

基于粒子系统的实时降雨、降雪模拟^①

刘 芳¹, 佟 巍²

¹(东北石油大学 计算机与信息技术学院, 大庆 163318)

²(大庆油田第一采油厂信息中心, 大庆 163318)

摘 要: 基于粒子系统理论实现了虚拟环境中降雪、降雨及其在风场中的模拟. 首先阐述了粒子系统的基本模拟原理, 对在风场影响下的雨、雪特征进行描述与分析, 提出了雨、雪模拟的整体思路, 结合降雨、降雪的特点和空气动力学理论对雨、雪进行了模拟. 改进四点水波纹线性模拟算法, 采用八点水波纹线性算法模拟雨滴落入水面形成的水波纹, 应用雾化因子实现降雪环境的模糊问题.

关键词: 粒子系统; 降雨; 降雪; 模拟

Real-Time Simulation of Rain and Snow Based on Particle System

LIU Fang¹, TONG Wei²

¹(Computer and Information Technology, Northeast Petroleum University, Daqing 163318, China)

²(Information Center, the First Oil Production Company, Daqing Oilfield, Daqing 163318, China)

Abstract: Based on the theory of particle system, the simulation of the snow, rain and their real simulation in wind field in the virtual environment was implemented in this paper. The paper firstly illustrated the basic simulation principle of the particle system, and then described and analyzed the characteristic of the rain and the snow under the influence of the wind field; put forward the overall train of thought of the rain and the snow simulation. The paper simulated the rain and the snow according to the characteristics of the rain and the snow, the aerodynamics theory of them. By improved the Four-point linear water ripple simulation algorithm, the Eight-point water ripple linear algorithm was adopted to simulate the water ripple formed by the raindrops, atomization factor was applied to solve the fuzzy problem of snow environment.

Key words: particle system; rain; snow; simulate

自然景物模拟, 是计算机图形学领域具有挑战性的研究方向之一^[1,2]. 雨、雪是虚拟环境中较为常见的一种自然景物, 真实的模拟降雪和降雪能够增强虚拟环境的沉浸感和真实感. 虽然人们对于雨雪并不陌生, 但要给它们下一个确切的定义, 构建一个具体的模型却十分困难. 在虚拟环境中, 如何建立恰当的模型, 生成具有真实感的、实用性的降雪、降雨效果, 是本文要研究的问题.

1 粒子系统概述

1983 年 Reeves 提出粒子系统理论并提出用该理

论模拟不规则物体, 它的基本思想是: 把不规则模糊物体看作是由众多粒子组成的粒子团, 粒子团中的每个粒子均有自己的属性, 粒子随时间的推移不断运动并根据所模拟的景物的动力学属性来改变自身的状态^[3]. 粒子系统建模是一种过程模型, 即利用各种计算过程生成模型各个体素的建模技术, 它将几何建模和行为建模统一了起来.

粒子系统的基本工作步骤如下:

① 粒子的产生包括粒子产生的空间、粒子产生的时间以及数量. 根据模拟对象的不同, 粒子产生的空间也不同, 如、雨雪、火焰、瀑布, 空间都有所不同. 粒

① 基金项目: 黑龙江省高等教育科学研究“十二五”规划课题(HGJXH B1110131)

收稿时间: 2014-10-29; 收到修改稿时间: 2014-12-08

子数量的确定是一个经验值, 一般需要反复实验得到.

② 粒子的属性, 每个粒子都具有一定的属性, 粒子的属性决定了粒子的状态, 根据模拟对象的不同, 粒子的属性可能不完全相同, 但一般都包括以下属性:

形状、大小、颜色、透明度、位置、速度、生命期等.

③ 粒子的运动, 新的粒子产生并初始化后, 粒子根据初始化时速度的大小及方向开始运动, 同时, 粒子的相关属性也随着时间的变化而不断的改变. 粒子的空间位置主要由粒子的前一位置、粒子的速度和每帧的时间间隔决定.

