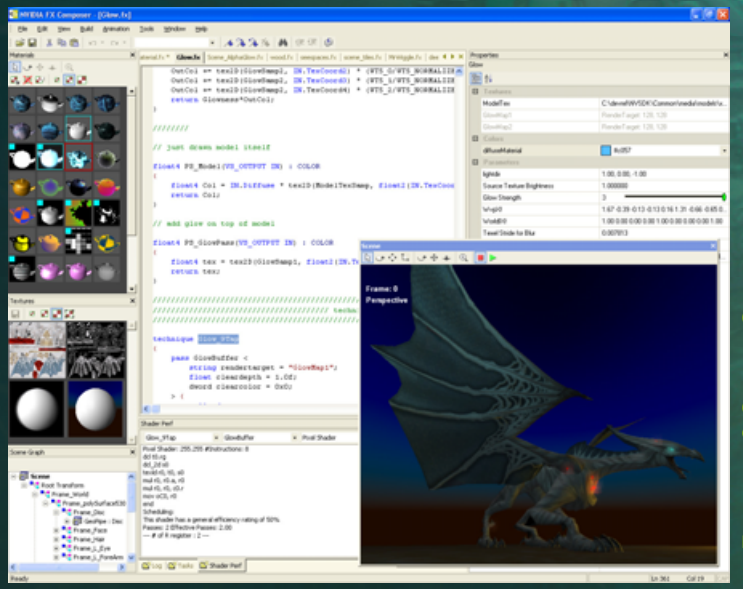
- 着色工具对编程人员和设计人员都很重要
- 现代游戏引擎要具备完善的功能，其工具必须支持下面的概念：
  - 渲染至纹理（RTT）
  - 多对象渲染（MRT）
  - 可以渲染像模版、alpha混合之类的状态等
  - 自定义纹理贴图（例如：标准立方体、杂色等）
  - 管理细节，确保复杂命令与任何具体游戏引擎的渲染循环相匹配
  - 可编写脚本
- 在各个制作阶段，如何在游戏引擎中存取结果？

开发着色器：艺术家

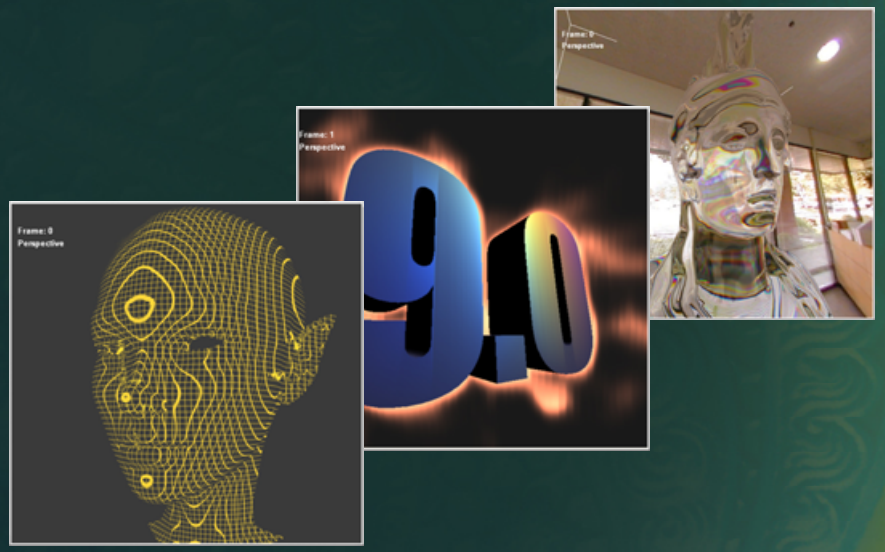
- 艺术家需要随时查看设计对象，而不仅仅是猜测以后的视觉效果
- 他们不仅需要利用正确的模型，还需要利用正确的光照环境，*这样开发出来的着色器和模型才能真正应用到游戏中去。*
- 通常情况下，不同的DCC应用程序（Maya、Max、XSI、以及……），渲染执行也不同——都不合乎实际游戏的需要。
- 我们也想为控制台游戏设计者提供方便，使他们能够查看同一游戏不同版本（DX9、DX8、Xbox、PS……）的模型。

HLSL和FX编辑器

- 专为这个任务制作的工具
- 结合不同着色器
- 自定义着色器
- 无需重写、额外的（软件开发工具包）SDK和运行时间层，就可以前后移动
- 表现优化工具
- C# 与 VB脚本编写
- http://www.fxcomposer.com/



Everquest® Content Courtesy
Sony Online Entertainment Inc.

