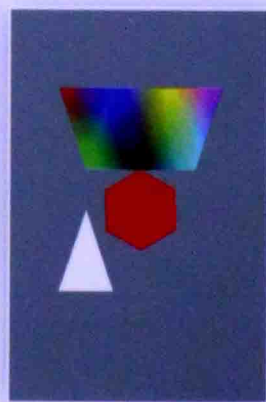


普通高等教育“动画与数字媒体专业”规划教材

iPhone 3D 游戏编程基础

袁冠远 编著

罗 林 审



清华大学出版社



普通高等教育“动画与数字媒体专业”规划教材

iPhone 3D游戏编程基础

袁冠远 编著

清华大学出版社
北京

内 容 简 介

本书主要介绍如何使用 OpenGL ES 在 IOS 平台开发交互式 3D 图形程序,重点是游戏开发。全书首先介绍必要的数学工具,然后讲解相关的 3D 概念。书中内容几乎涵盖了 OpenGL ES 中所有基本运算,例如,图元的绘制、光照、纹理、Alpha 融合、模板,以及如何使用 OpenGL ES 实现游戏中所需的技术。

本书内容深入浅出,内容广泛,实践性强,不仅可作为大学本科生教材,也适合各种游戏开发培训机构作为 OpenGL ES 编程的培训教程,对于从事 IOS 3D 游戏程序设计、可视化系统设计或其他图形应用程序开发的开发人员来说,也是一本不可多得的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

iPhone 3D 游戏编程基础 / 袁冠远编著. —北京:清华大学出版社, 2013

普通高等教育“动画与数字媒体专业”规划教材

ISBN 978-7-302-33053-0

I. ①i… II. ①袁… III. ①移动电话机—游戏—程序设计—高等学校—教材 IV. ①TN929.53

中国版本图书馆 CIP 数据核字(2013)第 145948 号

责任编辑:白立军

封面设计:常雪影

责任校对:李建庄

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京国马印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:19

字 数:476 千字

版 次:2013 年 9 月第 1 版

印 次:2013 年 9 月第 1 次印刷

印 数:1~2000

定 价:34.50 元

产品编号:049465-01

当前,移动互联网方兴未艾,被称为世界第三次科技浪潮,它拥有传播、交易、查询、地理定位等功能,强大而便捷的操作方式使它快速融入人们的生活中,成为当代人群社交活动、企业宣传不可缺少的一部分。苹果公司旗下的移动设备(如 iPhone、iPad 等)以其强大的功能、友好的操作界面成为人们的首选。苹果公司移动设备应用程序的开发也成为 IT 行业新的盈利途径。在苹果公司平台所有应用中,游戏应用最受欢迎,数量也最多,排名靠前的应用几乎都是游戏应用。因此,苹果公司移动设备游戏开发前景被非常看好。

目前,市面上关于苹果公司移动设备应用开发的书籍已经琳琅满目,但是,关于使用 OpenGL ES 技术做 3D 应用开发的书籍却不多见,中文版就更少了。本书由浅入深,循序渐进,系统地介绍了 OpenGL ES 技术的方方面面。

对于初学者,最好按照章节顺序从头到尾阅读本书,因为本书的章节编排经过了精心设计,保证了难度的循序渐进。

第 1 章首先介绍了必要的数学基础。

第 2 章讲解了怎样使用开发工具——Xcode 创建一个 OpenGL ES 项目。

第 3 章~第 7 章介绍了 OpenGL ES 的基本理论,包括基本图元的渲染、颜色、顶点格式和纹理,其中第 4 章穿插了游戏循环理论。

第 8 章~第 17 章系统地介绍了 OpenGL ES 相关的 3D 图形技术,包括渲染流水线、混合、多重纹理、光照、顶点索引、顶点缓存对象、深度缓存、模板缓存等 3D 图形技术,其中第 9 章给出了基本的 3D 物体——立方体的创建和渲染;第 13 章介绍了怎么创建更复杂的几何体,包括球体和圆柱体。

第 18 章~第 22 章介绍了几个实用技术,包括 3D 模型的加载、3D 碰撞检测、地形的绘制、天空的绘制、拾取。

此外,每章最后都提供了思考题,以提示读者对该章主要内容进行回顾,培养读者的独立思考能力、实际编程能力和创新能力。

本书用的所有实例程序均已通过调试,可直接在 Xcode 中编译运行。

由于时间关系,本书使用的部分插图来源于参考文献和互联网,在此对原作者一并表示感谢!

