文本复制检测报告单(全文标明引文)

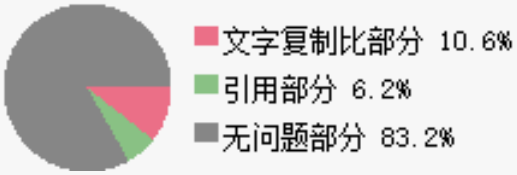
№:ADBD2017R_20110928222448201705172302575093708924 29　　检测时间:2017-05-17 23:02:57

检测文献:　50060943843028650谭鹏
作者:　谭鹏
检测范围:　中国学术期刊网络出版总库
中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库
中国重要会议论文全文数据库
中国重要报纸全文数据库
中国专利全文数据库
互联网资源(包含贴吧等论坛资源)
英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)
港澳台学术文献库
优先出版文献库
互联网文档资源
图书资源
CNKI大成编客-原创作品库
个人比对库
时间范围:　1900-01-01至2017-05-17

检测结果

总文字复制比：16.8%　　跨语言检测结果：-
去除引用文献复制比：10.6%　　去除本人已发表文献复制比：16.8%
单篇最大文字复制比：4.4%

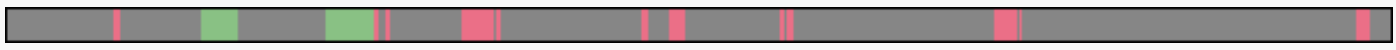
重复字数：[2353]　　总段落数：[2]
总字数：[14033]　　疑似段落数：[2]
单篇最大重复字数：[623]　　前部重合字数：[447]
疑似段落最大重合字数：[1896]　　后部重合字数：[1906]
疑似段落最小重合字数：[457]



指　标：□疑似剽窃观点　☑疑似剽窃文字表述　□疑似自我剽窃　□一稿多投　□疑似整体剽窃　□过度引用　□重复发表

表格：0　　脚注与尾注：2

19.6%（1896）　50060943843028650谭鹏_第1部分（总9670字）
10.5%（457）　50060943843028650谭鹏_第2部分（总4363字）



（注释：无问题部分　文字复制比部分　引用部分）

1. 50060943843028650谭鹏_第1部分　　总字数：9670

相似文献列表　文字复制比：19.6%(1896)　疑似剽窃观点：(0)

1	基于漏斗的实时VBR视频最短路径平滑算法 袁俊杰;徐小良; -《计算机系统应用》- 2010-07-15	5.1%（489） 是否引证：是
2	计算机游戏设计原理以及游戏引擎的设计思想 穆俊; -《硅谷》- 2014-02-08	3.9%（374） 是否引证：是
3	A*算法寻找路径 潘海波; -《黑龙江科技信息》- 2009-06-25	3.8%（366） 是否引证：否
4	移动Flash流媒体关键技术的研究 袁俊杰(导师：徐小良) -《杭州电子科技大学硕士论文》- 2009-12-01	3.3%（322） 是否引证：否
5	基于BREW的手机游戏的研究和实现	3.1%（296）

	孙亭南(导师：邵时;潘荫荣) -《华东师范大学硕士论文》- 2006-10-01	是否引证：否
6	基于VR技术的学生群体紧急疏散模拟研究	3.0%（293）
	江湉湉(导师：张际平) -《华东师范大学硕士论文》- 2010-09-01	是否引证：否
7	基于XNA的游戏设计与实现	3.0%（286）
	黄进(导师：宋成;马迪芳) -《北京交通大学硕士论文》- 2011-06-01	是否引证：否
8	3DXML标准在协同装配中的研究与应用	2.6%（251）
	曹翔(导师：黄志球) -《南京航空航天大学硕士论文》- 2007-01-01	是否引证：否
9	直线的DDA算法的C实现（opengl实现） - zhouyelihua - 博客频道 - CSDN.NET	2.3%（218）
	-《网络（http://blog.csdn.net）》- 2013	是否引证：否
10	人工智能寻路算法在电子游戏中的研究和应用	2.2%（212）
	詹海波(导师：沈刚) -《华中科技大学硕士论文》- 2006-10-01	是否引证：否
11	人工智能寻路算法在电子游戏中的研究和应用 - docin.com豆丁网	2.2%（212）
	-《互联网文档资源（http://www.docin.com）》- 2012	是否引证：否
12	深入浅出A-Star算法 - 豆丁网	2.1%（206）
	-《互联网文档资源（http://www.docin.com）》- 2015	是否引证：否
13	A*寻路初探 - 自由天地 -	2.0%（194）
	-《网络（http://blog.donews.c）》- 2012	是否引证：否
14	A寻路算法.doc	2.0%（194）
	-《互联网文档资源（http://max.book118.c）》- 2015	是否引证：否
15	A*寻路算法初探 - 思月行云 - CSDN博客	1.9%（187）
	-《网络（http://blog.csdn.net）》- 2011	是否引证：否
16	AStar寻径算法小结_leiii	1.9%（187）
	-《网络（http://blog.sina.com）》- 2016	是否引证：否
17	3D游戏引擎设计及其关键技术-百度文库	1.8%（176）
	-《互联网文档资源（http://wenku.baidu.c）》- 2012	是否引证：否
18	蚁群算法-百度文库	1.8%（176）
	-《互联网文档资源（http://wenku.baidu.c）》- 2012	是否引证：否
19	基于HTML5与Node.js的游戏引擎的设计与开发	1.8%（176）
	徐佳宾(导师：曹士坷) -《南京邮电大学硕士论文》- 2016-11-18	是否引证：否
20	群体动画中自主智能体的行为控制	1.7%（163）
	杨延(导师：张建中) -《电子科技大学硕士论文》- 2009-06-01	是否引证：否
21	一款基于ARPG的“仙剑”手机游戏的设计与实现	1.1%（109）
	蔡丽鸳(导师：孙涌) -《苏州大学硕士论文》- 2016-09-01	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

