

# 基于粒子系统的三维降雪场景仿真

周 强,汪继文

(安徽大学 计算机科学与技术学院,安徽 合肥 230039)

**摘 要:**通过研究粒子系统方法,结合面向对象的 C++ 语言,建立粒子系统的过程模型,分析实际降雪物理运动过程,建立雪粒子系统三维模型,简化雪花下降过程中的运动模型,省去对速度的实时每帧控制。通过控制雪粒子位移来模拟受风力影响的实际宏观降雪雪景,同时通过参数控制实现风力强弱的影响效果和大雪小雪的景观控制,并在笛卡尔三维坐标系的  $z$  轴上引入指数函数  $F = e^x$  作为控制因子,使雪粒子满足近大远小的透视投影视点效果,具有三维景观效果,并结合 OpenGL 图形接口对降雪场景进行渲染。实验结果表明,该方法具有逼真的三维仿真效果且代码具有良好的可扩展性。

**关键词:**粒子系统;雪景模拟;OpenGL;纹理映射

中图分类号:TP302

文献标识码:A

文章编号:1673-629X(2017)01-0130-04

doi:10.3969/j.issn.1673-629X.2017.01.029

## Three-dimensional Simulation of Snowing Based on Particle System

ZHOU Qiang, WANG Ji-wen

(College of Computer Science and Technology, Anhui University, Hefei 230039, China)

**Abstract:** A falling snow algorithmic structure is proposed based on particle system, incorporating the object-oriented C++ language. A 3D model of particle system is built by analyzing the falling process of snow, simplified the motion model of the snowflakes falling process and saved each frame of real-time control for speed. The realistic snowing is simulated which affected by wind through controlling the particles displacement. And on the  $z$  axis of three-dimensional Cartesian coordinate system, the index function  $F = e^x$  is introduced as the controlling factor so that the simulation matches the perspective effect of perspective projection. The simulation is combined with OpenGL graphic interface. The experimental results show that this method has a realistic 3D simulation effect and the codes enable a good scalability.

**Key words:** particle system; snowscape simulation; OpenGL; texture mapping

### 1 概 述

随着现代社会的发展,人们对虚拟现实的需求越来越大,在计算机游戏、动画以及影视广告中都有着广泛的应用需求。而自然景物诸如海浪、云、烟、火焰、雨、雪的虚拟模拟,因其运动的不规则性和动态性,成为计算机图形模拟中的难点问题。

粒子系统 (particle system) 作为模拟不规则物体最成功的算法之一,被研究应用至今。自 Reeves W T<sup>[1-2]</sup> 于 1983 年首次提出粒子系统,并成功模拟了火焰、爆炸等效果以来,国内外专家已经用粒子系统方法的思想成功模拟出了许多自然景物,并不断对模拟进行改进,以追求更高的真实性和实时性。Coutinho 等<sup>[3]</sup> 在他们的研究中用粒子系统方法生成雨滴。Latta

L 等<sup>[4]</sup> 采用 GPU 进行粒子系统模拟,能实时处理超过 100 万个粒子,使得大规模粒子模拟变成了可能,也使越来越多的人开始关注并研究基于 GPU 的粒子系统动画模拟。国内的袁霞和张玉琢<sup>[5]</sup> 对粒子系统的方法及应用作了较为详尽、系统的阐述。金小进等<sup>[6]</sup> 利用粒子系统模拟了火焰,并在渲染过程中分别从实时性和真实感两方面对粒子系统进行了优化。汪继文等<sup>[7]</sup> 利用粒子系统进行了烟花仿真,并实现了对烟花形状的控制。潘秋羽等<sup>[8]</sup> 提出了基于粒子系统的快速的云三维仿真算法,能快速地产生形状各异的真实感较强的三维云。何亮等<sup>[9]</sup> 基于粒子系统模拟了动态雪景,但是所建物理模型较简单并没有考虑环境因素。徐利明等<sup>[10]</sup> 提出了一种在大型场景漫游系统中实时模拟

收稿日期:2016-03-07

修回日期:2016-06-15

网络出版时间:2017-01-04

基金项目:安徽省省级重点自然科学基金项目(KJ2013A009)

作者简介:周 强(1991-),男,硕士研究生,研究方向为计算机仿真、图形图像处理;汪继文,博士,教授,博士研究生导师,研究方向为计算流体力学、计算机仿真、图像处理、智能算法等。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20170104.1028.046.html>

