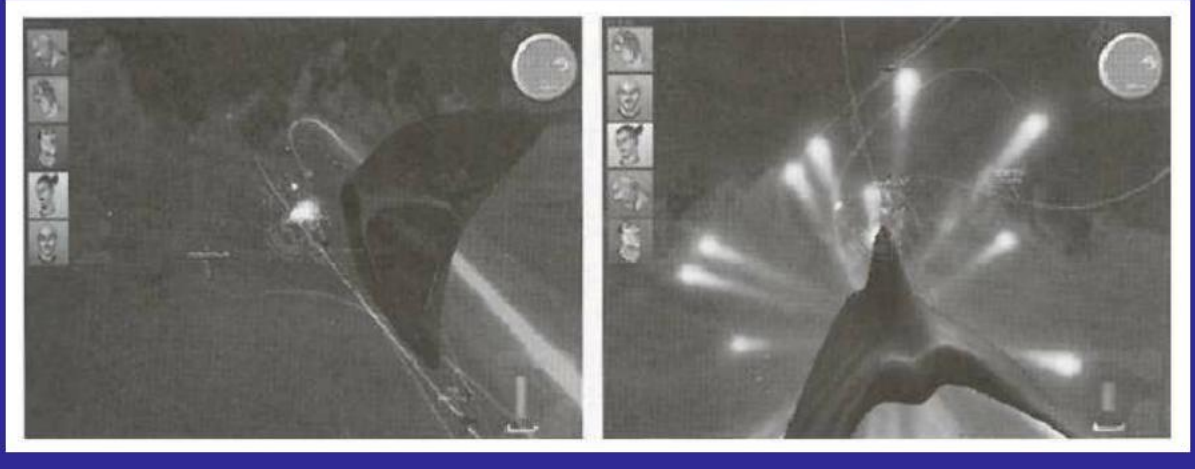
游戏中的特效绘制技术

华东师范大学软件学院

Billboard技术

- 几何与图像混合绘制方法；
- 一个带有纹理的四边形，可随着相机的运动而运动，常常与Alpha配合使用；
- 四种：
  - 平行屏幕的Billboard技术：法向与视线重合
  - 平行屏幕的Billboard技术：法向与视线平行
  - 视点朝向的Billboard技术：法向取视点到Billboard中心的连线
  - 轴向Billboard技术

- 这种方式的billboard在模拟很多自然现象是很有用。[298]讨论如何使火焰、烟、和爆炸等看起来不重复。
- Dobashi[182]绘制云，并且生成其阴影



Impostor技术

- 替身图方法利用游戏每帧间的连续性，采用二维图像和三维模型的投影替代三维物体；
- 创建替身图：
  - 将三维物体绘制到一个纹理中
- 绘制替身图
  - 相机变焦或物体距离变化时，二维投影尺寸变化；
  - 尺寸大于某一阈值，更换替身图

- Imposter是从当前视点出发，将一个复杂的物体绘制到一张图像上，作为texture贴到一个多边形上，如billboard。其动机为：显示一张图像的时间远小于画一大堆多边形的时间。其代价为需要更新的图像

优点：

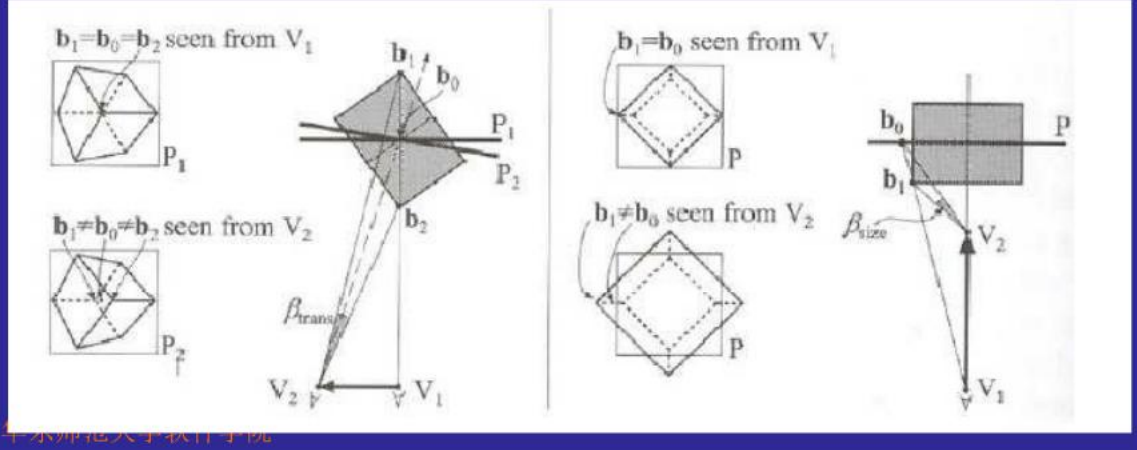
绘制远处的物体很有效

一堆小的静态物体

与用极简单的模型相比，imposter不以牺牲细节为代价

由于imposter是用图像表达的，可以用低通方式采样，以获取景物的焦距不准的情形.