使用A*平滑消除3D游戏中的画面抖动

专业：计算机科学与技术

学生：谭鹏指导教师：陈杰

摘要

[摘要] 21世纪是属于互联网、属于计算机的一个世纪，也是属于电子游戏的一个世纪。在这个世纪刚刚开始的这么短短十几年里，计算机的性能和互联网得到了飞速的发展。电子游戏也以爆炸般的速度在发展。而3D游戏由于其对现实模拟的更为逼真、更能给予玩家一种冲击感，3D游戏站在了游戏发展的浪头上，引领者游戏的风骚。一大波的开发者正在涌入到3D游戏的开发阵营中来。可是3D游戏的开发过程中，很可能会遇到一些麻烦奇怪的问题，而画面抖动就是其中一种让很多开发者百思不得其解的奇怪问题。本文分析了3D游戏中画面抖动的原因，解释了3D游戏的一些基本概念，也为抖动提供了一种解决方案和方案的原理。旨在帮助新入3D游戏开发的游戏开发者能够轻松的跨过这道门槛。本文消除抖动所使用的A*算法为寻路的经典算法，本文将该算法与一些已知的算法进行组合、改造，形成了本文的算法。算法可靠性可以得到保证。同时，本文对于3D变换的部分提供了数学理论证明，保证可靠性。

关键词：3D游戏，平滑算法，画面抖动，A*算法

Abstract

[Abstract]The 21st century is the century that belongs to the Internet, belongs to the computer for a century, and also belong to the video games. In the just begun so short ten years of this century, the computer's performance and the Internet has been rapidly developed. The video games are also developing at an explosive rate. The 3D games, because of its realistic simulation of more realistic, more able to give players a sense of impact, are in the on the waves of game development, and are the leader coquettish of the game. A big wave of developers are pouring into the 3D game development camp. But in the process of development of 3D games, it is likely to encounter some strange problems, and the

screen shaking is one of the strange questions that a lot of developers hard to understand. This paper analyzes the causes of shaking in 3D games, explains some of the basic concepts of 3D games, and provides a solution for shaking. Designed to help the new 3D game developers can easily cross this threshold. In this paper, we use the classical algorithm of pathfinding -- A * algorithm for erasing shaking. In this paper, we combine the algorithm with some known algorithms to transform and form the algorithm. Algorithm reliability can be guaranteed. At the same time, this article provides a mathematical proof of the 3D transformation part to ensure reliability.

Keywords: 3D Gaming, smoothing algorithm ,shaking, A * algorithm

一、引言

1.1 背景

在*计算机游戏中，模拟游戏（Sim）、射击游戏（shooter）角色扮演游戏（RPG）是较早并且具有较强代表性的计算机游戏，这些游戏的设计、产生和应用为计算机游戏整体设计水平的提升奠定的良好的基础。在这些早期的游戏设计过程中，游戏设计者通过让游戏玩家利用一种非结构化的方式来进行游戏体验，这种游戏体验往往具有较为明确的目标，例如角色扮演游戏则是以剧情的进展、而射击游戏是以关卡的演进为明确的目标，但是在明确的设计结构下游戏玩家的动作是随意的，即在这种游戏设计原理下游戏玩家能够进行具有探索性的游戏方式，这种游戏方式的存在极大的提升了计算机游戏的可玩性并且为接下来计算机游戏的设计水平提升提供了重要助力。即游戏玩家通过自主探索在实际上可以有效反应出游戏设计的基本结构，这种结构在提升游戏系统明确性的同时可以较为集中的反应游戏设计接下来的进步方向。*[1]

随着个人电脑性能的提升、普通用户智能手机的普及，使得现如今人们所持有的设备的性能越来越高，3D游戏所需要的额外性能很容易就可以满足。所以3D游戏市场变得异常火热。但是在游戏开发过程中，作为一个开发者，我们时常会遇到游戏画面抖动的情况。如果任由这种情况存在，会降低玩家的体验，是对游戏可玩性的一大损害。

1.2 意义

可玩性是如同字面一样，说的是游戏让玩家玩下去的性质。对于电子游戏这一个特殊的软件来说，可玩性是其核心内容。可玩性主要受游戏类型与游戏设计、游戏操作与响应、学习曲线和精通难度的影响。

3D游戏相较于2D游戏来说，从图形学角度来讲，多了一个z轴，使得3D游戏是在空间的层次上进行展示，而2D是在平面的层次上进行显示。2D游戏现实只需要将屏幕的一部分直接投射到屏幕上就可以，但是3D不同，由于3D使得游戏有远近、深浅的概念，简单的投射已经不可行。3D游戏采用了人眼的模式，来展现游戏世界。如果游戏的过程中，游戏画面抖动，将会使玩家感觉自身在抖动，影响玩家在游戏中的判断，降低游戏的可玩性。

由于可玩性是电子游戏的核心，消除游戏中画面抖动势在必行。采用本文的方法可以消除开发过程中出现的画面抖动，提高游戏的体验，提高游戏的可玩性，增加用户粘度，减少开发负担。

1.3 现状

在当前，3D游戏除了一部分卡牌类游戏以外，其余的游戏基本上都需要涉及到画面的移动、人物的移动等。在这些情况下，出现画面抖动的情况非常高。应用消除抖动是势在必行的。而当前，在3D游戏开发过程中中，遇到了画面抖动，一般采用的是漏斗平滑算法来进行消除。