雨、雪的方法。单在雪粒子的模拟中,对粒子系统的方法就有不同的研究方向。刘金瑄等<sup>[11]</sup>基于粒子系统和 XNA 平台工作原理,利用 C#语言在 XNA 平台下对飘雪进行了仿真。许金生和宋万忠<sup>[12]</sup>基于 OpenGL 的 OSG 图形库对雪景进行模拟,真实地分析了雪粒子的受力情况。刘小玲等<sup>[13]</sup>则利用 GPU 的高速运算性能模拟了大规模的雨雪场景。除此之外,也有学者基于 Vega(MultiGen-Paradigm 公司工业软件)对雪粒子系统进行模拟,杨述华等<sup>[14]</sup>基于粒子系统和 Vega 对雨雪进行了模拟。

综上,基于粒子系统的模拟要建立合适的物理模型且尽可能满足模拟的真实性,同时也需要对粒子更新的过程尽可能优化使得程序满足实时性要求。文中在分析研究粒子系统的方法的基础上,利用 VS 开发工具在 Windows 平台下进行降雪雪景模拟,建立出粒子系统和雪花系统模型,对雪花的运动进行简化,优化模拟过程,模拟出受风力影响的实际降雪雪景,并能实现对风力大小和大小雪的景观控制,引入初等函数  $F = e^x$  作为控制因子控制粒子大小使得雪粒子具有近大远小的透视投影的三维效果,最后利用 OpenGL 渲染<sup>[15]</sup>模拟出降雪雪景。

## 2 粒子系统的基本原理

Reeves W T 于 1983 年首次提出粒子系统,并成功模拟了火焰、爆炸等效果。其基本思想是用许多基本形状的微小图元粒子来描述不规则的模糊物体,这些微小粒子具有形状、大小、颜色、位置、速度矢量、生命周期等属性,对这些属性进行初始化并施加控制函数进行动态改变,体现出被模拟物体的运动的动态性。同时,为了体现出不规则模糊物体运动的随机性,对相关的粒子属性设置随机控制。所有的粒子都经过产生—活动—消亡这样的过程。鉴于粒子系统这样的特性,它能很好地模拟海浪、云、烟、火焰、雨、雪等这样随机运动的模糊自然景物。

粒子系统实现的基本步骤大致如下:

- (1)产生新粒子;
- (2)对新粒子属性进行初始化;
- (3)更新粒子属性;
- (4)删除已经超过其生命周期的粒子;
- (5)渲染绘制粒子。

根据粒子系统的算法思想,可以模拟降雪的雪景。每一片雪花作为单个粒子,赋予其相关属性,定义好其数据结构,并根据粒子系统原理的实现步骤架构出粒子系统的过程函数。

利用 C++语言编写过程函数,结合 OpenGL 图形接口对雪花进行渲染。

## 3 雪粒子系统模型

基于粒子系统算法的基本理论及其过程性,用 C++语言可以编写出粒子系统的类,并编写出粒子系统类的过程方法,如 InitializeSystem() 初始化函数、Update() 粒子运动更新函数、Render() 粒子渲染函数、Emit() 粒子发射函数,再添加一些粒子数、时间等这样的成员变量。用面向对象的思想去处理的好处有很多,在具体模拟不同物体的时候,只要继承粒子系统抽象类,根据模拟物体的具体情况去覆盖父类的虚函数即可,具有良好的扩展性。粒子一般都具有形状、大小、颜色、位置、速度矢量、生命周期这样的属性,用这样一些基本类型的数据去描述定义粒子的数据结构,根据所要模拟的物体不同,需要不同的属性描述。

### 3.1 雪粒子属性

首先分析雪粒子的运动过程,确定雪粒子的属性。雪花在下落过程中,受重力、空气浮力以及风力的影响,从视觉上呈现出左右飘动徐徐下落的状态。雪花在下落过程中,受外力影响,加速度和速度都发生了变化,导致位移也发生变化。在进行三维模拟时,将三维坐标投影到二维表面(电脑屏幕),在二维平面建立三维笛卡尔坐标系,二维平面具有  $x, y, z$  三分量。在  $x$  轴上雪花受到风力影响位置左右飘动,在  $y$  轴上雪花受到重力和空气浮力影响做下降运动,在  $z$  轴上雪花有深浅之分,表示离视点的远近,在这种透视投影中,远处的物体看上去比近处的物体更小一些,所以在  $z$  轴上的粒子深浅程度影响雪粒子的大小属性。当飘落到地面后,雪粒子融化消失,粒子死亡被删除。