感谢罗林主任的特别支持,感谢我的家人对本书撰写的无私支持和付出。

由于作者知识和水平有限,错误和不妥之处在所难免,恳请读者批评指正! 本人的联系方式是 iosdesigner@qq.com。

袁冠远

2013 年 7 月于广州

目 录

Contents

- 第 1 章 数学基础 1
 - 1.1 向量 1
 - 1.1.1 向量相等 2
 - 1.1.2 向量的大小 2
 - 1.1.3 向量的规范化 2
 - 1.1.4 向量加法 3
 - 1.1.5 向量减法 3
 - 1.1.6 数与向量的乘积 3
 - 1.1.7 向量点积 3
 - 1.1.8 向量叉积 4
 - 1.2 矩阵 4
 - 1.3 变换 6
 - 1.3.1 平移变换 6
 - 1.3.2 缩放变换 7
 - 1.3.3 旋转变换 7
 - 1.3.4 矩阵复合 8
 - 1.4 思考题 8
- 第 2 章 创建 OpenGL ES 项目 9
 - 2.1 开发工具 9
 - 2.1.1 Mac 电脑 9
 - 2.1.2 iPhone 9
 - 2.1.3 Xcode 9
 - 2.1.4 Objective-C 10
 - 2.2 OpenGL ES 简介 10
 - 2.3 选择适当的 OpenGL ES 版本 11
 - 2.4 使用向导创建 OpenGL ES 项目 11
 - 2.5 从头开始创建 OpenGL ES 项目 15
 - 2.5.1 创建 Window-based Application 15
 - 2.5.2 清理 OpenGL ES 无关信息 16
 - 2.5.3 连接 OpenGL 与 Quartz 库 19

2.5.4	添加 UIView 子类——GLView	19
2.5.5	运行结果	27
2.6	应用程序设置	27
2.6.1	设置应用程序图标	27
2.6.2	处理启动画面	28
2.6.3	处理状态栏	28
2.6.4	运行结果	28
2.7	思考题	29
第3章	基本图元	30
3.1	OpenGL ES 坐标系	30
3.1.1	左手和右手坐标系	30
3.1.2	OpenGL ES 默认坐标系	30
3.2	图元	31
3.2.1	点图元	31
3.2.2	渲染点图元	31
3.2.3	线图元	36
3.2.4	渲染线图元	37
3.2.5	三角形图元	41
3.2.6	渲染三角形图元	42
3.3	思考题	46
第4章	游戏循环	47
4.1	基本的游戏循环	47
4.2	几种常见的游戏循环体	48
4.2.1	基于帧的循环体	48
4.2.2	基于时间的不定间隔循环体	49
4.2.3	基于时间的固定间隔循环体	49
4.3	IOS 游戏循环驱动器	50
4.4	IOS 游戏循环的实现	51
4.4.1	修改类声明	51
4.4.2	实现新定义的方法	52
4.5	游戏(动画)的启动与停止	54
4.6	实现简单动画	55
4.7	思考题	56
第5章	颜色	57
5.1	颜色理论	57
5.1.1	RGB 模式	57