- Imposter在什么情况下有效?什么情况下需要更新?
- 当视点仅仅是转动时，不用更新



- 更新是绘制中最为重要的计算
- 游戏
- 启发式修改纹理坐标已减少视差错误
  - 预定义一个最大距离，当经过n帧以后，其在屏幕上的阈值，则更新
  - 为了实其更有效，每次绘制一个大的纹理中的一些子区域
- 动态物体
  - 当其运动时，使用3D模型
  - 当其静止时，使用imposter

粒子系统

- 粒子系统是一系列独立个体的集合,它们以一定的物理规律和生命周期在场景中运动.
- 粒子具有一些属性：位置，速度，加速度，能量，方向等
- 在粒子运动过程中，粒子属性被显示，修改和更新
- 粒子的变化规律受到物理规律的影响
- 既有随机性，又有规律性
- 可以用来模拟烟，火焰，爆炸，雨雪，血溅等

- 基本过程
  - 初始化粒子
  - 当程序运行时
    - 如果粒子没有消亡
      - 根据粒子的速度更新粒子的位置
      - 根据粒子的加速度更新粒子的速度
      - 修改粒子的能量
    - 如果粒子的能量小于某一阈值
      - 设置粒子的状态为消亡
    - 如果粒子击中场景物体或其他粒子
      - 修改粒子的位置，方向，速度和能量
    - 显示粒子
  - 程序结束

难点

- 粒子：长方形，Pointsprite和三维几何模型，Billboard
- 初始化函数和状态更新函数

- 加快速度：粒子消亡后设置初始化，不释放出内存
- 与游戏引擎的无缝连接

爆炸

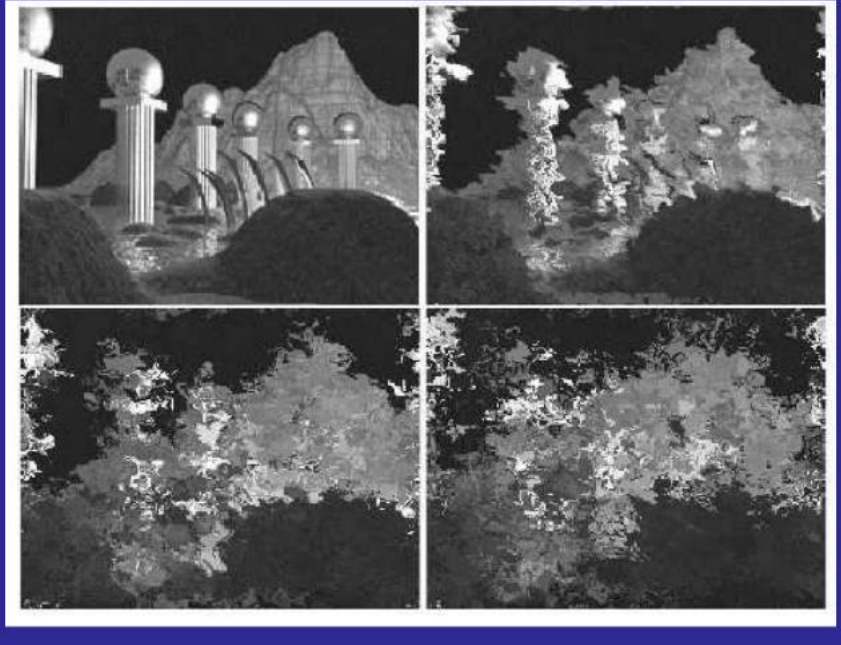
- 最常见的就是采用粒子系统模拟，也有采用Billboard模拟爆炸的。
- 粒子分步在以爆炸点为中心的球面，从球心以高速和巨大的能量向外发射
- 有遮挡的爆炸：在半球面上分布粒子

火焰

- 二维随机函数
  - 设置火源：在预期的位置生成火焰的中心
  - 选择随机函数生成火焰的形状和颜色
  - 选择正确的图像模糊算法，模拟火焰热源扩散

- Billboard＋精灵动画
  - Perlin噪音函数生成火焰
  - 数个Billboard模拟火苗运动
  - 扰动纹理来实现火苗的串升和风吹效果
  - 模拟用的动态纹理可以采用实拍的视频