而一般所采用的漏斗平滑算法在八格子寻路时效率不好，并不适用于某些类型的游戏。

1.4 其他抖动消除算法简述

漏斗平滑算法是3D游戏中，消除抖动时最常见的一种平滑算法。

漏斗平滑算法过程是基于边界的。每次判断下一个点是否在漏斗中，再会构建新的漏斗继续进行判断，不在则把之前在的点设为新的起点来构建漏斗，重复进行。但是这个算法是基于边界和点的。在某些情况下会效率比较低下，不适用于某些游戏。

*如图1所示，O(t)和U(t)构成的区域可以看作是一个单调多边形。我们用Π(s,w) 来表示多边形内点 s至点 w 的最短路径，那么 Π(s,e) 即为所求最短路径。显然，Π(s,e)只可能与多边形的凹点(例如 v)相交，不可能与凸点(例如 z)相交。因此，计算最短路径只需要考虑凹点。假设已知 Π (u,s) = [u,s] 及 Π (s,w) =[s,v,w]，那么漏斗就是由 Π(u,s) 以及 Π(s,w) 构成的*V形结构，用 Fuw *表示，而 s (V 形底端)为漏斗的底。Fuw是由最初漏斗经过数次更新后生成的，最初的漏斗由起始点 s 与 O(t)、U(t) 的第一个凹点的连线构成的，如图中虚实线所示。*

图1

*在现有漏斗 Fuw 的基础上，计算出漏斗底至下一凹点 x 的最短路径 Π(s,x)，那么漏斗就更新为 Fux= Π(u,s) + Π(s,x)，依次类推计算后续凹点直至终点 e，即可得到 F *e，而 F *e 的下半部分即为所求的 Π(s,e)，在 Fuw 的基础上，计算 Π(s,x) 的方法如下。首先，如果 s 到 x 的直线不与 Fuw 相交，那么 Π(s,*x) 就是直线 sx，否则，在 Fux 中寻找一个点 vi，使得 vix 与 Fux相切。用公式可以表示为：

tg(vi-1vi)≥tg(vix)≥tg(vivi+1)

其中 tg 表示线段的斜率， v i-1 v i 以及 v i v i+1 为 Fuw 中v i 的前后线段。找到点 v i 后连接v ix，那么 Π(s,x) = [s,…vi,x][2]。

图2

以图2来说明漏斗的过程：

1．起点与两边边界最开始的点连线，这两条线构成了一个漏斗。

2．分别测试两边边界上的下一个点与起点的联系是否在前面的点生成的漏斗中。如果都在，则执行3；若一条在一条不在，则执行4；若都不在则执行5。

3．以新的两条线原起点构成新的漏斗，重复2。

4．以在漏斗中的那条新线为新边界与另一边的原边界和原起点构成新的漏斗，重复2；

5．如果都不在，则以两条线中的最短的一条中的端点为新的起点，这条线段为路径中的一段加入到结果集合中。重复1。

图2-A中起点与最近的两个点组成漏斗。2-B中下侧边界的下一点与起点连线，该线在漏斗中。图2-C中上边界的下一点与起点连线，连线在原漏斗中。则以这两条线为新的漏斗边。2-D图中，下边界下一点与起点连线，线在新的漏斗中，2-E图中，上边界下一点与起点连线，不在漏斗中，舍弃。下边界新线与原上边界线组成新的漏斗。2-F图中下边界下一点与起点连线，不在漏斗中，舍弃，上边界下一点与起点连线，不在漏斗中，舍弃。选择原漏斗边界中最短一条的端点为新起点。如图2-G。重新开始漏斗处理。

1.5 工作与创新点

由于漏斗平滑算法是基于边界的平滑算法，在某些游戏中，算法效率较低，并不适用，而A*算法一般用于最短路径的查找中，不能直接应用与平滑中，于是通过对传统A*算法的拓展，实现了一个平滑算法，用于消除3D游戏中出现的画面抖动。

本文所使用的算法基于A*算法，本文中运用得数学知识，均有数学证明，保证了算法的可靠性。同时，本算法基于A*算法，方便移植到实际项目中。

二、相关算法

2.1 A*算法

A*算法原本是一种求解最短路径直接搜索方法，是一种启发式搜索算法。

公式表示为：

f(n) = g(n)+h(n)

其中，f(n) 是从初始状态经由状态n到目标状态的代价估计，g(n) 是在状态空间中从初始状态到状态n的实际代价，h(n) 是从状态n到目标状态的最佳路径的估计代价。

A*算法的具体过程为：

1．把起始格添加到开启列表。

2．寻找开启列表中F值最低的格子，这就是当前格。将这个格子加入到关闭列表。

3. 对于当前格相邻的每一个格子，如果这一格不可通过或者已经在关闭列表中，那么就略过这一格；如果这一格不在开启列表中，把这一格添加进去开启列表，并且把当前格作为这一格的父节点，记录这一格的F,G,和H的值；如果这一格已经在开启列表中，则检查这一格的G值是否更低，如果是，就把这一格的父节点改成当前格，并且重新计算这一格的G和F值；如果你把目标格添加进了关闭列表，则说明路径被找到，或者没有找到目标格，但这个时候开启列表已经空了则说明路径不存在则结束这一步执行4。遍历完所有邻格后，执行2。

4. 保存路径。从目标格开始，沿着每一格的父节点移动直到回到起始格。这就是你的路径。

2.2 弗洛伊德路径平滑算法

弗洛伊德算法是解决任意两点间的最短路径的一种算法，能够正确解决有向图或存在负权的图的最短路径问题，同时也常常被用于计算有向图的传递闭包。弗洛伊德算法的时间复杂度为O(N3)，空间复杂度为O(N2)。