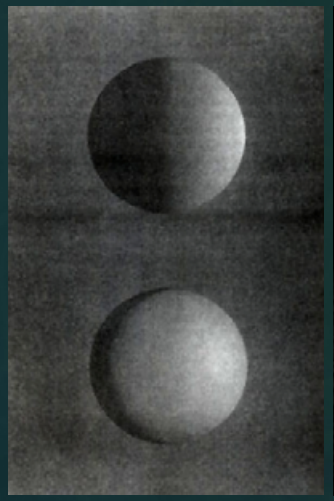


着色草稿本

- FX编辑器为艺术家和编程人员提供了一个有利环境，使他们可以实现复杂的构思，而无需使用C++编写整个游戏引擎！

建造着色器库

- FX编辑器搭载了许多样例着色器
  - 任何HLSL FX着色器都可以通过其他的着色器工具来使用
- 做试验、保存试验成果——总有一天您会用到它们!
- 保存、交换和收集这些着色器



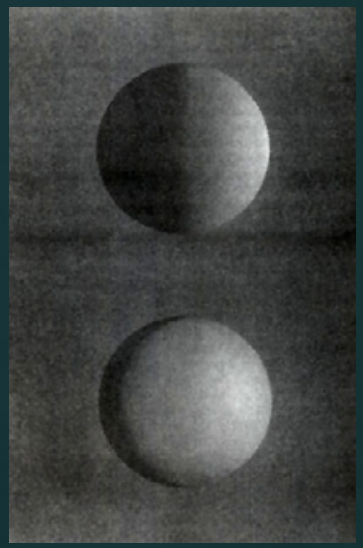
Ruskin's Shading Exercises, 1877

Bjorke's Dumb Mistake, 2003

草稿本实例：
将铅笔草图转化为着色器

- 着色后的场景很容易转化为着色器
- 作为色彩参考非常有用
- 注意微小细节（比如JPEG杂色），否则可能破坏效果

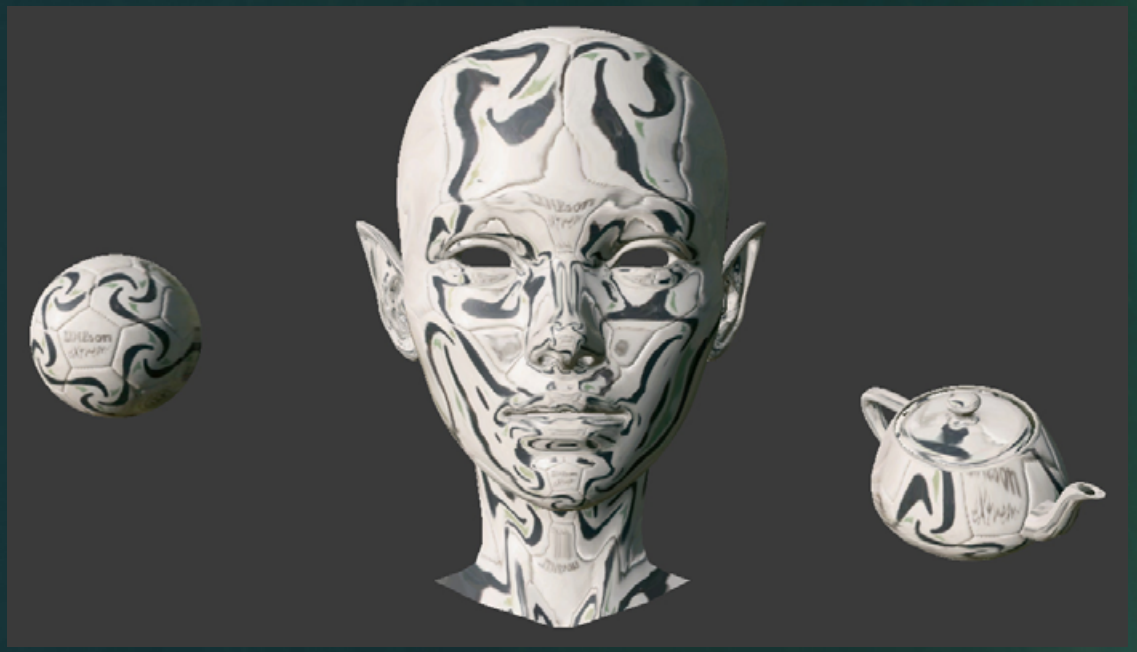
1877



2004

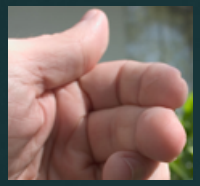
精彩实例

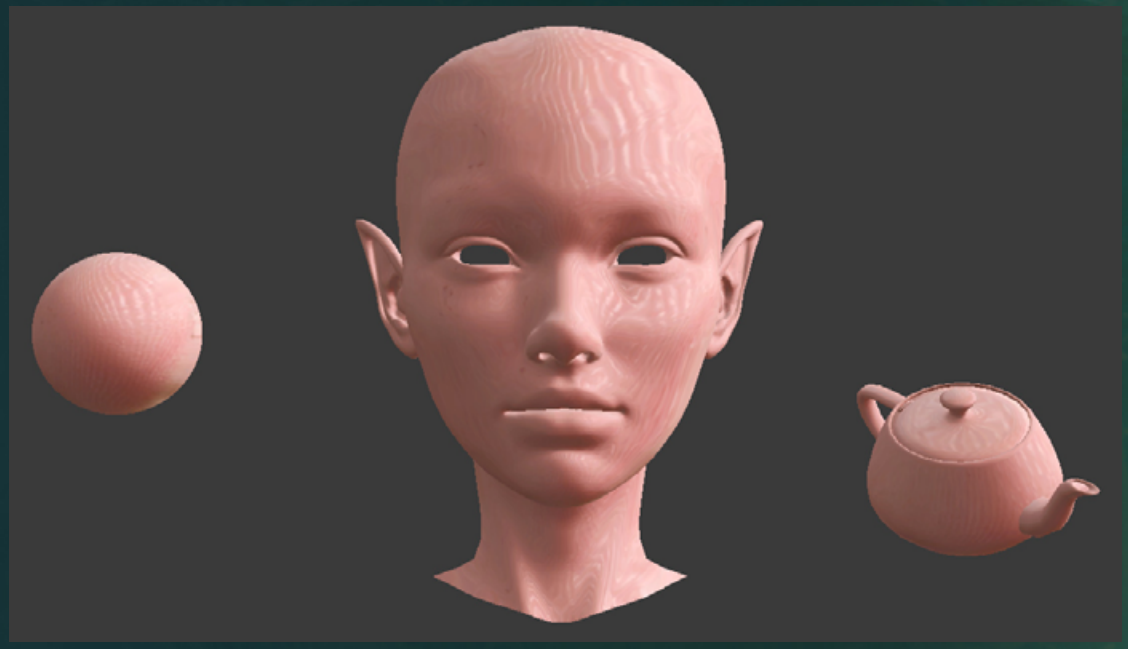
- 照片会扭曲变形
- 虽然可能很少有用到的时候，但至少很*便宜*——仅有一个周期！
- 我们能否对此作些什么呢？



形状是否必须为球体？

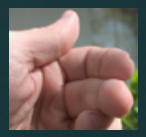
- 并不一定，如果我们可以用Photoshop来编辑一下的话
  - 我喜欢 “淡化” 和 “模糊” / “橡皮图章” 工具