④ 粒子的消亡, 粒子产生后, 经过一定的时间间隔, 由于其生命周期结束或者其它原因从系统中被除去的过程称为粒子消亡过程.

⑤ 粒子的渲染, 当某一帧的粒子状态全部确定以后, 粒子控制机制会调用绘制函数将其送往显示缓存, 得到用户所需的效果.

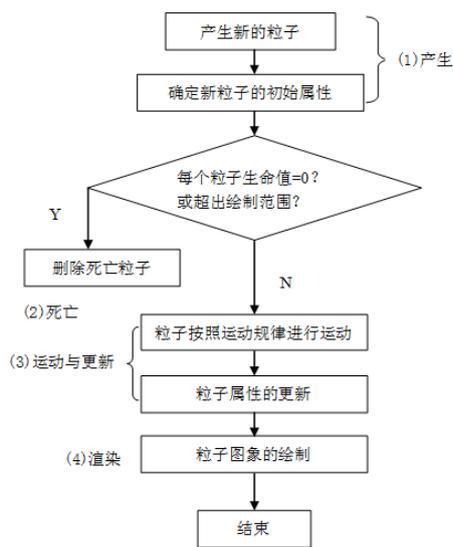


图 1 粒子系统基本工作流程

在基于粒子系统方法进行模拟的过程中, 如果对粒子处以相同的算法就会产生规则的图形图像, 这会具有较强的人工处理痕迹, 为了避免这个问题, 在用粒子系统解决问题的过程中通常会引入“随机处理方法”, 这是粒子系统模拟解决不规则物体的核心思想, 这个随机处理思想会体现在基于粒子系统方法模拟生成图像的整个过程中, 如图 1 所示的粒子产生过程、粒子属性的确定、粒子的运动、属性更新以及粒子死

亡等过程.

2 关键技术及实现

2.1 基于粒子系统的雨的模拟

降雨在自然界中较为常见, 它的运动十分不规则, 因此使用粒子系统来进行模拟是十分合适的. 每一个雨滴可用一个粒子来表示, 基于适当的运动模型, 采用合适的渲染方法, 就能够模拟降雨这一自然景物^[4].

(1) 雨粒子的基本属性: 包括外观、空间、运动、生命周期等. 本文对雨的外观采用了球形粒子, 为了便于计算, 在不显著影响实时性的前提下, 对初始化分布算法进行适当简化, 将雨粒子分布区域由视锥所构成的棱台简化为眼前的一个长方体, 即产生于虚拟场景中的矩形天空里.

虽然雨粒子整体的运动趋势是向下运动, 但是对于每一个粒子还是表现出很大的随机性, 因此对粒子的速度, 我们不能用固定数值.

第 i 个粒子的速度描述如下:

$$particlelist[i].velocity = VelocitySnow + VelocityDelta \times rand();$$

$VelocitySnow$ 为粒子速度的平均值, $VelocityDelta$ 为粒子速度的方差, 它们的数据类型都是三维矢量 ($vector3$), $rand()$ 是 $[-1, 1]$ 上均匀分布的随机函数.

对于粒子生命周期的判断, 以粒子的运动范围为判断依据, 即当粒子运动到可视范围之外时, 则认为该粒子的生存期结束, 粒子消失, 系统停止对其的计算和绘制.