弗洛伊德算法的主要思路如下：从任意节点A到任意节点B的最短路径存在2种情况，要么是直接从A到B，又或者是从A通过了若干个节点X之后再到B。所以，我们假设d(AB)为节点A到节点B之间存在的最短路径的距离，对于节点A与节点B之间的每一个节点X，d(AX)是节点A到节点X的距离，d(XB)是节点X到节点B的距离，判断是否存在d(AX) + d(XB) < d(AB)，如果存在，则可以证明从A到X再到B的路径比A直接到B的路径短，我们便设置d(AB) = d(AX) + d(XB)，这样一来，当我们遍历完所有节点X，d(AB)中记录的便是节点A到节点B的之间的最短路径的距离。

其算法描述：

1.对于目标节点，两点直达的距离是已知的，则首先将这两点的距离设置为该距离，如果不能直接地从一点到达另一点，那么将两点的距离设为无穷大。

2.对于包含目标节点在内的每一对顶点 u 和 v，看看是否存在一个顶点w使得从顶点u 到顶点w再到顶点v的路径长度比已知的路径更短。如果存在就将这条路径设为最短路径和并将距离更新。

3.遍历完所有点后，所剩下的最短路径即为所求最短路径，最短路径长度即为该路径的距离。

弗洛伊德路径平滑算法是一种在平滑算法内部应用了弗洛伊德算法的平滑算法。

弗洛伊德路径平滑算法试用的是通过A*寻路算法得到的最短路径后进行平滑，它的主要步骤只有两步：一、如果路径中存在相邻的点共线，那么就将共线的点进行合并，只留下端点；二、去掉路径中存在的多余拐点，如果能够直接到达，就直接到达，不经过这些拐点。这个过程如下图所示：

图3

去掉共线点

图4

去掉多余拐点

图5

可以看到，使用弗洛伊德路径平滑算法处理后的路径表现的与我们所期望的一致，路径中存在的节点大大减少，而且路径不存在多余的拐点，能够直接到的就直接到了。

判断几点共线的主要原理在于，如果在平面直角坐标系中存在三点A(x1,y1), B(x2,y2), C(x3.y3),这三点的横坐标和纵坐标满足以下关系：

x2-x1=x3-x2 y2-y1=y3-y2

则说明A,B,C三点共线，那么只需要保留处于端点的两点，除去中间的点。对于空间直角坐标系，原理类似，A(x1,y1,z1),B(x2,y2,z2),C(x3,y3,z3),当存在以下关系时，A、B、C三点共线：

x2-x1=x3-x2 y2-y1=y3-y2 z2-z1=z3-z2

这种情况下也可以消除这三点中的中间点。

对于多余拐点的消除，原来主要是：若路径中存在的节点一系列A,B,C,D,E,F,G，且节点A和节点G直接地连线所经过的节点中没有一个节点是不可通过的节点，那我们可以称节点A与节点G之间不存在障碍物。如果两格节点之间不存在障碍物，那么这两个两点间的所有其他节点都是可以被除去的。例如上述A,B,C,D,E,F,G这些节点，假设节点A与节点G之间不存在障碍物，那么我们将A与G之间的B,C,D,E,F节点全部都除去，最终形成的路径将只剩下A与G两个节点。

2.3 DDA算法

数值微分法即DDA法(Digital Differential Analyzer)，是一种基于直线的微分方程来生成直线的方法。

算法描述：

设(x1，y1)和(x2，y2)分别为所求直线的起点和终点坐标，由直线的微分方程得：

dx/dy = (y2-y1)/(x2-x1) = m = 直线斜率 = ?y/?x (1)

那我们就可通过计算x方向上的增量?x引起y的改变来生成所需的直线。

xi+1 = xi+?x (2)

yi+1 = yi+?x * m (3)

我们也可通过计算由y方向的增量?y引起x的改变来生成直线，但是这比改变x方向的增量少见：

yi+1 = yi +?y (4)

xi+1 = xi + ?y /m (5)

我们x2-x1与y2-y1中较大者作为步进方向(假设x2-x1较大)，取该方向上的增量为一个象素单位(?x =1)，然后利用式(1)计算另一个方向的增量(?y =?x*m =m)。通过递推公式(2)至(5)，把每次计算出的(xi+1，yi+1)经取整后送到显示器输出，则得到扫描转换后的直线。

之所以取x2 - x1和y2 - y1中较大者作为步进方向，是考虑沿着线段分布的象素应均匀。

三、抖动消除

在了解抖动发生的原因之前这之前，我们需要首先了解3D游戏的发展之路以及一些基本概念。

“游戏”一词泛指棋类游戏例如象棋和《大富翁》；纸牌游戏，例如梭哈和二十一点；赌场游戏例如轮盘和老虎机；军事战争游戏、计算机游戏、孩子们一起玩耍的游戏。在计算机的语境下，“游戏”一词会使我们在脑海中浮现出一个虚拟世界，玩家可以控制人物、动物或玩具。

绝大部分游戏是软实时互动基于代理计算机模拟的例子。

在电子游戏中，会用数学方法来为真实世界的子集建模，从而使这些模型在计算机中运行。显然，这些模型只能是显示或者想象世界的简化或者近似版本，因此，数学模型是现实或者虚拟世界的模拟。

基于代理模拟是指，模拟中多个独立的实体（称作代理）一起互动。

所有的互动游戏都是时间性模拟的，即游戏世界是动态的——随着游戏事件和故事的展开，游戏的状态随着时间改变。游戏也必须响应人类玩家的输入，这些输入是游戏本身不可预知的，这也说明游戏是互动时间性模拟的，大多数游戏会实时回应用户输入，这即为互动实时模拟。

软实时是指即使错过期限却不会造成灾难性的后果。所有游戏都是软实时的。

模拟虚拟世界需要用到很多数学模型。数学模型分为解析式和数值式。例如，一个刚体因为地心引力而以恒定加速度落下，其分析式数学模型可写为：

y(t)= gt2 +v0t+y0

分析式模型可为其自变量设任何值来求值。例如上式，给予初始条件v0和y0、常量g，就能设任何时间t来求y(t)的值。在电子游戏中，用户的输入是不能预知的，因此不能预期对整个游戏完全适用分析式建模。

刚体受地心引力落下的数值式模型可写为：

y(t+ t)=F(y(t),y`(t),y``(t),...)