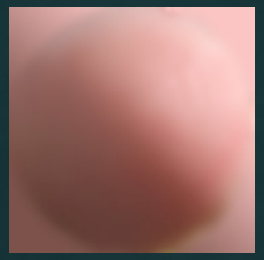




调整颜色

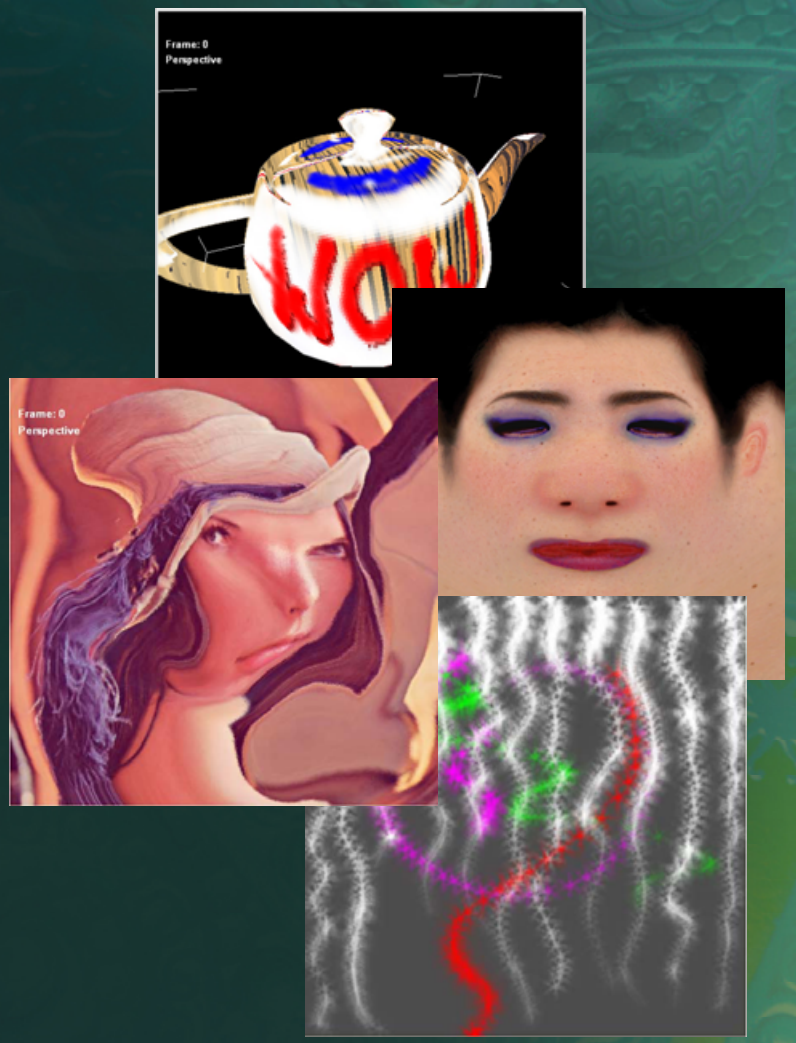
- 纹理进行高斯模糊化，隔离颜色
- 与其他着色模型混合使用，效果极佳





在FX编辑器内绘制草图

- 当我们绘制草图时：
- FX编辑器可以截取鼠标事件
- 我们可以利用这一功能，完全从FX着色器上编写微型应用程序



电影：管理规模

- 电影场面规模宏大
- 大量几何图形
- *大量着色器*
  - *玩具故事：*1300个着色器
  - *错误：*双倍
  - *Monsters公司：*“数以千计”
- 大量合成层
  （有时候数以百计）
- 脚本工具（Perl、C#、VB、Python、Mel… 所有想得到的）
- *长时间表*
  - 即时渲染缩短了时间……