5.1.2 颜色深度	57
5.1.3 颜色立方体模型	57
5.2 设置渲染颜色	58
5.3 顶点颜色	59
5.4 着色模式	60
5.5 思考题	61
第 6 章 顶点格式	62
6.1 顶点属性	62
6.2 顶点属性的数据类型	63
6.3 支持不同数据类型的函数	63
6.4 交错顶点数组	64
6.4.1 原理	64
6.4.2 使用交错顶点数组	65
6.5 使用结构体定义顶点数据	66
6.6 思考题	68
第 7 章 纹理	69
7.1 概述	69
7.2 纹理坐标	69
7.3 在 OpenGL ES 中使用纹理	71
7.3.1 定义顶点信息结构体	71
7.3.2 设置顶点信息	71
7.3.3 创建纹理	71
7.3.4 渲染带纹理属性图元	77
7.3.5 运行	78
7.4 纹理寻址模式	78
7.4.1 重复纹理寻址模式	79
7.4.2 箝位纹理寻址模式	79
7.4.3 分别设置 s 轴和 t 轴的纹理寻址模式	80
7.5 使用多个纹理	80
7.6 思考题	81
第 8 章 渲染流水线	82
8.1 渲染流水线概述	83
8.2 变换过程中的坐标系	83
8.3 基于 OpenGL ES 的图形变换	84
8.3.1 概述	84
8.3.2 世界变换	86

8.3.3	视图变换	86
8.3.4	投影变换	89
8.3.5	视口变换	92
8.4	思考题	92
第9章	渲染正方体	93
9.1	正方体顶点的组成	93
9.2	使用 OpenGL ES 默认变换渲染正方体	93
9.2.1	准备项目	93
9.2.2	定义顶点属性结构体	93
9.2.3	定义顶点数据	94
9.2.4	渲染	95
9.2.5	运行结果	96
9.3	设置投影矩阵以适配屏幕的宽高比	96
9.3.1	修改投影矩阵	96
9.3.2	运行结果	97
9.4	设置模型变换及结果	97
9.4.1	设置模型变换	97
9.4.2	运行结果	98
9.5	设置背面剔除及运行结果	98
9.5.1	设置背面剔除	98
9.5.2	运行结果	99
9.6	设置透视投影及运行结果	99
9.6.1	设置透视投影	99
9.6.2	运行结果	101
9.6.3	使用另一种透视相机模型	101
9.7	设置视图变换及运行结果	102
9.7.1	设置视图变换	102
9.7.2	运行结果	102
9.8	使立方体动起来	102
9.9	渲染第二个立方体	103
9.10	思考题	105
第10章	混合	106
10.1	实例：半透明效果	106
10.1.1	实验准备	106
10.1.2	渲染玻璃	107
10.1.3	运行结果	109
10.1.4	启用混合	109

10.2	混合原理	110
10.2.1	混合原理简介	110
10.2.2	混合公式	111
10.3	渲染带有 alpha 通道的纹理	112
10.4	多次纹理混合	113
10.5	alpha 测试	115
10.6	思考题	117
第 11 章	多重纹理	118
11.1	原理	118
11.1.1	纹理单元	118
11.1.2	纹理混合瀑布	119
11.1.3	纹理环境参数的设置	119
11.1.4	纹理分量	120
11.1.5	不同混合模式下纹理分量的计算公式	121
11.1.6	合并模式(Combine)下纹理分量的计算公式	121
11.1.7	操作	122
11.2	实例：颜色、纹理混合	123
11.2.1	定义顶点属性结构体	123
11.2.2	定义顶点数据	124
11.2.3	声明各四边形渲染函数	124
11.2.4	渲染函数	125
11.2.5	渲染第一个四边形	126
11.2.6	渲染第二个四边形	126
11.2.7	渲染第三个四边形	127
11.2.8	渲染第四个四边形	128
11.2.9	渲染第五个四边形	128
11.2.10	渲染第六个四边形	130
11.3	实例：精灵表动画	131
11.3.1	精灵表	131
11.3.2	图像合成	131
11.3.3	代码实现	132
11.3.4	调用代码	137
11.4	实例：光照纹理	139
11.4.1	原理	139
11.4.2	代码	139
11.5	思考题	141

