动漫游戏设计
系列教程

网络游戏设计教程

房晓溪　编著

附赠光盘一张

中国水利水电出版社
www.waterpub.com.cn

动漫游戏设计
系列教程

网络游戏设计教程

房晓溪　编著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

网络游戏是动漫游戏的基础部分。

本书通过丰富的实例全面讲述了网络游戏设计制作的三维编程技术，注重克服难点，突出重点。从必需的数学物理知识和计算机编程的基本概念，到 Direct3D 的编程实践，始终将培养学生的动手能力放在首位，使学生更容易接受所学的内容。学习完本书内容，将具备完整的三维游戏编程技术理论和实践能力，能够胜任三维编程制作的职位，具备强劲的就业竞争力。

本书可以作为本科及高职高专学生的教科书，也可以作为希望从事三维编程开发的初学者的入门参考书。本书配有案例光盘，以便读者可以进行深入研究。

图书在版编目（CIP）数据

网络游戏设计教程 / 房晓溪编著. —北京：中国水利水电出版社，2008

（动漫游戏设计系列教程）

ISBN 978-7-5084-5051-3

Ⅰ. 网… Ⅱ. 房… Ⅲ. 计算机网络—游戏—应用程序—程序设计—高等学校—教材 Ⅳ. G899

中国版本图书馆 CIP 数据核字（2007）第 158534 号

书　　名	**网络游戏设计教程**
作　　者	房晓溪　编著
出版发行	中国水利水电出版社（北京市三里河路 6 号　100044） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）63202266（总机）、68331835（营销中心）、82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	北京市天竺颖华印刷厂
规　　格	889mm×1194mm　16 开本　18 印张　457 千字
版　　次	2008 年 1 月第 1 版　2008 年 1 月第 1 次印刷
印　　数	0001—4000 册
定　　价	35.00 元（含 1CD）

丛书序

动漫游戏是一种集剧情、美术、音乐、动画、程序等为一体的复合技术，一名动漫游戏专业的从业人员必须兼具软件行业专家和艺术家的创造力。从电影时代、电视时代、网络时代，到现在的移动媒体时代，动漫游戏的表现形式和内容不断发展变化，动漫游戏设计制作、经营的各个环节迅猛发展，带来了动漫游戏人才需求量的巨大缺口，尤其是创作兼技术优异的复合型设计人才更是供不应求。为推动我国动漫产业的发展、培养本土动漫专业人才，作者集多年动漫游戏设计与制作教学和著书的经验推出“动漫游戏设计系列教程”。为培养中国民族动漫、游戏人才，推动我国动漫、游戏产业快速发展贡献力量。

本套“动漫游戏设计系列教程”共有八本：

- 动漫游戏美术基础教程
- 动漫游戏美术构成教程
- 动漫游戏场景设计教程
- 动漫游戏角色设计教程
- 动漫游戏像素设计教程
- 网络游戏设计教程
- 手机动漫游戏设计教程
- 游戏引擎教程

本套动漫游戏丛书可以作为本科及高职高专学生的教科书，也可以作为希望从事动漫游戏事业的各个层次的动漫游戏爱好者的入门参考书。为方便读者学习，本套丛书大部分配有光盘，以便读者进行深入研究。

作者

2007年2月

前 言

动漫游戏产业是近年来正迅速崛起的新兴行业，已经成为中国 IT 领域增长最快的产业之一。这个产业的发展，快速地拉动了相关产业的发展，尤其对计算机产业的发展起到了推波助澜的作用。当计算机以及网络技术发展到一定程度和规模时，就不再成为动漫游戏作品表现的障碍，反而成为动漫游戏作品尽情表现的舞台。不管动漫游戏如何发展，网络游戏程序设计永远是动漫游戏技术的重要基础部分。为了让有志于从事动漫游戏行业的爱好者打好基础，本书对网络游戏设计进行系统、详细的介绍，进行开放性的思维训练，培养学生的想象力与创造力，为以后学好动漫游戏设计打下坚实的理论基础。

本书首先介绍了基础图形学知识和怎样用代数方法来对面和线建模，包括用于 3D 数学运算的 D3DX 库中包含的类和子程序，使读者对 D3D 怎样与图形硬件相互作用有了基本的认识。介绍了 2D 图片是如何存储的、页面切换和深度缓冲是如何初始化的、什么是 D3D 渲染管线。介绍了如何创建为 3D 世界进行几何描述的 2D 图形并设定一个虚拟摄像机。然后重点讲述了实用的 Direct3D 编程技术和如何使用 D3DXCreateText 函数来创建和渲染 3D 文字。进一步理解了使用渐进网格的好处以及如何用渐进网格接口 ID3DXPMesh。接着讲解了如何实现一个灵活的摄像机类，让它可以用作飞行模拟摄像机和第一人称视角摄像机；学习了对地形使用纹理和灯光的技术、粒子特性以及在 Direct3D 中怎样描述一个粒子。最后讨论了顶点和像素着色器及其高级着色语言（HLSL）。学完本书，读者能够设计并完成 3D 网络程序设计作业和作品。