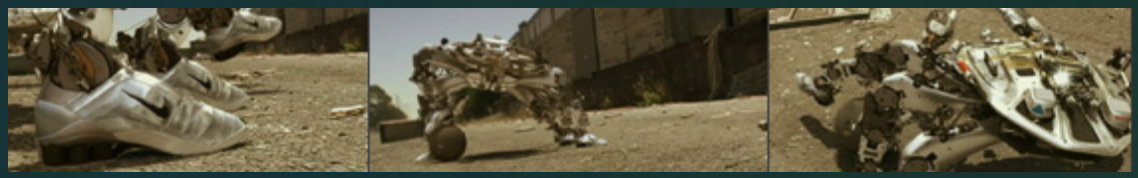
FX编辑器：管理规模

- 游戏规模也在扩展
- 管理大量着色器和模型是个繁杂的工作
- FX编辑器使用.NET组件，所以.NET可控制FX编辑器来迅速建造场景、导出图像、指定着色器、导出数据等。
- 使用C# 或Visual Basic

"See all the shaders in a directory"
-- Scene Generated by C# Script

长时间表

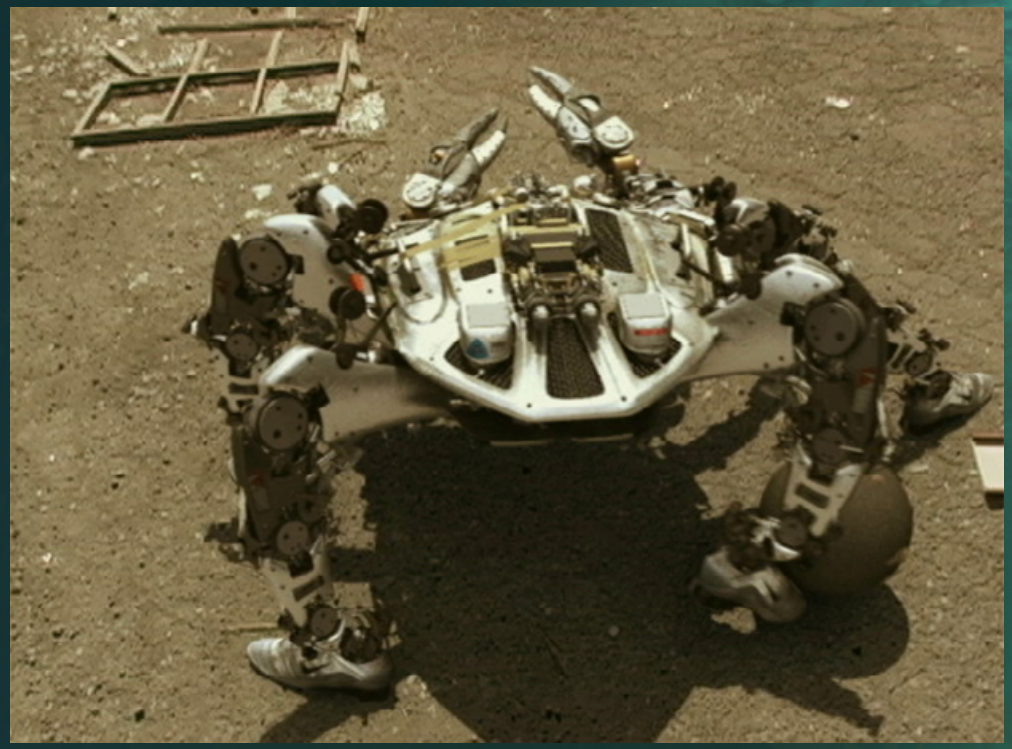
- 电影预算大，时间长，所以有可能开发出非常精彩的特效
- 但是：这些特效必须及早就确定好，这样电影镜头才能前后一致
- 有时候，这样做会限制创新
- 最快的创新作品：*电视广告*