Image Warping



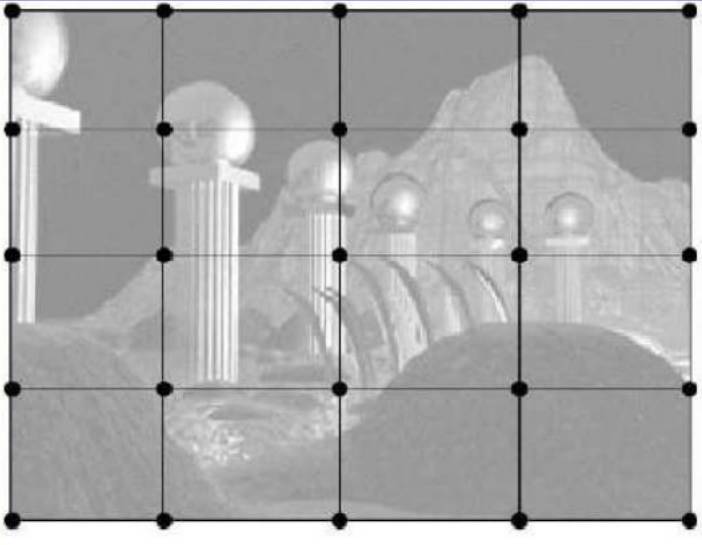


Figure 14.2

Putting a grid of squares down on top of an image, in preparation to warp it.

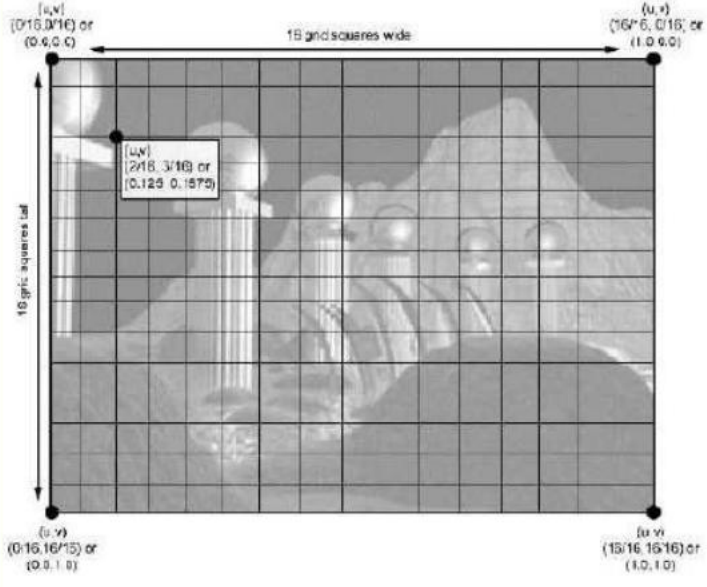


Figure 14.5

Calculating the texture coordinates for a particular grid point.

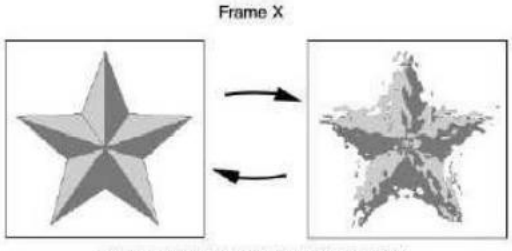


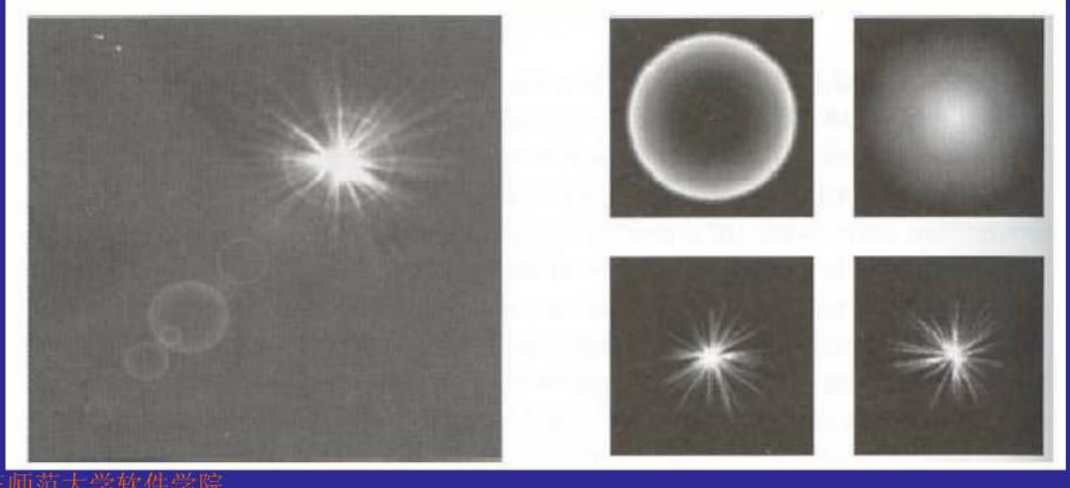
Figure 14.6

Swapping textures to feed the warped image back onto itself.