第 12 章 光照	142
12.1 光照的组成	142
12.2 材质	143
12.3 法线	143
12.4 法线的计算	144
12.5 光源	145
12.6 实例：向场景中添加光照	147
12.6.1 设置顶点法向量	147
12.6.2 设置光源	149
12.6.3 立方体渲染函数	149
12.6.4 渲染各立方体	150
12.7 思考题	153
第 13 章 创建几何体	154
13.1 程序框架重构	154
13.1.1 通用函数和结构体	154
13.1.2 基类 GLViewBase	158
13.1.3 派生类 GLView	164
13.2 圆柱体建模	164
13.2.1 剖分方法	164
13.2.2 底面建模	165
13.2.3 侧面建模	166
13.2.4 圆柱体类	166
13.2.5 调用圆柱体类	170
13.3 球体建模	174
13.3.1 剖分方法	174
13.3.2 球体类	176
13.3.3 调用球体类	180
13.4 思考题	182
第 14 章 顶点索引	183
14.1 原理	183
14.2 实例：使用顶点索引建模球体	184
14.3 思考题	188
第 15 章 顶点缓存对象	189
15.1 原理	189
15.1.1 定义缓存对象名	189

15.1.2	生成缓存对象名	189
15.1.3	绑定缓存对象	189
15.1.4	传输缓存数据	190
15.1.5	渲染	190
15.2	使用顶点缓存对象优化球体模型	190
15.3	思考题	194
第 16 章	深度缓存	195
16.1	原理	195
16.1.1	准备深度缓存	196
16.1.2	清除深度缓存	197
16.1.3	启用深度测试	197
16.1.4	设置深度测试函数	197
16.1.5	渲染	198
16.2	实例：使用深度缓存	198
16.2.1	准备深度缓存	198
16.2.2	清除深度缓存	198
16.2.3	启用深度测试渲染图元	199
16.2.4	渲染结果	200
16.3	控制深度缓存更新	200
16.4	思考题	200
第 17 章	模板缓存	201
17.1	原理	201
17.1.1	准备模板缓存	202
17.1.2	清除模板缓存	203
17.1.3	启用模板测试	204
17.1.4	设置模板测试函数	204
17.1.5	设置模板更新方式	204
17.1.6	渲染	205
17.2	实例：镜面效果	205
17.2.1	概述	205
17.2.2	镜面变换	206
17.2.3	准备模板缓存	206
17.2.4	清除模板缓存	207
17.2.5	渲染	207
17.3	思考题	208



第 18 章	加载 3D 模型	209
18.1	常见 3D 模型文件格式简介	209
18.2	OBJ 文件格式简介	211
18.3	一些 OBJ 关键字	211
18.4	MTL 关键字	212
18.5	OBJ 文件实例	214
18.6	MTL 文件实例	214
18.7	WaveFrontObjLoader	215
18.7.1	WaveFrontObjLoader 的特点	215
18.7.2	WaveFrontObjLoader 的组成	215
18.7.3	WaveFrontOBJScene 类	215
18.7.4	WaveFrontOBJGroup 类	216
18.7.5	使用方式	216
18.8	实例：加载 OBJ 模型并渲染	218
18.8.1	准备工作	218
18.8.2	修改 GLView 类的定义	219
18.8.3	初始化	219
18.8.4	渲染	220
18.9	改进	221
18.9.1	改进材质类	221
18.9.2	修改 WaveFrontOBJGroup 类的定义	221
18.9.3	修改 WaveFrontOBJGroup 类的实现	222
18.9.4	为 WaveFrontOBJScene 类添加 render 方法	226
18.9.5	调用改进的类	226
18.9.6	渲染结果	228
18.10	思考题	228
第 19 章	3D 碰撞检测	229
19.1	定义包围信息	229
19.2	为 WaveFrontOBJGroup 类添加包围信息	230
19.2.1	修改 WaveFrontOBJGroup 类的定义	230
19.2.2	实现 Bounding 属性	232
19.2.3	实现 calculateBoundingInfo 方法	232
19.3	为 WaveFrontOBJScene 类添加包围信息	233
19.3.1	修改 WaveFrontOBJScene 类定义	233
19.3.2	实现 Bounding 属性	234