综上,雪粒子的大体结构为:

```
struct particle_t
{
    vector3_t  p_pos; //粒子当前位置
    vector3_t  p_prevPos; //粒子的上一帧位置
    vector3_t  p_velocity; //粒子的速度矢量
    vector3_t  p_acceleration; //粒子的加速度
    float  p_life; //粒子的生命周期
    float  p_size; //粒子大小
    float  p_sizeDelta; //随时间的粒子大小改变量
};
```

其中定义三维矢量的数据类型 vector3\_t, 包含  $x, y, z$  三坐标轴分量参数。

### 3.2 雪粒子系统模型初始化

雪粒子系统模型构建好后,就可以编写雪景系统类了。雪景类继承粒子系统抽象父类,并对粒子系统类的虚函数如 Update() 和 Render() 进行覆盖,根据具体降雪过程来实现粒子系统各个过程函数。

雪花从空中飘向地面,粒子源定义在整个输出窗

口上部,其  $y$  值高度是固定的,为窗口高度  $m\_height$ ,在  $x$  轴和表示离视点远近的  $z$  轴上初始粒子位置则是随机的,用随机函数控制。第  $i$  个粒子的初始位置公式描述如下:

ParticleList[  $i$  ]. pos.  $y = m\_height$

ParticleList[  $i$  ]. pos.  $x = m\_origin.x + FRAND * m\_width$

ParticleList[  $i$  ]. pos.  $z = m\_origin.z + FRAND * m\_depth$

其中,FRAND 定义为  $((float) rand() - (float) rand()) / RAND\_MAX$ ,是一个  $(-1, 1)$  之间的随机数。 $m\_origin.x$  和  $m\_origin.y$  是坐标轴原点分量, $m\_width$  表示窗口宽度, $m\_depth$  则是  $z$  轴上的深浅值。

初始粒子的大小是固定的,可以设为固定值。同时对雪粒子的初始运动速度状态进行设置。在这里,为了简化之后的雪粒子更新步骤,提高系统的实时性,对每个雪粒子的初始加速度添加一个随机控制函数,使得每个粒子的初始速度状态不一致,而这个速度状态在之后的每一帧更新中不再改变。虽然是匀速状态,但是通过大量速度状态不一致的雪粒子,同样可以营造出雪花左右飘动徐徐下落的视觉效果。这样,在传统的雪粒子模拟中,需要在每一帧中更新的速度和位移信息,只需要更新位移属性即可,而且这样的简化并不影响模拟的真实性,并且提高了系统的实时性,提升了程序效率。在  $x$  轴上的初始加速度表示雪粒子受风力影响速度的改变,通过随机函数控制,不同的雪粒子在该轴上的加速度不同,通过粒子相互间速度的不同表明所受风力场的变化,同理,在  $y$  轴上加速度的不同表明所受重力和浮力的变化, $z$  轴上加速度的不同表明不同雪花飘离视点的远近。虽然在每一帧更新中速度将不再变化,但是通过整体大量的粒子与粒子之间的速度差异模拟表现出所受外力场的随机变化。第  $i$  个粒子的初始速度描述公式如下:

ParticleList[  $i$  ].  $m\_velocity.x = SNOW\_VELOCITY.x + FRAND * V\_VARIATION.x$

ParticleList[  $i$  ].  $m\_velocity.y = SNOW\_VELOCITY.y + FRAND * V\_VARIATION.y$

ParticleList[  $i$  ].  $m\_velocity.z = SNOW\_VELOCITY.z + FRAND * V\_VARIATION.z$

其中, $SNOW\_VELOCITY$  是雪粒子的初速度,是一个定义好的三维常矢量,初始速度简化为自由落体,即  $z$  轴和  $x$  轴初始速度为 0,只有竖直下落的  $y$  轴初速度。 $V\_VARIATION$  也是定义好的初始加速度三维常矢量。在  $x$  轴上加速度受风力影响随机改变,通过对它参数绝对值的大小进行设置可以表示风力强弱,通过对正负的方向设置数据表示所受风力方向。通过对  $z$  轴上

的加速度随机变化表示随时间的推移离视点的远近,从而影响粒子的大小,达到近大远小的透视投影三维效果。

粒子的初始属性设置完毕,雪粒子从窗口上部产生。由 Emit() 粒子发射函数产生,只要粒子数小于程序定义的最大粒子数,就不断循环粒子初始化函数产生粒子。这里对最大粒子数也进行可控设置,从而根据需要可以模拟大雪还是小雪的降雪场景。