也就是说，该刚体在（t+ t)未来事件的高度，可以用目前的高度、高度的第一导数，高度的第二导数及目前的时间t为参数的函数来表示。为实现数值式模拟，通常需要不断重复的计算，以决定每个离散时间的系统状态。游戏也是如此运作的，一个主游戏循环不断执行，在循环的每次迭代中，多个游戏系统，例如人工智能、游戏逻辑、物理模拟等，就会有机会计算或者更

新其下一个离散时间的状态。这些结果最后可渲染成图形显示、发出声效或者输出至其他设备。

游戏引擎这个术语在20世纪90年代中期形成，这与第一人称射击游戏如id software公司的《DOOM》有关。《DOOM》将其软件构架划分为核心软件组件（如三维图形渲染系统、碰撞检测系统和音频系统等）、美术资产、游戏世界、构成玩家游戏体验的游戏规则。这样的划分非常有价值，另一个开发商取得了这样的游戏的授权之后，只需要制作新的美术、关卡布局、武器、角色、游戏规则等，对引擎软件做出很少的修改，就可以把游戏打造成新产品。

现在主流、常见的商业引擎有：Value公司的Source引擎，Epic的Unreal引擎，以及在移动端很常见的跨平台商业引擎Unity3D，Cocos2dx等。另外还有很多游戏公司有自己专用的私有引擎。

一般认为，首个三维第一人称射击游戏是《德军司令部》。这款游戏有美国的id software于1992年制作，他引领游戏产业进入到了令人们兴奋的领域。Id software又相继开发了《DOOM》、《Quake》等游戏。随着20多年的发展，3D游戏已经变得非常常见，基本上已经是随处可见。现如今的游戏大多数是3D游戏。

3.1 抖动与抖动发生的原因

在3D游戏开发的过程中，当在游戏内对游戏中的对象进行移动、转向时，游戏的画面会发生剧烈的抖动，会使玩家晕眩，影响玩家体验。

要明白抖动的发生，首先需要明白电子游戏是怎么运作的。首先，需要认识到电子游戏也仅仅是一个可编译和运行的程序。就像其他的程序一样，main函数也是它第一个被执行的函数。游戏的结构如下图所示：

图6

上图包含了五种状态，电子游戏最重要的三种状态在循环中互相连接，这个循环通常称作游戏循环。游戏循环式进行玩家输入，更新游戏内部数据，更新显示的地方。

初始化是尤其开始的地方，通常包含了启动动画以及一个包含了音量、画质等选项的选项菜单，在这个过程中会进行一些必要参数的设定，以及申请内存，设置堆栈，检测游戏环境以及加载显示驱动（比如包含集显和独立显卡的机器来加载独立显卡）。在这些都已经设置完毕后，玩家选择了开始游戏，则跳转到下一个状态。

玩家输入是对玩家所使用的输入设备例如键盘、鼠标、游戏手柄等进行监听，对玩家的输入的信息进行采集，并且在游戏内部中以游戏能够处理的形式进行存储。供后一个状态使用。

更新游戏内部是游戏的心脏，游戏根据玩家的输入，来改变游戏中的设定值，例如对玩家操作的人物进行移动，玩家游戏中的敌人的运动与反应，同时准备好显示所需要的所有数据与图像。

更新显示是将计算后的画面投射到屏幕上显示出来。一般来说，你可以选择对每个物体挨个在屏幕上绘制或者说按照例行办法，先把所有的对象计算好，然后进行一次性的绘制。在屏幕上进行绘制是一个花费时间比较多的过程。所以一次性进行绘制将会有比较好的性能。

结束游戏是玩家选择退出游戏或者玩家通关之后的一种正常状态。一般会有一个结束动画或者提示语来告诉玩家游戏已经结束了。同时也会进行数据的保存，释放内存，消除堆栈等工作。

摄像机指的不是现实社会中用来照相的那个摄像机，但是两者发挥的作用基本上是一样的。电子游戏中的摄像机用另外的一个词来说是“视角”，也就是看游戏世界的角度。展现在玩家屏幕上的画面便是通过摄像机的“照”出来的画面。

由于玩家的画面是由摄像机照出来的，那么画面抖动会有两个原因：游戏世界在抖动或者是摄像机在抖动。 游戏世界我们没有进行直接操作，那么按道理来说不会发生抖动，这个可能性就可以被排除掉。那么抖动发生的原因在于摄像机在抖动。

开发游戏时，当对游戏内部的对象进行移动、旋转操作时时，由于游戏本身的设定，摄像机也会跟随着玩家移动的对象进行移动、旋转，这在酷跑类游戏中很常见。由于游戏是帧驱动的，游戏循环每执行一次，游戏就播放一帧。每一帧中都会执行摄像机位置的变换，使之看向所需要看向地方，摄像机不停变换位置，而且是无规则的。不规则的位置变化会引起抖动。