根据上述分析, 雨粒子的数据结构如下:

```

Struct vector
{
    Float x;           //矢量的 x 轴向
    Float y;           //矢量的 y 轴向
    Float z;           //矢量的 z 轴向
}

Struct Particle
{
    Vector pos;        //粒子的位置
    Vector velocity;   //粒子的速度
    Float d;           //球形雨粒子的直径
    Float color[4];    //粒子的颜色
}
  
```

(2) 雨粒子的运动模型

雨的动力学行为较为复杂. 实际观测表明, 雨滴的最小直径为 0.5 mm, 最大直径不超过 6mm, 最大的降落速度为 9m/s. 由于不同大小雨滴达到匀速下降时存在速度差, 下降速度较快的大雨滴与下降速度较慢的小雨滴相遇就会聚合成更大的雨滴, 但随着雨滴体积的增大, 空气阻力也随之增大. 当雨滴直径大于 6mm 时, 空气阻力超过了使雨滴保持整体的分子内聚力, 大雨滴便又碎裂分解成小雨滴, 发生大雨或暴雨天气时的降雨就会出现上述现象^[5]. 因此, 实际雨滴的落地速度可能已经达到匀速极限速度, 也可能仍处于加速降落状态. 为了适合模拟的需要, 本文对雨的行为做如下简化假设.

(a) 在下降过程中, 雨滴不会发生分裂、聚合现象.

(b) 在模拟过程中, 有雨滴处于加速状态、也有雨滴处于加速后的匀速极限运动状态, 假设处在匀速运动状态的雨滴保持这一速度运动, 速度不做更改.

经过模拟对比建立了雨粒子的运动模型包括无风时雨滴自由降落运动模型, 无风时雨滴匀速降落运动模型.

无风时雨滴自由降落运动模型:

$$dv/dt = \frac{1}{1 + \frac{1}{2}\rho} \left[(1 - \rho)g - \frac{3\rho}{4d} v^2 C_d \right]$$

模拟计算中取雨滴密度 $\rho = 1000 \text{kg/m}^3$, 空气密度 $\rho = 1.22 \text{kg/m}^3$, 时间步长 $h = 0.02 \text{s}$, 重力加速度 $g = 9.8 \text{m/s}^2$, 空气运动粘滞系数 $\nu = 1.493776 \times 10^{-5} \text{m}^2/\text{s}$, 计算精度控制速度误差小于 0.0002m/s 时认为已达到匀速降落. 根据数值模拟计算结果与实测雨滴速度比较, 当雨滴直径 $d < 1.5 \text{mm}$ 时, 雨滴形状能近似保持圆球形.

无风时雨滴匀速降落运动模型:

$$v = \begin{cases} 3.8435d^{0.9201} & 0.5 \text{mm} \leq d \leq 1.5 \text{mm} \\ 4.6254\sqrt{d} & 1.5 \text{mm} \leq d \leq 4.0 \text{mm} \\ 9.0 & 4.0 \text{mm} \leq d \leq 6.0 \text{mm} \end{cases}$$

上式为雨滴达到匀速降落时的极限速度分段公式, 其中雨滴直径 d 以 mm 为单位, 速度 v 以 m/s 为单位. 根据雨滴降落运动微分方程

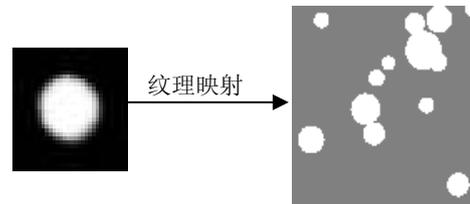
$$mdv/dt = mg - m_a g - \frac{1}{2} m_a dv/dt - \frac{1}{8} C_d \rho_a v^2 \pi d^2$$

雨滴所受的重力、空气浮力、加速降落产生的作用力和空气粘性阻力等, 即 \vec{F} . 从而得到 $\frac{dv}{dt} \approx \vec{F}$, 即

$$\frac{dv}{dt} \approx \vec{F}_g + \vec{F}_a + \vec{F}_f + \vec{F}_v$$

它们是雨粒子不断运动的动力.

基于上述运动模型^[6], 本文选择圆球体作为雨粒子的基本形状, 但是 OpenGL 绘制球体会采用较多的三角形面片, 这样会增加粒子绘制的开销, 在同等条件下试验, 采用真实绘制球形粒子的帧频为 20 帧/秒, 而采用正方形粒子绘制的帧频为 80 帧/秒, 因此, 雨粒子的绘制采用基于 OpenGL 二维纹理映射的方法来实现^[7]. 如图 2 所示.