### 3.3 雪粒子运动更新

粒子初始化后,就是粒子的运动更新了。对每个雪粒子个体来说其下降过程中速度将不再改变,但每个雪粒子速度状态不尽相同,通过雪粒子个体间的随机状态差异模拟出实际降雪的随机运动状态效果。通过对运动模型的简化,从而在 Update() 中省去了对速度的实时每帧控制,提高了系统的实时性,提升了程序效率,优化了模拟。位置的每帧数学模型公式为:

$$S = S_0 + \int v dt$$

第  $i$  个粒子的位置描述公式如下:

ParticleList[  $i$  ]. pos = ParticleList[  $i$  ]. pos + ParticleList[  $i$  ]. velocity \* Time

在  $z$  轴上引入指数函数  $F = e^x$  作为控制因子, $F = e^x$  在  $(-\infty, +\infty)$  上单调递增且在原点的函数值为 1,其值域恒大于 0。指数函数的数学模型特征可用来粗略控制随时间推移雪粒子离视点的近大远小变化,从而模拟出降雪的三维效果。第  $i$  个雪粒子的大小描述公式为:

$f = \text{Particle}[i].\text{pos}.z$

Particle[  $i$  ]. size =  $e^f * \text{Particle}[i].\text{size}$

当  $z$  值改变到超过一个定值范围时,即表明该粒子离视点过远或者过近,将不在显示它。而当  $y$  轴的值下降到一个定值(即程序中设定的地面  $y$  值)时,表明雪花降落到地面,删除释放雪粒子,并产生新雪粒子发射。

这样雪粒子系统对粒子系统函数 Update() 进行了覆盖。

### 3.4 雪粒子的渲染

如果要使所模拟的物体更加逼真,一般采用纹理贴图的方式。OpenGL 的纹理映射(Texture Mapping)功能支持将一些像素数据经过变换(不规则变化也支持)将其附着到多边形表面,使得模拟更加逼真。比如将真实的地板图像纹理映射到矩形上,就可以逼真地模拟出地面效果,而且在变换多边形时,多边形上的纹理图案也随之变化,也匹配透视投影的近大远小效果。OpenGL 作为目前主流的图形 API 之一,与 C 语言结合紧密,并且有强大的可移植性和渲染效果,提供了



强大的库函数进行调用,支持一维纹理、二维纹理和三维纹理。文中启用二维纹理将雪花纹理映射到雪粒子上对雪粒子进行渲染。

纹理映射的基本操作步骤如下:

- (1)图像准备:既可以在程序中生成图像,也可以读取图像文件。
- (2)生成纹理号:适用于多个纹理反复切换时,一个纹理时不需要。
- (3)设置当前纹理:切换纹理,也称为纹理绑定。
- (4)定义纹理:指定当前纹理的图像。
- (5)设置纹理参数:指定纹理的重叠方式和插值方式。
- (6)设置纹理环境参数:决定怎样使用纹理颜色,例如是否与光照色合成。
- (7)启动纹理功能,绘制场景,给出顶点的纹理坐标和几何坐标(纹理坐标可以直接参数指定,也可启用自动计算纹理坐标功能)。
- (8)退出前删除纹理。

文中采用32×32的bmp格式雪粒子纹理图片,如图1所示。

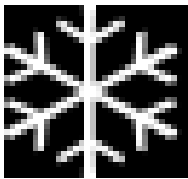


图1 雪粒子纹理

通过调用Bitmap类中LoadBitmapFileWithAlpha()函数读取图像,并将其初始化渲染到雪粒子上。

4 实验结果

实验硬件环境为主频为3.20 GHz的i5CPU,内存4 GB,显卡为ATI Radeon HD 5450的PC机,选择VS开发工具在Windows平台下,利用OpenGL图形库进行模拟。参数如表1所示。

表1 参数设置

最大粒子数	粒子初速度	粒子初始加速度
600	(0.0f, -0.25f, 0.0f)	(-0.5f, 0.5f, 0.1f)
100	(0.0f, -0.25f, 0.0f)	(0.0f, 0.5f, 0.1f)

通过表1的参数设置进行模拟,分别得到图2、图3两种视觉模拟效果。图2为受东风影响的大雪降雪雪景,图3为无风的小雪降雪雪景,均满足视觉效果,可按照参数设置自行选择所需模拟的降雪场景。