3.2 抖动消除原理

既然画面的抖动是由于摄像机的抖动造成的，那么，我们要消除画面抖动，就只需要消除摄像机的抖动就可以了。画面抖动都发生在画面移动时即摄像机朝向或者位置发生变化时。

指　标
疑似剽窃文字表述

1. 公式表示为：
   f(n) = g(n)+h(n)
   其中， f(n) 是从初始状态经由状态n到目标状态的代价估计，g(n) 是在状态空间中从初始状态到状态n的实际代价，h(n) 是从状态n到目标状态的最佳路径的估计代价。
2. 过程为：
   1．把起始格添加到开启列表。
   2．寻找开启列表中F值最低的格子，这就是当前格。将这个格子加入到关闭列表。
   3. 对于当前格相邻的每一个格子，如果这一格不可通过或者已经在关闭列表中，那么就略过这一格；如果这一格不在开启列表中，把这一格添加进去开启列表，并且把当前格作为这一格的父节点，记录这一格的F,G,和H的值；如果这一格

已经在开启列表中，

3. 如果是，就把这一格的父节点改成当前格，并且重新计算这一格的G和F值；如果你把目标格添加进了关闭列表，则说明路径被找到，或者没有找到目标格，但这个时候开启列表已经空了则说明路径不存在
4. 保存路径。从目标格开始，沿着每一格的父节点移动直到回到起始格。这就是你的路径。

2.2 弗洛伊德路径平滑算法

5. 之所以取x2 - x1和y2 - y1中较大者作为步进方向，是考虑沿着线段分布的象素应均匀。
6. 年代中期形成，这与第一人称射击游戏如id software公司的《DOOM》有关。
7. 开发商取得了这样的游戏的授权之后，只需要制作新的美术、关卡布局、武器、角色、游戏规则等，对引擎软件做出很少的修改，就可以把游戏打造成新产品。

脚注和尾注

1. [1] 穆俊. 计算机游戏设计原理以及游戏引擎的设计思想[J]. 硅谷,2014,(03):49+98..
2. [2] 袁俊杰,徐小良. 基于漏斗的实时VBR视频最短路径平滑算法[J]. 计算机系统应用,2010,(07):42-46..

2. 50060943843028650谭鹏_第2部分

总字数：4363

相似文献列表　文字复制比：10.5%(457)　疑似剽窃观点：(0)

1	基于BREW的手机游戏的研究和实现 孙亭南(导师：邵时;潘荫荣) -《华东师范大学硕士论文》- 2006-10-01	6.6%（287） 是否引证：否
2	基于VR技术的学生群体紧急疏散模拟研究 江湉湉(导师：张际平) -《华东师范大学硕士论文》- 2010-09-01	6.3%（274） 是否引证：否
3	A*算法寻找路径 潘海波; -《黑龙江科技信息》- 2009-06-25	5.9%（257） 是否引证：否
4	基于XNA的游戏设计与实现 黄进(导师：宋成;马迪芳) -《北京交通大学硕士论文》- 2011-06-01	5.6%（244） 是否引证：否
5	深入浅出A-Star算法 - 豆丁网 -《互联网文档资源（http://www.docin.com）》- 2015	4.7%（206） 是否引证：否
6	第三部分 搜索原理(Part1,2,3,final) - 豆丁网 -《互联网文档资源（http://www.docin.com）》- 2016	4.4%（194） 是否引证：否
7	A*寻路初探 - 自由天地 - -《网络（http://blog.donews.c）》- 2012	4.3%（187） 是否引证：否
8	A*寻路算法初探 - 思月行云 - CSDN博客 -《网络（http://blog.csdn.net）》- 2011	4.3%（187） 是否引证：否
9	AStar寻径算法小结_leiii -《网络（http://blog.sina.com）》- 2016	4.3%（187） 是否引证：否
10	A寻路算法.doc -《互联网文档资源（http://max.book118.c）》- 2015	4.3%（187） 是否引证：否
11	3D游戏引擎设计及其关键技术-百度文库 -《互联网文档资源（http://wenku.baidu.c）》- 2012	3.9%（169） 是否引证：否
12	蚁群算法-百度文库 -《互联网文档资源（http://wenku.baidu.c）》- 2012	3.9%（169） 是否引证：否
13	群体动画中自主智能体的行为控制 杨延(导师：张建中) -《电子科技大学硕士论文》- 2009-06-01	3.7%（163） 是否引证：否
14	人工智能寻路算法在电子游戏中的研究和应用 詹海波(导师：沈刚) -《华中科技大学硕士论文》- 2006-10-01	3.6%（159） 是否引证：否
15	人工智能寻路算法在电子游戏中的研究和应用 - docin.com豆丁网 -《互联网文档资源（http://www.docin.com）》- 2012	3.6%（159） 是否引证：否
16	个人信用信息法律问题研究 李佳成(导师：蓝寿荣) -《南昌大学硕士论文》- 2016-05-28	3.6%（156） 是否引证：否
17	情感管理视野下高校学生公寓管理研究 沈东益(导师：宋艳) -《吉林大学硕士论文》- 2014-04-01	1.9%（85） 是否引证：否
18	紫花苜蓿耐盐材料的鉴定及胚状体耐盐性诱变研究 张学云(导师：袁庆华) -《中国农业科学院硕士论文》- 2013-05-01	1.9%（84） 是否引证：否
19	基于DSP数字EFO烧球系统的研究与设计 梁付根(导师：赖华) -《昆明理工大学硕士论文》- 2016-03-01	1.1%（49） 是否引证：否
20	蒙古族婚礼服饰研究 贺俊杰(导师：苏和平) -《中央民族大学硕士论文》- 2016-06-15	1.1%（47） 是否引证：否