本书由房晓溪编著，刘春雷、房方、纪赫男也参与了资料的收集和整理工作。在此表示衷心感谢。

作者

2007年3月

目 录

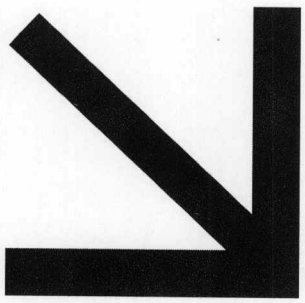
目 录

目 录

目 录

目 录

目 录



第一部分

必备的数学知识

第一部分

必备的数学知识

第 0 章
必备的数学知识

在这部分将介绍本书所要用到的数学知识。讨论的主题是向量、矩阵和相应的变换，当然还有一些有关面和线的内容。

此外，还将展示 D3DX 类中相关的数学模型和执行特殊变换的函数。

目标

- 学习向量的几何和代数性质及其 3D 计算机图形程序
- 学习矩阵以及学会使用它们来变换 3D 图形
- 学习怎样用代数方法来对面和线建模及其 3D 图形程序
- 熟悉用于 3D 数学运算的 D3DX 库中包含的类和子程序

0.1 三维空间中的向量

几何学中，用有向线段表示向量，如图 0.1 所示。向量的两个属性是长度和顶点所指的方向。因此，可以用向量来模拟既有大小又有方向的物理模型。例如，在第 14 章中要实现的粒子系统，用向量来模拟粒子的速度和加速度。其他时候，在 3D 计算机图形学中仅用向量来模拟方向。例如，人们常常想知道光线的照射方向，多边形的朝向，以及在 3D 世界中摄像机所看的方向。向量为在三维空间中表示方向提供了方便。

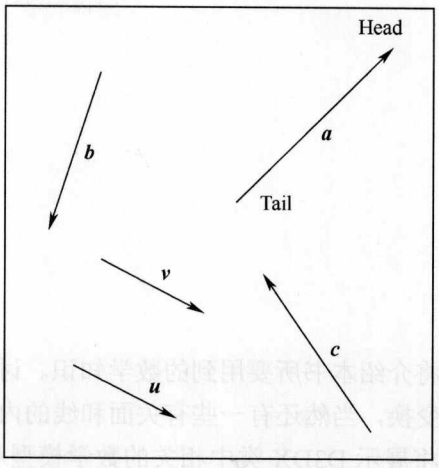


图 0.1

向量与位置无关。有同样长度和方向的两个向量是相等的，即使它们在不同的位置。观察彼此平行的两个向量，如在图 0.1 中 ***u*** 和 ***v*** 是相等的。

下面继续学习左手坐标系。图 0.2 显示的是左手坐标系和右手坐标系。两者不同的是 Z 轴的方向。在左手坐标系［见图 0.2（a）］中 Z 轴是向书的里面跑的，而右手坐标系［见图 0.2（b）］是向书外跑的。

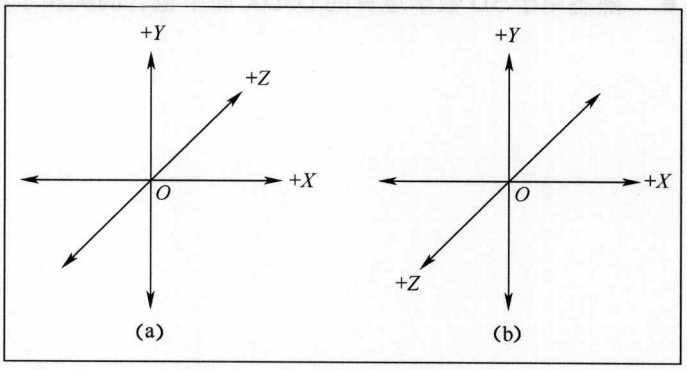


图 0.2

因为向量的位置不能改变它的性质，人们能把所有向量平移使其尾部和坐标系的原点重合。因此，当一个向量在标准位置时，可以通过头点来描述向量。图 0.3 显示的是图 0.1 中的向量在标准位置的样子。