5 结束语

根据粒子系统的算法思想,结合面向对象的C++

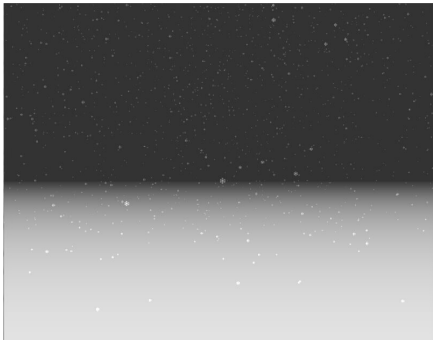


图2 大雪有风实验效果图

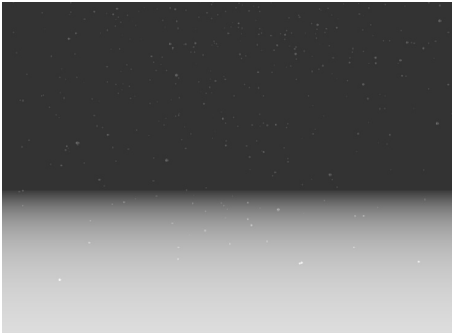


图3 小雪无风实验效果图

语言,建立出粒子系统的过程模型,分析了降雪过程,建立了雪粒子系统的三维模型,在简化雪花下降过程中的运动模型的情况下,通过控制雪粒子位移来模拟实际降雪雪景,简化了粒子更新控制步骤,并在z轴上引入控制因子控制雪粒子大小满足近大远小的透视投影效果,结合OpenGL图形接口成功渲染出了3D降雪场景。同时,对大小雪以及风力强弱实行了参数可控,可以根据需要模拟不同的降雪场景,实验结果也表明其具有逼真的三维仿真效果。

文中代码具有良好的可扩展性,在此基础上,可以将其应用于各种其他不规则物体(如烟花、雨水等)的模拟。模拟场景也可更加丰富多元化,同时考虑更加复杂的天气情况。

参考文献:

[1] Reeves W T. Particle systems—a technique for modeling a class of fuzzy objects[J]. ACM SIGGRAPH Computer Graphics,1983,17(3):359–375.

[2] Reeves W T, Blau R. Approximate and probabilistic algorithms for shading and rendering structured particle systems[J]. ACM SIGGRAPH Computer Graphics,1985,19(3):313–322.

[3] Coutinho B B,Oliveira A A F,Atencio Y P,et al. Rain scene animation through particle systems and surface flow simulation by SPH[C]//SIBGRAPI conference on graphics, patterns and images. [s. l. ]:IEEE Computer Society,2010:255–262.

[4] Kolb A,Latta L,Rezk–Salama C. Hardware–based simulation

现简单、路径长度最短的新三维移动路径规划方法。该方法适用于航空信标节点辅助三维定位,具有较好的灵活性。设计了六种典型实现路径,且能够任意组合,如 SCAN 和 S-CURVES 结合的三维移动路径等。灵活多变的移动路径给人们提供了更多的路径规划选择,较短的移动路径长度降低了航空信标节点的能量消耗。未来的工作方向是寻找到适合该类规划方法的定位方法,从而提高该类三维定位的定位精度。

#### 参考文献:

- [1] Bahl P, Padmanabhan V N. RADAR: an in-building RF-based user location and tracking system[C]//Proceedings of IEEE INFOCOM. [s. l.]: IEEE, 2000: 775-784.
- [2] Weng Y, Xiao W, Xie L. Total least squares method for robust source localization in sensor networks using TDOA measurements[J]. International Journal of Distributed Sensor Networks, 2011(3): 1063-1067.
- [3] Girod L, Estrin D. Robust range estimation using acoustic and multimodal sensing[C]//Proceedings of IEEE/RSJ international conference on the intelligent robots and systems. [s. l.]: IEEE, 2001.
- [4] Wang J, Chen Y, Fu X, et al. 3DLoc: three dimensional wireless localization toolkit[C]//IEEE 30th international conference on distributed computing systems. [s. l.]: IEEE, 2010: 30-39.
- [5] Chen Y, Liu Z, Fu X, et al. Theory underlying measurement of AOA with a rotating directional antenna[C]//Proceedings of IEEE INFOCOM. [s. l.]: IEEE, 2013: 2490-2498.
- [6] 林 玮, 陈传峰. 基于 RSSI 的无线传感器网络三角形质心定位算法[J]. 现代电子技术, 2009, 32(2): 180-182.
- [7] 唐明虎, 张长宏, 管风彪. 无线传感器网络 APIT 定位算法[J]. 微型机与应用, 2010, 29(21): 1-4.
- [8] 张 震, 闫连山, 刘江涛, 等. 基于 DV-hop 的无线传感器

网络定位算法研究[J]. 传感技术学报, 2011, 24(10): 1469-1472.