由于现今的游戏通常都是帧驱动的游戏，我们需要在游戏循环每次执行的时候，让摄像机位置、朝向的改变变得平滑也就是使得摄像机看向的位置移动变得平滑，这样，在游戏的过程中，我们所看到的画面是在进行平滑的变换，就如同我们自身走动一般。

抖动消除有一个前提，时间上要是平滑的。也就是说要么是每一帧执行所用的时间是一样那要么就处理成一样的效果。现代游戏是帧驱动的，我们不能保证每一帧所执行的时间是一致的，但是我们可以获得每一帧执行所消耗的时间，在处理时，我们以消耗的时间为影响因子乘如，这样便实现了时间上的平滑。

再者是路径上的平滑，平滑无非两种，一种是延迟，一种是根据以前的速度进行加速度计算，然速度变化变慢。

本文中消除抖动的平滑算法分为两部分：直行时平滑部分和转动时平滑部分。分别对应在直行时位移所导致的抖动和转向时旋转导致的抖动。

3.3 算法实现

3.3.1不转向时平滑算法

在直行时，基本上不会涉及到摄像机朝向的变化，也基本上不会导致摄像机旋转。那么我们可以使摄像机朝向某一个确定的方向，控制摄像机沿着平滑的路径，以平滑的速度运动。

对于平滑的路径，我们使用A*寻路找到这样的路径。采用以下步骤：

1．把起始格添加到开启列表。

2．寻找开启列表中F值最低的格子，这就是当前格。将这个格子加入到关闭列表。

3. 对于当前格相邻的每一个格子，如果这一格不可通过或者已经在关闭列表中，那么就略过这一格；如果这一格不在开启列表中，把这一格添加进去开启列表，并且把当前格作为这一格的父节点，记录这一格的F,G,和H的值；如果这一格已经在开启列表中，则检查这一格的G值是否更低，如果是，就把这一格的父节点改成当前格，并且重新计算这一格的G和F值；如果你把目标格添加进了关闭列表，则说明路径被找到，或者没有找到目标格，但这个时候开启列表已经空了则说明路径不存在则结束这一步执行4。遍历完所有邻格后，执行2。

4. 保存路径。从目标格开始，沿着每一格的父节点移动直到回到起始格。这就是你的路径。

该过程的位代码为：

开启列表 ← 起始格
关闭列表 ← {}
repeat
当前格 ← min(F(开启列表))
foreach 相邻格(当前格) do
if 相邻格(当前格) in 关闭列表 or 相邻格(当前格) no access then
skip
end
if 相邻格(当前格) not in 开启列表 then
开启列表 ← 相邻格(当前格)
end
if 相邻格（当前格） in 开启列表 then
if G(P(相邻格(当前格)))< G(当前格) then
当前格 ← 父节点(相邻格(当前格))
renew(F(相邻格(当前格)))
renew(G(相邻格(当前格)))
end
end
end
until 目标格 in 关闭列表 or 开启列表 = {}
save path

使用A*寻路算法之后，我们得到了一条摄像机移动的路径。这个路径是由一系列的点组成的。对于这些点，可能有重复的点,我们需要使用弗洛伊德路径平滑算法来对结果进行简化、平滑。

算法过程：

对于初识连续的点A1, A2, A3… An中的某个点Ai (x i, y i, z i),如果前一点Ai-1 (x i-1,y i-1,zi-1)和后一点Ai+1 (xi+1, yi+1, zi+1),如果xi-xi-1=zi+1-zi且yi-yi-1=yi+1-yi且zi-zi-1=zi+1-zi那么这三点共线，则在点集中除去Ai点。将所有共线点处理后得到新的点集A1 ,A2, A3,.. Ax,如果A j，A j+1, … A j+ m中的点都不是不可移动点且A j 到 A j +m之间没有障碍物，则删除A j+1到A j+m-1的所有点。

伪代码如下：

初识点集 ← {A1,A2,A3….An}
i ← 1
repeat
if Ai,Ai-1,Ai+1共线 then
delete Ai
end
i ← i+1
until i = n
i ← 1
repeat
if Ai,Ai-1,Ai+1都可以移动 and Ai, Ai+1间没有障碍 then
delete Ai
end
until
通过上述方法，得到了平滑的路径。

得到了平滑的路径之后，我们需要对速度进行处理。由于游戏是一帧一帧进行执行，每一帧所耗费的时间可能不一样，因此，摄像机移动的距离不能简单的设置为一段距离，应当以一段距离乘以一帧执行的时间。距离选取不应太长，否则会出现移动不连贯，出现跳帧的现象。

如果在运动的过程中有速度变化，则需要有一个连续的速度变化过程，不能瞬间就变化到某个速度去。例如启动运动的时候，游戏中物体会由静变化到动，这在这个过程中，需要有一个加速的过程。可以采用一个恒定的加速度。

3.3.2 转向时的平滑算法

2D游戏中由于只有x,y两根坐标轴，只会在平面上进行移动，不会存在转向的问题。但是在3D游戏之中，很有可能会有转向。在这种情况下，摄像机的朝向会发生改变，朝向变化也需要进行平滑。

