

Broadview®  
www.broadview.com.cn

Tencent | 腾讯  
INTERACTIVE ENTERTAINMENT | 互动娱乐

腾讯游戏学院  
TENCENT INSTITUTE OF GAMES

# 腾讯游戏开发精粹

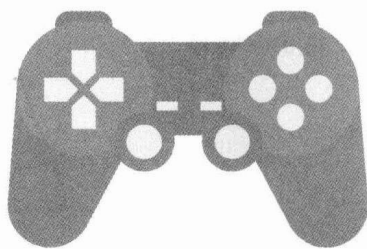
腾讯游戏 编著

 中国工信出版集团

 电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY  
http://www.phei.com.cn

# 腾讯游戏开发精粹

腾讯游戏 编著



电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

《腾讯游戏开发精粹》是腾讯游戏研发团队的技术结晶，由10多名腾讯游戏资深技术专家撰写而成，整理了团队在自主游戏研发的道路上积累沉淀的技术方案，具有较强的通用性及时效性，内容涵盖游戏脚本系统及开发工具、数学和物理、计算机图形、人工智能与后台架构等。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

### 图书在版编目（CIP）数据

腾讯游戏开发精粹/腾讯游戏编著. —北京：电子工业出版社，2019.9

ISBN 978-7-121-36602-4

I. ①腾… II. ①腾… III. ①游戏程序—程序设计—文集 IV. ①TP317.6-53

中国版本图书馆CIP数据核字（2019）第096804号

责任编辑：张春雨

印 刷：山东华立印务有限公司

装 订：山东华立印务有限公司

出版发行：电子工业出版社

北京市海淀区万寿路173信箱 邮编：100036

开 本：787×980 1/16 印张：18.25 字数：419千字

版 次：2019年9月第1版

印 次：2019年9月第1次印刷

定 价：79.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 [zlbs@phei.com.cn](mailto:zlbs@phei.com.cn)，盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

本书咨询联系方式：[tencentgamesgems@tencent.com](mailto:tencentgamesgems@tencent.com)。

# 编委会

顾问：崔晓春 夏琳 马冰冰

主编：叶劲峰

编委：郭智 刘安健 匡西尼

安柏霖 王杨军 沙鹰

审校：董磊 刘雅 陈若毅

# 推荐序

前不久，一位高中生物老师，也是我小孩同班同学的妈妈，主动问我这几年新开设的游戏设计和电子竞技专业是否值得报考，她的孩子对制作游戏非常有兴趣。听到这个咨询，我长舒了一口气：游戏正逐渐被越来越多的人认可。

我们知道，电子游戏是“第九艺术”，是各种艺术和技术能力的集大成者。记得在 10 多年前，我参与开发腾讯第一款自研游戏《QQ 幻想》的时候，随着项目的进展，我深深体会到，开发出一款高质量的 MMORPG 游戏需要很强的、综合的技术能力。比如一些技术细节：如何在最大限度限制瞬移外挂的情况下，处理弱网络环境下角色的移动拉扯和同步？怎样高效地实现基于决策树的各种 NPC 的 AI，使得 NPC 的行为更贴近自然和有趣？游戏内经济系统的平衡，各种角色之间能力的平衡，还有上百万用户同时在线时的稳定性、热更新、热切换……诸如此类，在当时的技术背景下，是颇为严峻的挑战。

我们也知道，知识只有分享才更有价值。站在巨人的肩膀上，会看得更远；有了前辈的经验加持，也会成长得更快。《腾讯游戏开发精粹》这本书汇集了腾讯游戏在游戏开发中的部分精华，从客户端到服务器端，从物理引擎到工具链，从图形学到 AI，各领域均有被验证过的解决方案呈现。腾讯是一家具有高度社会责任感的公司，它愿意将这些能力和经验无私奉献给大家，为行业的发展贡献自己的一点力量。互相学习，一起进步，是我们的希望。

我们还知道，电子游戏的特征之一是互动，在饭桌、在房间、在地铁……我们都可以看到各种玩游戏情景的开心愉快，它已经融入我们的日常生活之中。娱乐是人的天性，但让生活更美好，是我们每一个游戏从业者的使命。从虚拟世界到现实生活，再从现实生活到虚拟世界，用技术的手段来改变生活，未来就靠你了！