- [9] 马 震, 刘 云, 沈 波. 分布式无线传感器网络定位算法 MDS-MAP(D)[J]. 通信学报, 2008, 29(6): 57-62.
- [10] Priyantha N B, Balakrishnan H, Demaine E D, et al. Mobile-assisted localization in wireless sensor networks[C]//Proceedings of the IEEE INFOCOM. Miami, FL: IEEE, 2005.
- [11] Sichert M L, Ramadurai V. Localization of wireless sensor networks with a mobile beacon[C]//IEEE international conference on mobile ad-hoc and sensor systems. [s. l.]: IEEE, 2004: 174-183.
- [12] Kim K, Lee W. MBAL: a mobile beacon-assisted localization scheme for wireless sensor networks[C]//Proceedings of 16th international conference on computer communications and networks. [s. l.]: IEEE, 2007: 57-62.
- [13] Liu Z, Chen Y, Liu B, et al. HAWK: an unmanned mini helicopter-based aerial wireless kit for localization[C]//Proceedings of IEEE INFOCOM. [s. l.]: IEEE, 2012: 2219-2227.
- [14] Ssu K F, Ou C H, Jiau H C. Localization with mobile anchor points in wireless sensor networks[J]. IEEE Transactions on Vehicular Technology, 2005, 54(3): 1187-1197.
- [15] Koutsonikolas D, Das S M, Hu Y C. Path planning of mobile landmarks for localization in wireless sensor networks[J]. Computer Communications, 2006, 30(13): 86.
- [16] Huang R, Zaruba G V. Static path planning for mobile beacons to localize sensor networks[C]//IEEE international conference on pervasive computing and communications workshops. [s. l.]: IEEE, 2007: 323-330.
- [17] Jiang J, Han G, Xu H, et al. LMAT: localization with a mobile anchor node based on trilateration in wireless sensor networks[C]//Global telecommunications conference. [s. l.]: IEEE, 2011: 1-6.

(上接第 133 页)

- and collision detection for large particle systems[C]//Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on graphics hardware. [s. l.]: ACM, 2004: 123-131.
- [5] 袁 霞, 张玉琢. 粒子系统方法及其应用[J]. 云南师范大学学报: 自然科学版, 2003, 23(3): 14-16.
- [6] 金小进, 马尧海. 基于粒子系统的火焰模拟与优化[J]. 计算机工程与设计, 2010, 31(5): 1118-1120.
- [7] 汪继文, 胡文平, 金余峰. 基于粒子系统的 8 字动态烟花仿真[J]. 计算机仿真, 2010, 27(10): 211-214.
- [8] 潘秋羽, 毕硕本, 陆良虎, 等. 基于粒子系统三维动态云的快速仿真算法[J]. 系统仿真学报, 2014, 26(1): 85-89.
- [9] 何 亮, 巴力登. 基于粒子系统的动态雪景模拟[J]. 西北大学学报: 自然科学版, 2010, 40(4): 603-606.

- [10] 徐利明, 姜显明. 基于粒子系统与 OpenGL 的实时雨雪模拟[J]. 计算机仿真, 2005, 22(7): 242-245.
- [11] 刘金瑄, 吴频频. XNA 环境下粒子系统的飘雪仿真[J]. 西安科技大学学报, 2013, 33(4): 436-443.
- [12] 许金生, 宋万忠. 基于 OSG 粒子系统的雪景模拟[J]. 计算机工程与设计, 2012, 33(4): 1509-1513.
- [13] 刘小玲, 杨红雨, 郭虎奇. 基于 GPU 粒子系统的大规模雨雪场景实时模拟[J]. 计算机工程与设计, 2012, 33(6): 2398-2401.
- [14] 杨述华, 廖守亿, 王仕成, 等. 基于粒子系统和 Vega 的实时雨雪模拟[J]. 计算机应用, 2008, 28: 238-240.
- [15] Wright R S, Haemel N, Sellers G, et al. Open GL 超级宝典[M]. 第 5 版. 北京: 人民邮电出版社, 2012: 208-229.