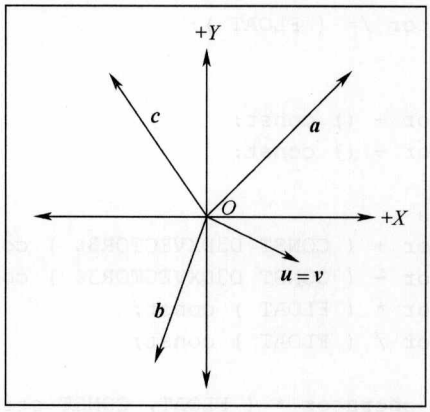


图 0.3

通常用小写字母表示一个向量，但有时也用大写字母表示。如二维、三维和四维向量可分别表示为：$\boldsymbol{u} = (u_x, u_y)$, $\boldsymbol{N} = (N_x, N_y, N_z)$, $\boldsymbol{c} = (c_x, c_y, c_z, c_w)$。现在介绍 3D 中的 4 个特殊向量，就像图 0.4 显示的。分量都是 0 的向量称为零向量，它被表示成加粗的 $\mathbf{0} = (0, 0, 0)$。接下来 3 个特殊的向量是标准向量。分别将其叫做 $\boldsymbol{i}$、$\boldsymbol{j}$ 和 $\boldsymbol{k}$ 向量，它们分别沿着坐标系的 X 轴、Y 轴和 Z 轴，并且只有 1 个单位长：$\boldsymbol{i} = (1, 0, 0)$，$\boldsymbol{j} = (0, 1, 0)$和 $\boldsymbol{k} = (0, 0, 1)$。

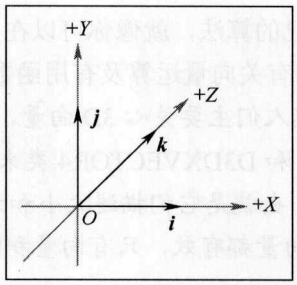


图 0.4

注意，只有 1 个单位长的向量叫做单位向量。

在 D3DX 库中，人们能用 D3DXVECTOR3 类表示三维空间中的向量。它的定义是：

```
typedef struct D3DXVECTOR3 : public D3DVECTOR
{
public:
    D3DXVECTOR3() {};
    D3DXVECTOR3( CONST FLOAT * );
    D3DXVECTOR3( CONST D3DVECTOR& );
    D3DXVECTOR3( CONST D3DXFLOAT16 * );
    D3DXVECTOR3( FLOAT x, FLOAT y, FLOAT z );

    // casting
    operator FLOAT* ();
    operator CONST FLOAT* () const;

    // assignment operators
    D3DXVECTOR3& operator += ( CONST D3DXVECTOR3& );
    D3DXVECTOR3& operator -= ( CONST D3DXVECTOR3& );
    D3DXVECTOR3& operator *= ( FLOAT );
```

```
    D3DXVECTOR3& operator /= ( FLOAT );

    // unary operators
    D3DXVECTOR3 operator + () const;
    D3DXVECTOR3 operator - () const;

    // binary operators
    D3DXVECTOR3 operator + ( CONST D3DXVECTOR3& ) const;
    D3DXVECTOR3 operator - ( CONST D3DXVECTOR3& ) const;
    D3DXVECTOR3 operator * ( FLOAT ) const;
    D3DXVECTOR3 operator / ( FLOAT ) const;

    friend D3DXVECTOR3 operator * ( FLOAT, CONST struct D3DXVECTOR3& );

    BOOL operator == ( CONST D3DXVECTOR3& ) const;
    BOOL operator != ( CONST D3DXVECTOR3& ) const;
} D3DXVECTOR3, *LPD3DXVECTOR3;
```

注意，D3DXVECTOR3 是从 D3DVECTOR 继承而来的。它的定义是：

```
typedef struct _D3DVECTOR {
    float x, y, z;
} D3DVECTOR;
```

就像数值一样，向量有它们自己的算法，就像你可以在 D3DXVECTOR3 定义中看到的数学运算，而不需要知道它们怎么使用。有关向量运算及有用函数的知识，在后续章节中会详细介绍。

注意：在 3D 图形程序中，虽然人们主要关心 3D 向量，但有时也会用到 2D 和 4D 向量。在 D3DX 库中提供了 D3DXVECTOR2 和 D3DXVECTOR4 类来分别表示 2D 和 4D 向量。不同维数的向量有着和 3D 向量一样的性质，也就是它们描述大小和方向，仅仅是在不同的维数中。所有这些向量的数学运算对于不同维数向量都有效，只有向量积除外。这些运算可以通过论述 3D 向量扩展到 2D、4D，甚至 n 维向量。

0.1.1 向量相等

几何学上，有同样方向和长度的两个向量相等。数学上，通常认为有同样维数和分量的向量相等。例如，如果 $u_x = v_x$, $u_y = v_y$，且 $u_z = v_z$，那么$(u_x, u_y, u_z) = (v_x, v_y, v_z)$。在代码中能够用“==”判断两个向量相等。

```
D3DXVECTOR u(1.0f, 0.0f, 1.0f);
D3DXVECTOR v(0.0f, 1.0f, 0.0f);
if( u == v ) return true;
```

同样地，也能用“！=”判断两个向量不相等。

```
if( u != v ) return true;
```

注意：当比较浮点数时必须加以注意。因为浮点数不是精确的，通常认为相等的两个浮点数是有细微差别的；因此，需要测试它们是否近似相等。人们定义一个常数 EPSILON，把它当做非常小的 buffer。假如两个数和 EPSILON 相差很小，就说它们近似相等。换句话说，EPSILON 让浮点数有一定的精度。接下来的实例函数用 EPSILON 比较两个浮点数相等。

```
bool Equals(float lhs, float rhs)
{
 // if lhs == rhs their difference should be zero
```