Nike.Com campaign, Weiden + Kennedy, Dir Neill Blomkamp http://www.theembassyvfx.com/

NikeLab.COM

- 出品
  大使馆视觉特效
- 四周内完成!
- 利用Lightwave（光波）、Shake（振动）和NVIDIA Quadro图形芯片



Nike campaign, Weiden + Kennedy, Dir Neill Blomkamp
http://www.theembassyvfx.com/

阴影

- 阴影常常比照明更为重要
- 一旦产生阴影，就很难消除！

1998

2004

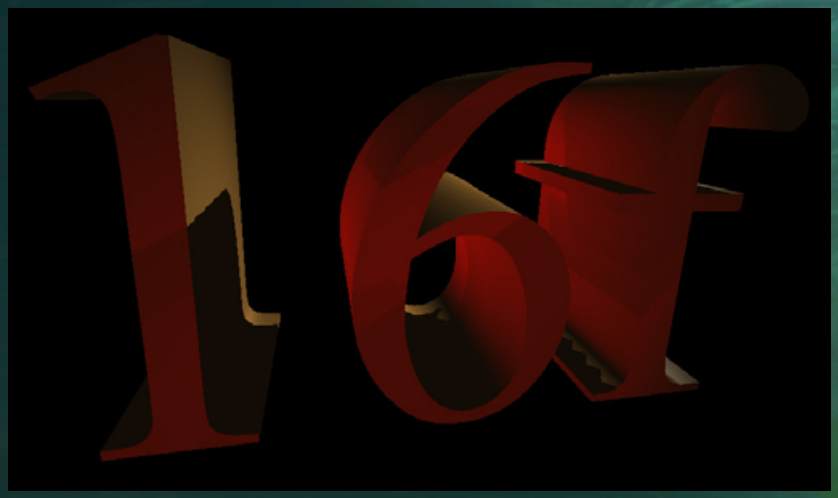
艺术课程：阴影

The Art Lesson

- 简单阴影：模版量或渲染至纹理
- 光线在什么地方?
- 共享光线
- “高级着色”课程将解决这类问题

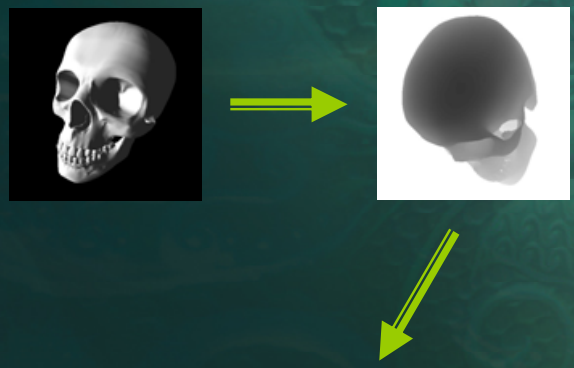
精美的阴影特效——半透明

- 阴影Z值也可以应用在物体背后
- “高级着色”课程将深入探讨这一技术（及更多的高级技术）



DXSAS – 可编写脚本的FX/HLSL

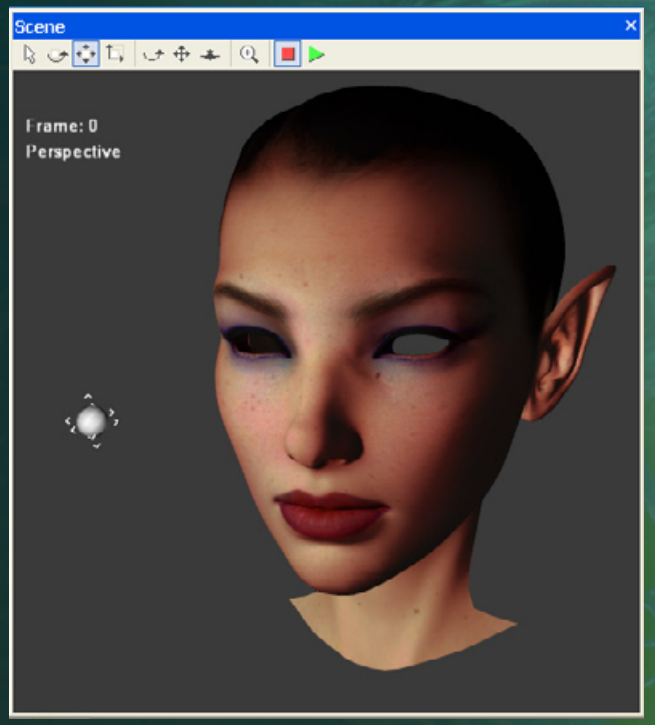
- DXSAS = “DirectX 标准注释与语义 ”是Microsoft公司的一项技术标准 * XNA的一部分
- 包括每个流程和技术的 “脚本” 语义
- 脚本定义了渲染对象，可以循环，还能互相调用
- HLSL “虚拟机”（VM）像矩阵数学一样处理数字



The Art Lesson

皮肤与着色