(a) 雨滴纹理

(b) 雨滴

图 2 雨粒子的纹理映射

(3) 水波纹模拟

由于雨滴落入水面后, 会形成水波纹, 因此水波纹的模拟是模拟真实降雨所必须的. 本文在实现水波纹模拟的时候, 对以前四点水波纹模拟算法作了一定的改进^[8], 改进的思路是在某一时刻水面上某点的振幅由该点周围的八个点以及该点自身的振幅来决定, 即由原来四个点改为八个点.

在算法实现中, 将水波的扩散过程视为水面上各点振幅的变化过程. 程序中定义了一个三维数组 $\text{wave}[k][i]$, 其中 $k=0$ 或 1 表示下一时刻的当前点的振幅, 0 表示当前时刻该点的振幅; i, j 分别表示水面控制点的列数和行数. 设初始状态时水面为一个平面, 各点的振幅数值均为 0 .

基于 OpenGL^[9], 水波纹模拟算法核心代码如下:

```

Void ProcessWave()
{ float temp;
  for (int i=1; i<WAVEX-1; i++)
    for (int j=1; j<MAVEY-1; j++)
      {
        temp = ((wave [kn][i-1][j] + wave [kn][i+1][j] +
                wave [kn][i][j-1] + wave [kn][i][j+1] +
                wave [kn][i-1][j+1] + wave [kn][i+1][j-1] +
                wave [kn][i-1][j-1] + wave [kn][i+1][j-1]) / 4)
                - wave [kl][i][j]; //波的传播
        temp = temp - (temp / damp); //传播的过程中减弱振幅
        if (temp < 0.01) temp = 0; //波消失
        wave [kl][i][j] = temp;
      }
}
void InvertWave () //交换缓冲区中的数据
{ int temp = kn;
  kl = kn;
  kn = kl; }

```

水波的产生需要波源,在降雨模型中,水波的波源是雨滴,当雨滴落入水面后记录下雨滴的水平坐标位置,就是水波的波源。另外,程序中 WAVEX 和 MAVEY 的选取也与降雨模型中,雨滴的初始化产生的水平位置及大小有关。程序中的 damp 为衰减率, $damp=1/2n$ 通过实验得知 n 的取值范围在 5~9 之间较为合适。

本文采用的八点水波振幅线性模拟算法是基于粒子系统的波的线性模拟算法,与四点相比较它能够得到较好的渲染效果,与传统的基于波的分析^[10]和基于物理模型的波浪模拟^[11]相比,它又能获得较快的渲染速度,又能够对水波的破碎性进行描述,因此适用于实时渲染的虚拟环境。

2.2 基于粒子系统的雪的模拟

雪景是也自然界中比较常见的自然景物。

(1) 雪粒子的基本属性

雪花形状进行统一处理,采用公认的六角形作为雪花粒子的基本形状,具体的绘制方法在粒子的绘制小节中介绍。

雪花的颜色。由于自然界中的雪是白色的,所以雪粒子的颜色取为白色。

雪花的尺寸大小。雪粒子的尺寸是一个经验值,通过反复实验,观察场景的降雪效果,才可确定。

第 i 个雪粒子大小描述如下:

$$particlelist[i].size = SizeSnow + SizeDelta \times rand()$$

在这里 size 表示该雪粒子的尺寸, sizesnow 表示雪粒子尺寸的均值, sizedelta 表示雪粒子尺寸的方差,

rand()是[-1,1]上均匀分布的随机函数。

最后本文确定的尺寸均值为 0.05f; 尺寸的方差为 0.01f。

其余属性与雨粒子类似。

(2) 雪粒子的运动模型

由于雪花是在空中下落,本身的质量非常小,因此它受外界环境的影响较大,并且雪花很轻,在此可以忽略其重力的影响^[12]。

本文在雪花粒子的运动上做一假设,认为雪花水平方向上的运动为空气运动,垂直方向上则为飘动。

首先确立无风时的运动描述:

