使用A\*平滑消除3D游戏中的画面抖动

作者：谭鹏

学院：计算机学院

专业：计算机科学与技术

导师：陈杰

# 目录

[目录 2](#_Toc30418)

[一、引言 3](#_Toc140)

[1.1 背景 3](#_Toc12158)

[1.2 意义 3](#_Toc9214)

[1.3 现状 3](#_Toc20839)

[1.4 其他抖动消除算法简述 3](#_Toc15260)

[1.5 工作与创新点 3](#_Toc11297)

[二、相关算法 4](#_Toc31191)

[2.1 A\*算法 4](#_Toc27860)

[2.2 \*\*算法 4](#_Toc4529)

[2.3 \*\*算法 4](#_Toc13070)

[三、原因与原理 5](#_Toc17095)

[3.1 抖动与抖动发生的原因 5](#_Toc3356)

[3.2 抖动消除原理 5](#_Toc720)

[3.3 算法实现 5](#_Toc8200)

[四、实验与分析 6](#_Toc6168)

[4.1 实验结果 6](#_Toc8370)

[4.2 结果分析 6](#_Toc21686)

[五、总结与展望 7](#_Toc11942)

# 摘要

# 一、引言

## 1.1 背景

“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、发出声效或者书橱至其他设备。

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。

1992年，3D realms公司发布了一部具有划时代意义的游戏——《德军司令部》。这部游戏开创了第一人称射击游戏的先河，但更重要的是，这部游戏创造性的在x轴和y轴的基础上再添加了一根z轴，正式的将游戏带入了3D的时代。现如今，3D游戏已经发展了25年了。现今的游戏大都是3D游戏。

3D游戏开发中，经常会遇到画面抖动的问题。画面抖动，将极大的影响玩家的游戏体验。

## 1.2 意义

画面抖动将会导致玩家在游戏过程中体验，降低游戏的可玩性，降低游戏的留存。消除画面抖动是不能避免的。

## 1.3 现状

除了A\*算法以外，还有。但

## 1.4 其他抖动消除算法简述

## 1.5 工作与创新点

# 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，重复如下的工作：

a) 寻找开启列表中F值最低的格子。我们称它为当前格。

b) 把它切换到关闭列表。

c) 对相邻的格中的每一个？

\* 如果它不可通过或者已经在关闭列表中，略过它。反之如下。

\* 如果它不在开启列表中，把它添加进去。把当前格作为这一格的父节点。记录这一格的F,G,和H值。

\* 如果它已经在开启列表中，用G值为参考检查新的路径是否更好。更低的G值意味着更好的路径。如果是这样，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。如果你保持你的开启列表按F值排序，改变之后你可能需要重新对开启列表排序。

d) 停止，当你

\* 把目标格添加进了关闭列表，这时候路径被找到，或者

\* 没有找到目标格，开启列表已经空了。这时候，路径不存在。

3.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 \*\*算法

## 2.3 \*\*算法

# 三、原因与原理

## 3.1 抖动与抖动发生的原因

抖动，指画面不断四处移动，不连贯。

游戏主要由游戏引擎

在介绍抖动发生的原因之前，我们首先要先了解3D游戏的基本原理。

3D游戏之中，由于地图或者场景过大，超出了屏幕的显示范围。

## 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。由于现今的游戏通常都是帧驱动的游戏，

## 3.3 算法实现

# 四、实验与分析

## 4.1 实验结果

## 4.2 结果分析

# 五、总结与展望