- 以便宜的漫射方式来表现表层以下的散射：
  - 通过将漫射着色计算中的(N·L)重新映射至 “((N·L)+w)/(1+w)”，我们能够将光线围绕在对象物体轮廓的周围
  - （不要担心数学细节——我们有实例！）
  - 因为这些都是漫射光线，所以可以在顶点着色器内完成



皮肤和直接反射

- 人越年轻，死皮越少
- 充满活力的皮肤细胞能够像小猫眼睛一样反光
- 因此，平滑的皮肤基调 = 年轻的外表
- Oren-Nayar 着色（昂贵）与 “纯灰色” 着色（便宜！）
- 结合灵感与创意

One Modern Variation

Traditional Grisaille Relief

光照

- 为光照部分着色——而不是为没有照到的地方着色
- 使用PS_3_0早期产品
  - 益处：使用“if”语句也有助于缩减批处理量
  - 编写一个着色器，面向ps_3 or ps_2进行编译
- 对于延迟着色，仅对光照到的像素进行着色
- 对于浮点像素而言，“黑效（Gloominance）”在各种情况下都是安全选择

Spotlight

智能光效

- Magy Seif El-Nasr 的“ELE” :极富表现力的光照引擎
- http://ist.psu.edu/SeifElNasr/
- 使用机器人技术中的负载平衡方程式来加大有限光束的可见度和表现“情绪”。

Mirage, El-Nasr et al, CIRA

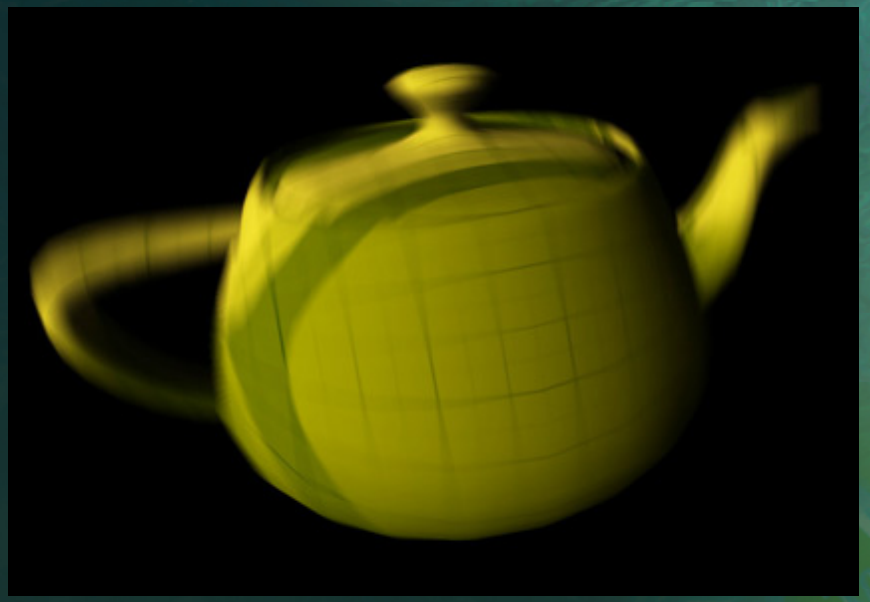
反射

- 在某些情况下，可以代替所有镜面效果
- 可以使用虚拟机（VM）来生成CUBE贴图
- 可以具有有限的射线（请参看后文）
- 能够以二次方程衰减方式确定距离（请参看后文）