转向时平滑需要考虑的是自然的转向不是就在原地为圆心做圆周运动，而是以原地以外的某个点为圆心做圆周运动，因此，在转向时，需要进行处理。通过圆的二维平面方程：

x-a2+y-b2= r2

得到圆的轨迹。摄像机的朝向与摄像机坐标与圆心的方向相同。

此时运动的长度可以根据角度来。相同时间转动的角度应该是确定的。则根据转过的角度θ来得到每一帧处于的点和朝向。设某点为(x, y, z),为了简化处理，假设圆心为(0,0,0),半径为r，在平面Ax + By = 0上画圆，单位时间转动的角度值为α，则新一点的坐标值为

x1=x±r1-cosα?t

y1=y±r1-sinα?t

z1=z

其中+号还是-号取决于方向。

对于一般情况下转向，就只简单的做一下数学上的推导，在实际游戏开发中，一般都会选择Ax+By=0这个平面为基准面。

在三维空间中，设圆的圆心为(x0, y0, z0),圆的半径为r,圆的法向量为(α, β, γ)，则以(x0, y0, z0)为圆心以r为半径的球面方程为

x-x02+y-y02+z-z02=r2

圆所在的平面方程为：

x-x0,y-y0,z-z0?α,β,γ=0

联立两式得到空间中圆的方程：

x-x02+y-y02+z-z02-αx-x0-βy-y0-γy-y0=r2

设之前的点为(x, y, z)，单位时间内转动的角度值为μ，则有方程：

x-x0,y-y0,z-z0?x1-x0,y1-y0,z1-z0=r2cosμ?t

x1-x0,y1-y0,z1-z0?α,β,γ=0

x1-x02+y1-y02+z1-z02=r2

联立上式，即可得到新的点的坐标。

四、实验与分析

4.1 实验结果

我们在所开发的demo为一个酷跑类型的游戏，游戏过程中，玩家控制游戏中的角色一直前进，中途可能会转向会有障碍物。摄像机在游戏过程中会一直跟随着玩家来一起运动。游戏demo使用的是Unity3D引擎来制作。

在使用平滑算法之前，游戏在匀速直行的时候，基本不会发生抖动。但是如果速度发生变化例如从静到动的时候或者有转向时或者突然横向平移时，游戏会发生剧烈的抖动。摄像机表现失控。可能会发生旋转。

将平滑算法加入到摄像机的控制脚本中之后，在匀速直线移动的时候，运动路径为直线 ，与之前一致，游戏表现正常，与加入平滑算法前的行为保持一致。在加速直线运动时，比如由静到动时，路径为直线，与之前一致，摄像机有一个加速过程。在加速过程中玩家可以很清晰的感觉到物体的速度加快，画面中物体接近摄像机的速度越来越快，过程中未发生抖动，摄像机失控的现象。转向时，摄像机现在会以转向处的外的某个点为圆心，以匀速做圆周运动转过，同时摄像机的朝向也逐步变化。屏幕上会展示出沿途的物体，转到了合适的方向之后，摄像机又开始执行，继续前进，等待玩家的指令输入。发生横向平移的时候，摄像机先水平方向的运动以一定加速度减慢，会同时在横向提供一个加速度使之加速，使之运动起来，朝向不发生改变。在这个过程之中，摄像机移动平缓，画面稳定，没有发生抖动、跳帧的情况。

4.2 分析与讨论

在应用算法前，匀速之前运动的时候，由于不存在速度上和路径上的不平滑，所以游戏过程中表现非常正常，没有出险抖动、跳帧、失控等不正常现象。但在加速、减数时，由于此时速度发生了变化，由于游戏没有对速度进行处理、平滑，所以速度的突然变化会导致游戏画面的抖动、跳跃。当有横向移动的时候，之前会直接瞬间横向移动，缺少了速度变化的过程，路径也不平滑，摄像机照出来的场景变化大、频繁，导致了抖动和跳帧。当转动时，最开始的是瞬间改变朝向，这样会发生非常剧烈的抖动，还可能会是摄像机失控。后面改为以原地为圆心旋转。但是这使得摄像机照出的画面显得很生硬，不自然。与现实差距较大。

在应用平滑算法之后，匀速直线运动由于路径、速度都是平滑的，结果与之前一样是在预料之中的，也是符合情景的。在加速、减速的直线运动时，之前由于速度不是平滑的，所以对速度进行平滑处理后，摄像机运动的速度有了一个平滑的变化，游戏画面变化不突兀、没有发生抖动也是符合预期结果的。转向时改变后游戏变化连续，未出现抖动现象，符合预期结果。横向移动时，可以感觉到前后速度与横向速度的变化，未出现抖动，符合预期结果。

五、总结与展望

通过应用平滑算法，使得游戏中摄像机能够平滑、稳定的进行移动、旋转，使得游戏的画面能够连续、稳定的进行变换，不会出现游戏画面的抖动。

本文所使用的平滑算法，在匀速直线运动时不会导致异常，在转向、变速运动、横向移动时，能够很好的解决画面抖动与画面不连续的问题，说明了本文的算法是较为可靠的。使能够应用到开发过程中的。

六、致谢

在本文的编写过程中，感谢我的指导老师陈杰老师的细心指导，老师在从设计拟题、文章的编写、文献的选取都提供了很大的帮助，在过程中每周五都会在望江实验室进行交流、了解进度、对不清楚的部分进行指导、说明。这篇文章的成形，离不开老师的心血。在此，谨向导师表示崇高的敬意和衷心的感谢!

通过这篇论文的撰写，使我能够等系统、全面的学习有关游戏开发的最新的、最先进的前沿理论知识，这对于我今后的工作和和我以后自身的发展，无疑是不可多得的宝贵财富。由于本人理论水平比较有限，论文中的有些观点和归纳和阐述难免有疏漏和不足的地方，欢迎老师和专家们指正。

7.本报告单仅对您所选择比对资源范围内检测结果负责。

8.Email：amlc@cnki.net http://e.weibo.com/u/3194559873 http://t.qq.com/CNKI_kycx