19.3.3	实现 calculateBoundingInfo 方法	235
19.4	渲染包围盒	236
19.4.1	概述	236
19.4.2	添加 WireFrameBox 类	236
19.4.3	修改 GLView 类	237
19.4.4	渲染结果	240
19.5	实现碰撞检测	240
19.5.1	概述	240
19.5.2	相关算法	241
19.5.3	添加 Square 类,用于绘制按钮	242
19.5.4	修改 GLView 类,实现实例	244
19.5.5	测试程序	250
19.6	扩展讨论	250
19.7	思考题	251
第 20 章	地形绘制基础	252
20.1	高度图	253
20.1.1	原理	253
20.1.2	创建高度图	253
20.1.3	加载 RAW 文件	253
20.1.4	访问高度图	254
20.2	创建地形几何信息	254
20.2.1	Terrain 类的定义	254
20.2.2	顶点的计算	256
20.2.3	索引的计算	258
20.2.4	纹理的计算	259
20.3	渲染地形	261
20.4	在地形中“行走”	261
20.5	调用地形类	265
20.6	思考题	268
第 21 章	天空绘制基础	269
21.1	原理	269
21.2	实现天空盒	270
21.2.1	创建天空盒渲染数据	270
21.2.2	创建天空盒纹理	271
21.2.3	渲染天空盒	272
21.3	调用天空盒类	273
21.4	思考题	276

第 22 章 拾取	277
22.1 概述	277
22.2 原理	278
22.2.1 射线的定义	278
22.2.2 计算拾取射线	278
22.2.3 变换物体到观察坐标系	280
22.2.4 射线/物体相交判定	280
22.2.5 封装 Picking 类	282
22.3 拾取实例	282
22.3.1 修改 Sphere 类	282
22.3.2 修改 GLView 类的定义	283
22.3.3 初始化变量	284
22.3.4 实现渲染包围球方法	285
22.3.5 修改 renderRoles 函数	285
22.3.6 实现拾取功能	287
22.3.7 修改 touchesBegan 函数	288
22.3.8 运行	288
22.4 思考题	288
参考文献	289

数学基础

本章简单介绍 3D 图形学所用到的数学工具,以便在开始编写演示程序和游戏时对所要使用的数学知识有一个基本认识。本章主要讨论向量(vector)、矩阵(matrix)、变换(transformation)等。本章所讲述的知识只是冰山一角,有关 3D 数学理论的知识远非一个章节就能讲述清楚的。如果希望学习更多的 3D 数学理论知识,可以阅读清华大学出版社出版的《3D 数学基础:图形与游戏开发》(书号:978-7-302-109464)一书。

1.1 向量

在几何学中,向量用一个有向线段来表示,如图 1.1 所示。向量的两个重要属性是长度(也称大小或者模)和方向。所以,向量在对同时具有长度和方向的物理量建模时十分有用。例如,当在地板上滚动一个球时,这个球就既有一个前进的方向;也有一个前进的速度,也就是数量大小。在 3D 图形学的某些应用场合,只用向量表示方向。例如,常常需要知道光线的走向、面的朝向以及 3D 场景中摄像机的观察方向等。

向量与位置无关,因此两个向量只要长度和方向相同,无论它们的起点是否相同,就认为二者相等。显而易见,这样的两个向量彼此平行。例如,在图 1.1 中,向量 u 和向量 v 相等。

图 1.1 中的向量可独立于具体的坐标系,因为向量在空间中并没有一个固定的位置(只与长度和方向有关)。这样,就可以将所有向量进行平移,使其起点与坐标原点重合,如图 1.2 所示。注意,由于 u 和 v 这两个向量相等,所以彼此完全重合。当一个向量的起始端与坐标原点重合时,称该向量处于标准位置。这样,就可以用向量的终点坐标来描述一个处于标准位置的向量。用于表述向量的坐标称为向量的分量。