Environment-mapped background, reflected card-shaped light source, 16-bit blending with overbright bloom

新领域：镜头特效

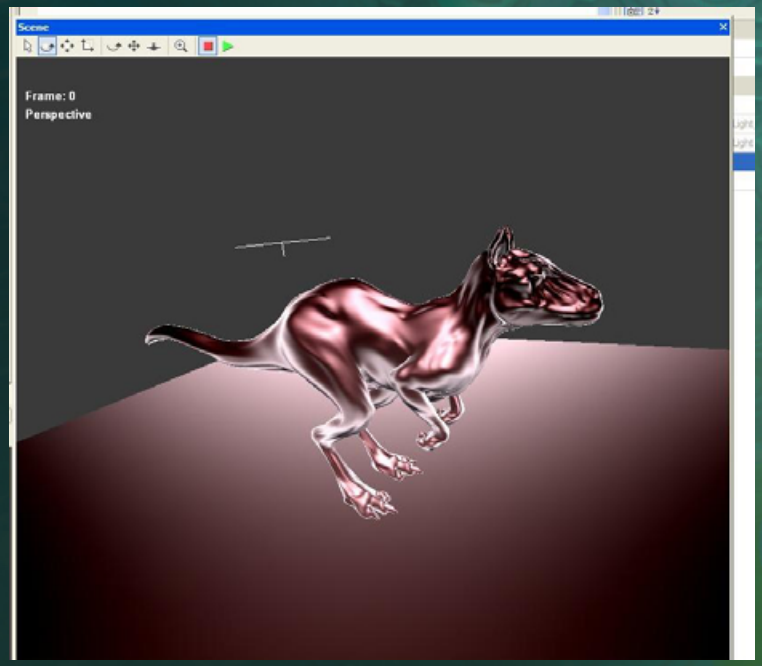
- “累积缓存”技术具有下列优势：
  - 动作模糊处理
  - 场景深度
  - 柔和阴影
  - 更多 ….
- 没有什么特别的着色要求，但着色器必须*速度快*



Motion blur

最大限度地利用Direct X 虚拟机

- 纹理生成
- CPU上的 “纹理着色器” 能够生成图像，或创建（包含可预见的功能）纹理
- 利用HLSL 固有属性的矩阵处理可以将复杂的阴影效果变成现实的功能