崔晓春

腾讯互动娱乐 公共研发运营体系负责人

# 编者序

游戏开发对于一般软件开发者来说，总像蒙上了一层神秘面纱。这可能是由多个原因造成的。首先，游戏开发的技术范畴比较广，一些技术如计算机图形学、物理模拟、实时网络同步等比较少应用在一般软件开发中。其次，游戏开发属于创意工业，对各类型游戏的需求有很多区别，不少技术没有形成标准，各家的技术方案、工作流程等也会有不少差异。最后，公司之间甚至公司之内也可能有技术壁垒，影响知识和技术的流通。这些情况不利于有兴趣的朋友进入此行业，从业者的进步也会受限，长远影响行业的发展，难以面对全球的激烈竞争。

编者在 20 世纪 90 年代的香港，互联网未普及之前，只能通过 BBS 收集一些国外“漂流”过来的游戏开发技术文档，例如《德军总部 3D》的三维室内场景渲染及纹理贴图技术、如何使用非标准的 Mode X 去做 VGA 256 色双缓冲区渲染等。在那个资讯匮乏的年代，每次遇到新技术的解密文档，编者都兴奋得如获至宝。

而在国内做游戏开发的“老鸟”，大概都会翻过千禧年代的《游戏编程精粹 (Game Programming Gems)》系列丛书。这套丛书影响了一整代的开发者，让我们能一窥世界各地游戏开发者的各种秘技，解决在游戏开发中遇到的各种共同问题，同时可以激发灵感，研发比书中更好的解决方案。

进入互联网信息爆炸的年代，我们能在网上接触无数的博客、问答等信息，可以更快速地知悉各种新技术。但同时，网上信息相对于传统出版来说，通常较为零散，品质参差不齐。从业者也基于保密原因，不会随便公开一些游戏开发中使用到的新技术。

本书受《游戏编程精粹》系列丛书的启发，希望鼓励腾讯游戏的工程师与业界同行分享一些实际应用在游戏里的技术，与行业共享。通过内部审核及编辑等机制，尽量筛选可对外公开、高品质的文章，也保证技术具有一定的通用性及时效性。对国内业界而言，希望这本书能成为一小步，促进更开放的未来，提升整体技术水平。

本书从提案到出版长达一年半的时间，除了依靠各位作者在忙碌的开发任务中抽空撰文，还必须感谢腾讯游戏学院院长夏琳女士的大力支持，也要感谢腾讯游戏学院的董磊、刘雅和陈

若毅使项目成功推进。我也衷心感谢本书的编委郭智、刘安健、匡西尼、安柏霖、王杨军和沙鹰（排名不分先后），他们都是腾讯游戏各个部门的技术专家，悉心为文章的内容把关。也非常感谢电子工业出版社的张春雨和葛娜协助出版事宜。

最后，希望本书能对读者有所帮助，如有任何意见请不吝通过邮件反馈给我们：  
[tencentgamesgems@tencent.com](mailto:tencentgamesgems@tencent.com)，期望在续篇再见。

叶劲峰

《腾讯游戏开发精粹》主编

腾讯互动娱乐 魔方工作室群技术总监

# 目 录

## 第一部分 游戏数学

第 1 章 基于 SDF 的摇杆移动	2
摘要	2
1.1 引言	3
1.2 有号距离场 (SDF)	3
1.3 利用栅格数据预计算 SDF	4
1.4 SDF 的碰撞检测与碰撞响应	5
1.5 避免往返	8
1.6 利用多边形数据预计算 SDF	9
1.7 其他需求	10
1.7.1 如何将角色从障碍区域中移出	10
1.7.2 角色不能越过障碍物的远距离移动	11
1.8 动态障碍物	12
1.9 AI 寻路	14
1.10 动态地图	14
1.11 总结	17
参考文献	17
第 2 章 高性能的定点数实现方案	18
摘要	18
2.1 引言	18
2.1.1 浮点数简介	18
2.1.2 32 位浮点数 (单精度) 表示原理	19
2.2 基于整数的二进制表示的定点数原理	19



2.2.1	32 位定点数表示原理	19
2.2.2	64 位定点数表示原理	20
2.3	定点数的四则运算	21
2.3.1	加法与减法	22
2.3.2	乘法	22
2.3.3	除法	23
2.4	定点数开方与超越函数实现方法	23
2.4.1	多项式拟合	24
2.4.2	正弦/余弦函数	25
2.4.3	指数函数	26
2.4.4	对数函数	27
2.4.5	开方运算	27
2.4.6	开方求倒数	28
2.4.7	为什么不用查表法	30
2.5	定点数的误差对比与性能测试	30
2.5.1	超越函数及开方的误差测试	30
2.5.2	性能测试	30
2.6	总结	31
	参考文献	31