即第 i 个粒子的位移和速度描述如下:

$$particlelist[i].pos = particlelist[i].pos + particlelist[i].velocity \times \Delta t;$$

$$particlelist[i].velocity = VelocitySnow + VelocityDelta \times rand();$$

通过反复实验确定雪粒子速度的均值和方差,如下:

$$velocitysnow=(0.0f, -1.0f, 0.0f);$$

$$velocitydelta=(0.25f, -0.2f, 0.25f);$$

数值表示大小,符号表示方向。

雪粒子的渲染也采用纹理映射的方式,采用的为正方形粒子。如图 3 所示。

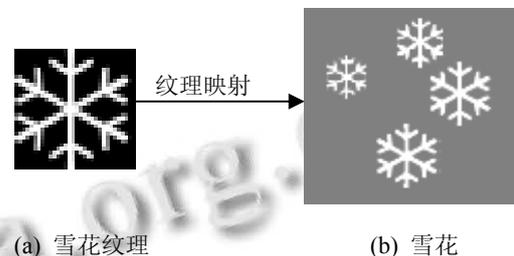


图 3 雪粒子的纹理映射

为了增加真实感可以再虚拟环境中加入雾化效果,距离视点较近的景物相对清晰,而远处的景物会随着距离的增加变得越来越模糊,使用雾化能够显著的增加视觉效果的真实感,深度感。本文对降雪环境的雾化模拟选择 $f = 1/e^{(density \times z)}$ 的模式, f 为雾化方程的混合因子。可用于模拟自然界的雾现象,使整个环境看上去较为均匀^[13]。

2.3 雨、雪在风场下的模拟

雨、雪粒子在空气中除了受到垂直方向的重力影响外,水平方向还会受到风力影响。本文给出两种常见简化的风力模型:阵风和稳定风。阵风指的是风力从零开始逐渐加大,达到顶峰后,然后再逐渐衰减至

零. 稳定风是指风力增加至一定程度后保持该强度并存在微小的变化最后逐渐衰减至零.

经简化的两种风力模型的数学表示如下:

阵风函数:

$$\begin{cases} at + b & 0 \leq t \leq tc \\ c - d(tc - t) / tc & tc \leq t \leq t_maxtime \end{cases}$$

稳定风函数:

$$\begin{cases} at + b & 0 \leq t \leq tc \\ c + d \cos(t) & tc \leq t \leq t_midtime \\ e - f(tc - t) / tc & t_midtime \leq t \leq t_maxtime \end{cases}$$

其中, tc 为时间常数, a 、 b 、 c 、 d 、 e 、 f 可根据实际情况考虑赋予不同的数值, 本文对阵风模型选取 $a=c=d=1$, $b=0$; 对稳定风 $a=c=e=f=1$, $d=0.3$.

对粒子加上风场的影响后, 能够明显增强虚拟环境的真实感^[14].

2.4 系统的实现

系统在 OpenGL 和 Visual Studio.NET 2010 开发环境中进行了实验, 测试的硬件平台为 AMD athlon(tm)64*2DualCore2.50GHz, 1G 内存对真实绘制球体和采用 OpenGL 纹理绘制的两种方式进行试验, 不同粒子数下两种试验方法的渲染帧速如表 1 所示.