Dinosaur with Physically-based car paint BRDF

合成与2D特效

- FP缓存使功能空前强大
- 许多乐趣…
- 颜色控制
- 最终 “润色”
- 混合模式
- 混合2D/3D角色
- 浮点像素

Halftoning Patterns

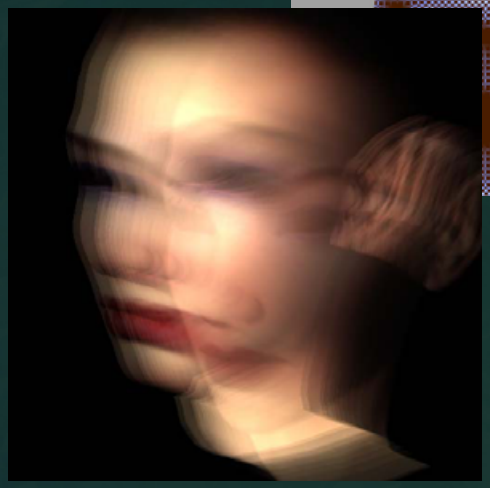
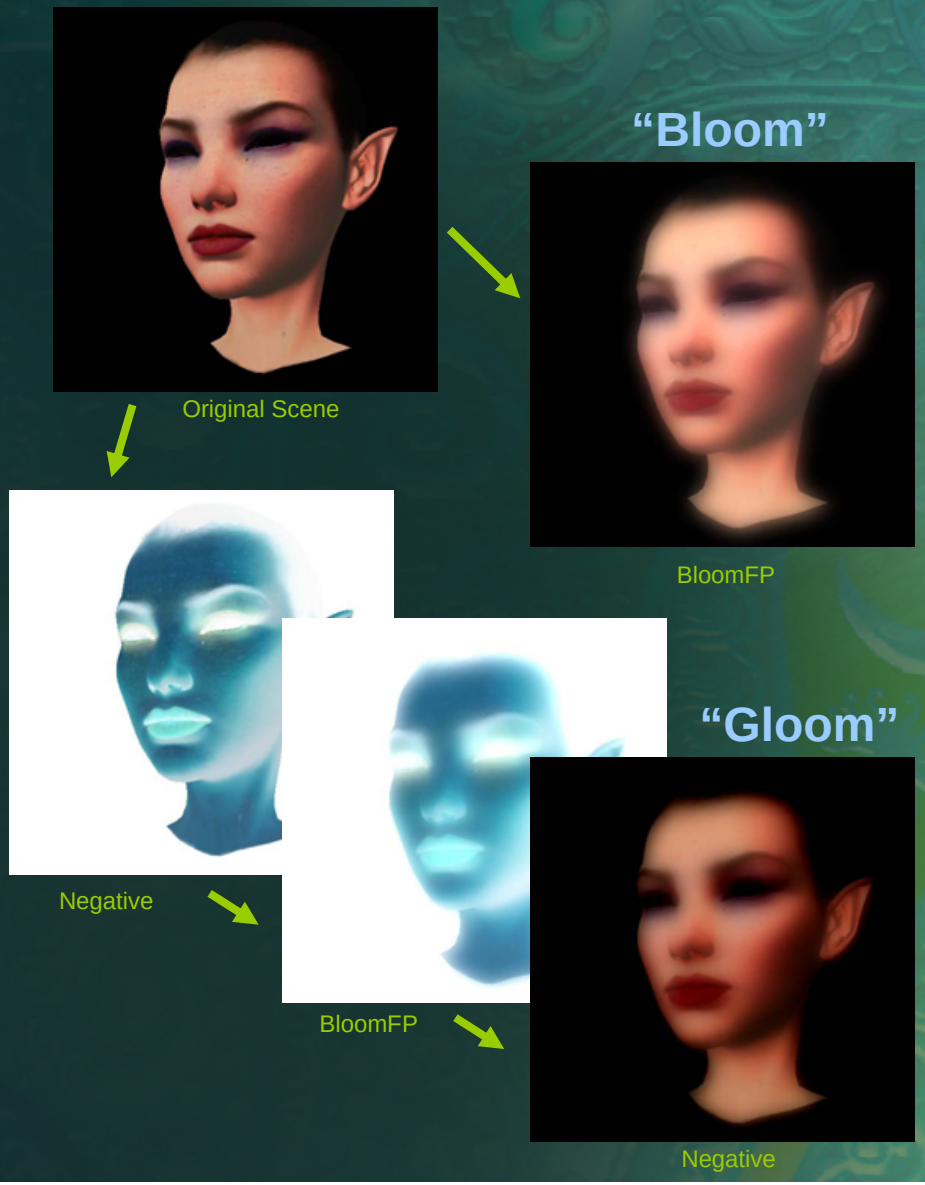


Image Trails

后处理：锦上添花

- 鲜亮的鲜花场景很好地展示了复杂性和规模方面的优势
  - 这有点像录音棚里的“回音”效果，令人扼腕叫绝！
- 我们可以在FX编辑器里堆叠图像特效，从而创造出新颖而且更为复杂的特效



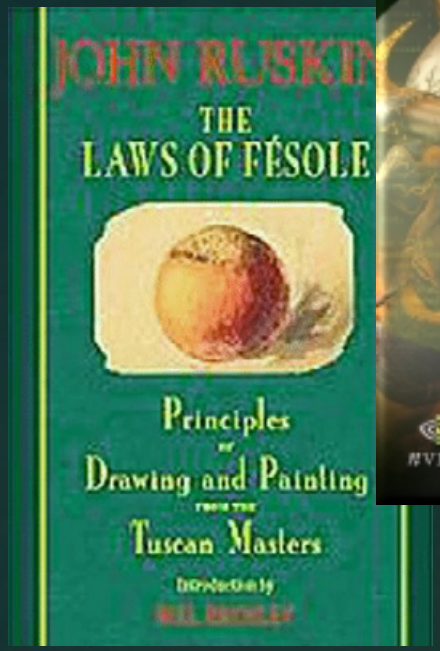
随心所欲！

- 现在，游戏也能有电影的着色效果了。也许像素还比不上电影，但角色已经具有极高的精美程度。
  - 习惯使用*大量*的着色器
  - 获取工具，开始操作
  - http://www.fxcomposer.com/
  - 操作着色器，尝试每个功能，用 “草稿本” 记下每个有用的主意

The End

推荐书目

- Randima Fernando: *GPU精华*
- John Alton:*用光线作画*
- Jon Ruskin: *Fésole定律，托斯卡纳大师们的绘画原则*



http://developer.nvidia.com/object/GPU_Gems_home.html

更多相关内容

- http://developer.nvidia.com/
- http://www.fxcomposer.com/
- http://developer.nvidia.com/object/sdk_effects.html
- kbjorke@nvidia.com