## 第二部分 游戏物理

第 3 章	一种高效的弧长参数化路径系统	34
	摘要	34
3.1	引言	34
3.2	端点间二次样条的构建	35
3.3	路径的构建	38
3.4	曲线的弧长参数化	39
3.5	曲线上的简单运动	42
3.5.1	跑动	42
3.5.2	跳跃	43

---

3.5.3 相邻路径的切换	44
3.5.4 曲线上的旋转插值	45
3.6 总结	46
参考文献	46
<b>第 4 章  船的物理模拟及同步设计</b>	<b>47</b>
摘要	47
4.1  浮力系统	48
4.1.1  浮力	48
4.1.2  升力	52
4.1.3  拉力	52
4.1.4  拍击力	53
4.1.5  阻力上限	54
4.2  引擎系统	55
4.2.1  移动、转向模拟	55
4.2.2  向心力计算	56
4.3  Entity-Component 及同步概览	56
4.4  浮力系统物理更新机制	57
4.5  总结	59
参考文献	59
<b>第 5 章  3D 游戏碰撞之体素内存、效率优化</b>	<b>60</b>
摘要	60
5.1  背景介绍	60
5.2  体素生成	62
5.3  体素内存优化	62
5.3.1  体素合并的原理	62
5.3.2  体素合并的算法	64
5.3.3  地面处理	65
5.3.4  水的处理	66
5.3.5  范围控制	67
5.3.6  内存自我管理	67

5.3.7	体素内存优化算法的效果	68
5.3.8	体素效率优化	69
5.4	NavMesh 生成	69
5.4.1	体素生成 NavMesh	69
5.4.2	获取地面高度	70
5.4.3	后台阻挡图	71
5.4.4	前台优先级 NavMesh	71
5.4.5	锯齿	72
5.5	行走、轻功、摄像机碰撞	73
5.5.1	行走	73
5.5.2	轻功	75
5.5.3	摄像机碰撞	75
	参考文献	76

### 第三部分 计算机图形

第 6 章	移动端体育类写实模型优化	78
	摘要	78
6.1	引言	79
6.2	方案设计思路	79
6.2.1	角色统一与差异元素分析	79
6.2.2	角色表现=人体+服饰	80
6.2.3	角色资源整合	83
6.2.4	资源制作与实现	84
6.3	具体实现	92
6.3.1	实现流程	92
6.3.2	CPU 逻辑	93
6.3.3	GPU 渲染	97
6.4	效果收益、性能分析和结语	97
6.4.1	方案优劣势	98
6.4.2	方案补充	99

6.4.3 应用场景	99
参考文献	100
第 7 章 大规模 3D 模型数据的优化压缩与精细渐进加载	101
摘要	101
7.1 引言	102
7.2 顶点数据优化	102
7.2.1 顶点数据合并去重	103
7.2.2 索引数据合并	104
7.2.3 顶点数据排序	104
7.2.4 子网格的拆分与合并	105
7.2.5 顶点数据编码压缩	105
7.3 有利于渐进加载的数据组织方式	112
7.4 总结	113
参考文献	114

## 第四部分 人工智能及后台架构

第 8 章 游戏 AI 开发框架组件 behaviac 和元编程	116
摘要	116
8.1 behaviac 的工作原理	117
8.1.1 类型信息	117
8.1.2 什么是行为树	118
8.1.3 例子 1	119
8.1.4 执行说明	119
8.1.5 进阶	120
8.1.6 例子 2	120
8.1.7 再进阶	123
8.1.8 总结	123
8.2 元编程在 behaviac 中的应用	125
8.2.1 模板特化	126
8.2.2 加载中的特例化	126

8.2.3 运行中的特例化	129
<b>第9章 跳点搜索算法的效率、内存、路径优化方法</b>	<b>131</b>
摘要	131
9.1 引言	132
9.2 JPS 算法	133
9.2.1 算法介绍	133
9.2.2 A*算法流程	133
9.2.3 JPS 算法流程	135
9.2.4 JPS 算法的“两个定义、三个规则”	135
9.2.5 算法举例	137
9.3 JPS 算法优化	138
9.3.1 JPS 效率优化算法	138
9.3.2 JPS 内存优化	144
9.3.3 路径优化	145
9.4 GPPC 比赛解读	146
9.4.1 GPPC 比赛与地图数据集	146
9.4.2 GPPC 的评价体系	148
9.4.3 GPPC 参赛算法及其比较	150
参考文献	151
<b>第10章 优化 MMORPG 开发效率及性能的有限多线程模型</b>	<b>152</b>
摘要	152
10.1 引言	152
10.1.1 多进程单线程模型	153
10.1.2 单进程多线程模型	153
10.1.3 单进程单线程模型	153
10.2 有限多线程模型	154
10.3 使用 OpenMP 框架快速实现有限多线程模型	156
10.4 控制多线程逻辑代码	158
10.5 异步化解决数据安全问题	159
10.6 对“不安全”访问的防范	160