表 1 雨粒子形状和数量对比实验结果

粒子数量	绘制方法	帧速率 (帧/秒)	可视效果
150	真实绘制	32	小雨, 基本满足视觉需要
	纹理模拟	91	
500	真实绘制	21	中雨, 满足视觉需要
	纹理模拟	78	
1000	真实绘制	10	大雨, 满足视觉需要
	纹理模拟	56	

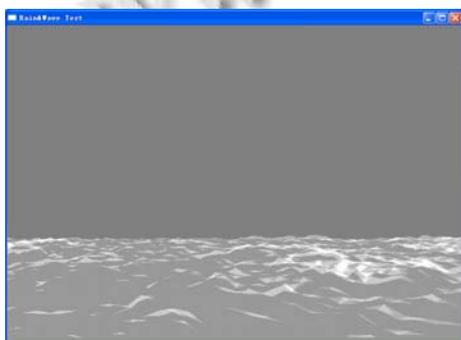


图 4 density=0.8 时的降雪渲染效果

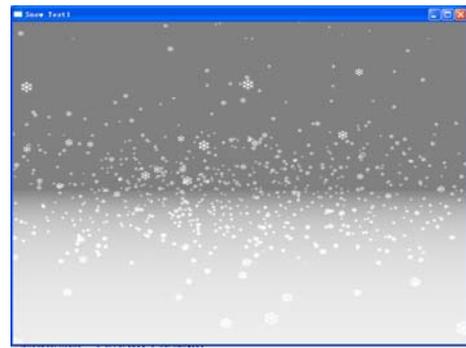


图 5 水波渲染效果图

3 结语

本文重点研究了基于粒子系统的雨、雪以及风场中的雨雪等自然现象的模拟. 针对降雪环境的模糊问题和雪花落地的积雪现象提出了将雾化效果应用于降雪虚拟环境中的方法, 通过选择雾化因子模式, 控制雾化浓度来控制环境的模糊程度. 对雨滴进行了模拟并针对雨滴落入水面形成的水波纹问题进行研究, 在四点水波纹算法基础上提出了适合实时渲染虚拟环境的八点水波纹线性模拟算法, 模拟效果良好. 并在风场的环境下对雨雪进行了模拟.

参考文献

- 周惠, 陈善广, 袁宏, 沈旭昆. 航天视景仿真特殊效果图生成技术. 系统仿真学报, 2001, 13(6): 711-713.
- 张芹, 谢隽毅, 吴慧中, 张正军. 火焰、烟、云等不规则物体的建模方法研究综述. 中国图象图形学报, 2000, 5(3): 186-190.
- Reeves WT. Particle system—a technique for modeling a class of fuzzy objects. Computer Graphics, 1983, 17(2): 359-376.
- 李苏军, 吴玲达. 基于粒子系统的实时雨模拟. 计算机工程, 2007, 33(18): 236-238.
- 罗维佳, 都金康, 谢顺平. 基于粒子系统的三维场地降雨实时模拟. 中国图象学报, 2004, 9(4): 495-500.
- 李苏军, 吴玲达. 基于粒子系统的实时雨模拟. 计算机工程, 2007, 33(18): 236-238.
- 左鲁梅, 黄心渊. 纹理映射技术在三维游戏引擎中的应用. 计算机仿真, 2004, 21(10): 146-148.
- 陈和平, 王早. 水波特效模拟原理及其快速实现方法. 计算机应用研究, 2005, 22(4): 252-253.
- Dave S, Mason W, Jackie N, Tom D. OpenGL 编程指南. 第 5 版. 北京: 机械工业出版社, 2006.
- Simon P, Michael A. Rendering Natural Waters. Eight Pacific Conference on Computer Graphics and Applications, October 2000. 140-143.
- 徐迎庆, 苏成, 李华, 刘慎权, 齐东旭. 基于物理模型的流水及波浪模拟. 计算机辅助设计与图形学学报, 1997, 9(2): 191-192.
- 王照, 李光耀. 基于粒子系统的模糊物体间交互模型设计, 2010, 30(Z1): 268-269.
- 刘芳, 刘贤梅. 基于粒子系统和雾化的实时降雪模拟. 微计算机应用, 2011, 32(7): 1-5.
- 徐利明, 姜昱明. 基于粒子系统与 OpenGL 的实时雨雪模拟. 计算机仿真, 2005, 22(7): 242-245.