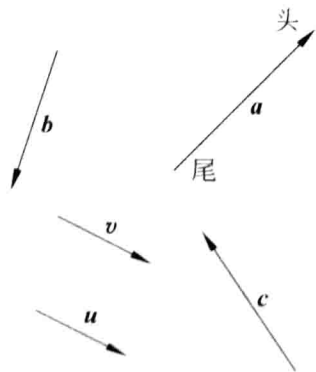


图 1.1 独立于具体坐标系的自由向量

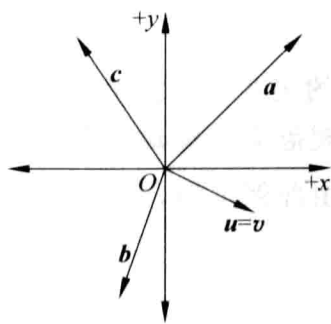


图 1.2 处于标准位置的向量

通常用小写的黑斜体字母来表示一个向量,有时也使用大写的黑斜体字母。例如,2D、3D、4D 向量分别表示为 $\mathbf{u}=(u_x, u_y)$, $\mathbf{N}=(N_x, N_y, N_z)$, $\mathbf{c}=(c_x, c_y, c_z, c_w)$ 。

1.1.1 向量相等

在几何学上,有同样方向和长度的两个向量相等。在数学上,有同样维数和分量的向量相等。例如,如果 $u_x=v_x$, $u_y=v_y$ 并且 $u_z=v_z$, 则 $(u_x, u_y, u_z)=(v_x, v_y, v_z)$ 。

注意,比较浮点数时要特别注意。由于浮点数的不精确性,通常认为应该相等的两个浮点数可能略有差别。所以,应测试它们是否近似相等。可以定义一个很小的常量——EPSILON,叫做“容差极限”。如果两个值之间的距离小于 EPSILON,就称二者近似相等。换句话说,EPSILON 给出了浮点数不精确度的公差(tolerance)。下面的函数说明了如何使用 EPSILON 来测试两个浮点数是否相等。

```
const float EPSILON=0.001f;
bool Equals(float lhs, float rhs)
{
    return fabs(lhs-rhs)<EPSILON?true : false;
}
```

1.1.2 向量的大小

在几何学中,向量的大小也称为向量的模,就是有向线段的长度。在数学上用两条双竖线将向量括起来表示向量的模。例如, $\mathbf{u}$ 的长度表示为 $\|\mathbf{u}\|$ 。

长度为 1 的向量称为单位向量。

长度是距离的度量,这就意味着可以用标准的毕达哥拉斯定理来计算它,公式如下:

$$\|\mathbf{u}\| = \sqrt{u_x^2 + u_y^2 + u_z^2}$$

例:求向量 $\mathbf{u}=(1, 2, 3)$ 和 $\mathbf{v}=(1, 1)$ 的大小。

解: $\|\mathbf{u}\| = \sqrt{1^2 + 2^2 + 3^2} = \sqrt{1+4+9} = \sqrt{14}$

$$\|\mathbf{v}\| = \sqrt{1^2 + 1^2} = \sqrt{2}$$

1.1.3 向量的规范化

向量的规范化也称为向量的归一化,就是使向量的模变为 1,即变为单位向量。可以通过将向量的每个分量除以该向量的模来实现向量的规范化,如下所示:

$$\hat{\mathbf{u}} = \frac{\mathbf{u}}{\|\mathbf{u}\|} = \left(\frac{u_x}{\|\mathbf{u}\|}, \frac{u_y}{\|\mathbf{u}\|}, \frac{u_z}{\|\mathbf{u}\|} \right)$$

我们约定在单位向量上加一个标记“^”,如 $\hat{\mathbf{u}}$ 。

例:规范化向量 $\mathbf{u}=(1, 2, 3)$ 和 $\mathbf{v}=(1, 1)$ 。

解:由前例可知, $\|\mathbf{u}\| = \sqrt{14}$, $\|\mathbf{v}\| = \sqrt{2}$, 因此:

$$\hat{\mathbf{u}} = \frac{\mathbf{u}}{\sqrt{14}} = \left(\frac{1}{\sqrt{14}}, \frac{2}{\sqrt{14}}, \frac{3}{\sqrt{14}} \right)$$

$$\hat{\mathbf{v}} = \frac{\mathbf{v}}{\sqrt{2}} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}} \right)$$