10.7 拆解大锁.....	161
10.8 其他建议.....	163
参考文献.....	164

## 第五部分 游戏脚本系统

第 11 章 Lua 翻译工具——C#转 Lua.....	166
摘要.....	166
11.1 设计初衷.....	166
11.2 实现原理.....	167
11.2.1 参考对比行业内类似的解决方案.....	167
11.2.2 翻译原理.....	168
11.2.3 翻译流程.....	168
11.3 翻译示例.....	170
11.4 实现细节.....	174
11.4.1 连续赋值.....	175
11.4.2 switch.....	175
11.4.3 continue.....	176
11.4.4 不定参数.....	177
11.4.5 条件表达式.....	178
11.5 运行性能.....	179
11.6 TKLua 翻译蓝图.....	179
11.6.1 类关系.....	180
11.6.2 类成员.....	180
11.6.3 方法体.....	181
11.7 发展方向.....	182
11.8 总结.....	184
参考文献.....	185
第 12 章 Unreal Engine 4 集成 Lua.....	186
摘要.....	186
12.1 引言.....	186

12.2	UE4 元信息	187
12.2.1	介绍	187
12.2.2	Lua 通过元信息与 UE4 交互	189
12.2.3	读写成员变量	189
12.2.4	函数调用	190
12.2.5	C++调用 Lua	191
12.2.6	小结	192
12.3	通过模板元编程生成“胶水”代码	192
12.3.1	接口设计	193
12.3.2	实现	195
12.3.3	读写成员变量	197
12.3.4	引用类型	198
12.3.5	导出函数	199
12.3.6	默认实参	200
12.3.7	默认生成的函数	202
12.3.8	C++调用 Lua	203
12.3.9	小结	203
12.4	优化	203
12.4.1	UObject 指针与 Table	203
12.4.2	结构体	204
12.4.3	运行时热加载	205

## 第六部分 开发工具

第 13 章	使用 FASTBuild 助力 Unreal Engine 4	208
	摘要	208
13.1	引言	209
13.2	UE4 分布式工具	209
13.2.1	Derived Data Cache (DDC)	209
13.2.2	Swarm	210
13.2.3	IncrediBuild	210

13.2.4	FASTBuild	211
13.3	在 Windows 系统下搭建 FASTBuild 工作环境	213
13.3.1	网络架构	213
13.3.2	搭建基本环境	214
13.3.3	可用性验证	215
13.4	使用 FASTBuild 分布式编译 UE4 代码和项目代码	219
13.4.1	准备工作	219
13.4.2	部署多机 FASTBuild 环境	220
13.4.3	编译 UE4 代码及对比测试	220
13.4.4	优化 FASTBuild	224
13.4.5	再次测试分布式编译 UE4 代码	227
13.5	“秒”编 UE4 着色器	228
13.5.1	准备工作	229
13.5.2	大规模着色器编译测试	237
13.5.3	材质编辑器内着色器编译测试	240
13.6	总结	243
第 14 章	一种高效的帧同步全过程日志输出方案	244
摘要		244
14.1	引言	244
14.2	帧同步的基础理论	245
14.2.1	基本原理	245
14.2.2	系统抽象	246
14.3	本方案最终解决的问题	247
14.4	全日志的自动插入	250
14.4.1	在函数第一行代码之前自动插入日志代码	250
14.4.2	处理手动插入的日志代码	251
14.4.3	对每行日志代码进行唯一编码	251
14.4.4	构建版本	252
14.4.5	整体工具流程及代码清单	252
14.4.6	为什么不采用 IL 注入	256
14.5	运行时的日志收集	256



14.5.1	整体业务流程	256
14.5.2	高效的存储格式	258
14.5.3	高性能的日志输出	259
14.5.4	正确选择合适的校验算法	259
14.6	导出可读性日志信息	260
14.7	本方案思路的可移植性	260
14.8	总结	261
<b>第 15 章</b>	<b>基于解析符号表，使用注入的方式进行 Profiler 采样的技术</b>	<b>262</b>
	摘要	262
15.1	进行测量之前的准备工作	263
15.1.1	注入的简单例子	263
15.1.2	注入额外的代码	264
15.1.3	注入的注意事项	265
15.2	性能的测量	267
15.2.1	时间的统计方法	267
15.2.2	针对函数的采样	268
15.2.3	测量实战	273
15.3	总结	276