- Wapping.ppt

透镜眩光和敷霜效果
Lens Flare and Bloom

- 透镜眩光是由于眼睛的晶状体或者相机的透镜直接面对强光所产生的现象。由一圈光晕和纤毛状的光环组成

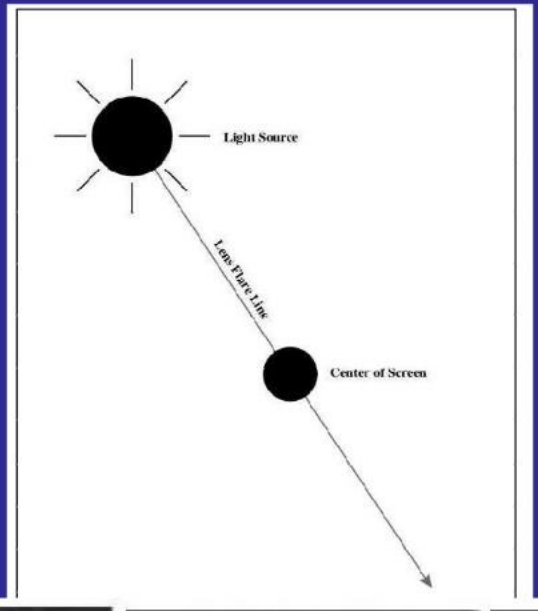


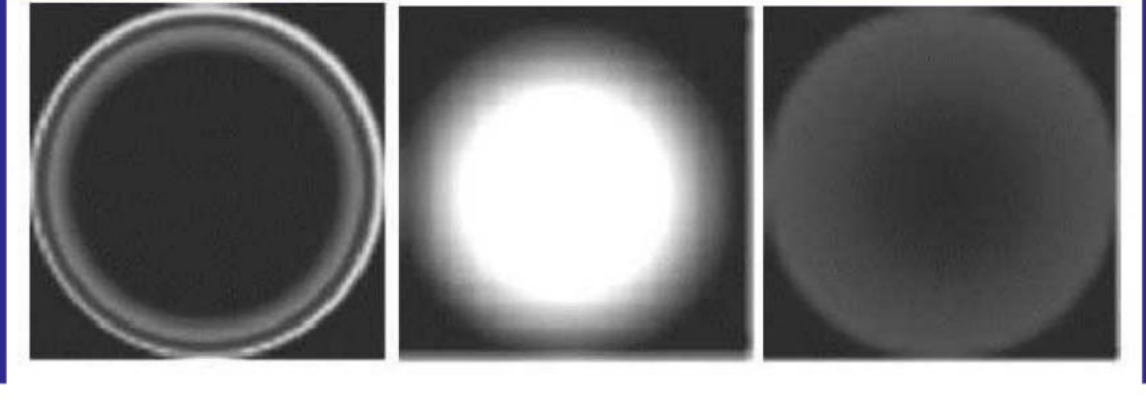
The Concepts

Lens flares themselves are simple little creatures. You draw a straight line that passes through both the light source and the center of the screen. You then arrange the flare spots along this line (see Figure 22.2).

Give each flare spot a color, an alpha component, a size, and a shape, and you've got the basics.

Note that these images are essentially alpha masks—the white pixels denote complete opacity, and the black pixels denote complete transparency. We'll modulate the texture colors with the vertex diffuse colors to colorize these images for the actual flare. We'll then render the flares using additive alpha blending.


Light Source
Lens Flare Line
Center of Screen



Process:

```
if the sun is on the screen
  intensity = 1, and we're done
else
  determine how far away the sun is from the edge of the screen by
  taking the greater of either the horizontal distance or the
  vertical distance.  call this "distance."
  take distance and divide it by the size of the flare border.
  Call this the "inverse intensity."
  if the inverse intensity is greater than one, set it equal to one.
  intensity = 1 - inverse intensity
end if
for each vertex of each lens flare spot
  multiply the alpha component of the vertex's diffuse color
  by the intensity.
  render the flare
next
```