

**计算机科学与通信工程学院**

**《面向对象程序设计课程设计报告》**

|  |  |
| --- | --- |
| 课题名称 | 基于Qt平台的火柴人打羽毛球游戏的复现 |
| 学生姓名 | 张家旗 |
| 学号 | 3230602065 |
| 专业班级 | 计算机科学与技术2303 |
| 指导教师 | 曹汉清 |
| 成绩 |  |

2023年 7 月 7日

[一、 系统需求分析和概要设计 1](#_Toc25403)

[1.1系统需求分析 1](#_Toc21914)

[1.1.1 目标 : 1](#_Toc13096)

[1.1.2 功能需求 : 1](#_Toc23404)

[1.1.3游戏功能 : 1](#_Toc17156)

[1.1.4非功能需求 : 1](#_Toc16772)

[1.1.5可用性要求 : 2](#_Toc25311)

[1.2 系统概要设计 2](#_Toc12848)

[1.2.1 系统功能模块 2](#_Toc19523)

[1.2.2 游戏控制模块 2](#_Toc16229)

[2.4 计分系统模块 2](#_Toc21923)

[1.2.3. 数据结构 3](#_Toc26860)

[二．系统总体设计 4](#_Toc32312)

[2.1 系统功能模块设计 4](#_Toc7500)

[2.1.1 主窗口模块 (Widget) 4](#_Toc32219)

[2.1.2 角色模块 (Player\_Left 和 Player\_Right) 4](#_Toc21180)

[2.1.3 球模块 (Ball) 4](#_Toc8348)

[2.1.4 场景管理模块 (QGraphicsScene) 5](#_Toc222)

[2.1.5 用户界面模块 (UI) 5](#_Toc13352)

[2.1.6 音效模块 (QSoundEffect) 5](#_Toc18704)

[2.2 用例分析与设计 5](#_Toc15669)

[2.2.1 主要用例 5](#_Toc19098)

[2.2.2 用例描述 6](#_Toc7576)

[2.2.3用例设计图示 7](#_Toc18697)

[2.3 系统运行流程 8](#_Toc22441)

[2.3.1启动阶段 8](#_Toc15970)

[2.3.2主菜单阶段 8](#_Toc16925)

[2.3.3 游戏初始化阶段 8](#_Toc38)

[2.3.4游戏进行阶段 8](#_Toc22416)

[2.3.5得分判断阶段 8](#_Toc5431)

[2.3.6游戏暂停阶段 8](#_Toc26430)

[2.3.7游戏继续阶段 8](#_Toc3103)

[2.3.8游戏结束阶段 9](#_Toc22732)

[2.3.9 图例演示 9](#_Toc23621)

[三、系统详细设计 10](#_Toc781)

[3.1 数据结构设计 10](#_Toc29201)

[3.1.1Player（玩家） 10](#_Toc22045)

[3.1.2Ball（球） 10](#_Toc17695)

[3.1.3 Court（场地） 10](#_Toc23385)

[3.1.4 ScoreBoard（计分板） 11](#_Toc4316)

[3.2 类的设计 11](#_Toc25072)

[3.2.1 类层次结构设计 11](#_Toc19275)

[3.2.2 类的描述 12](#_Toc9508)

[3.3核心算法的分析与设计 14](#_Toc4809)

[3.3.1 球的移动算法 15](#_Toc24952)

[3.3.2 碰撞检测算法 15](#_Toc21249)

[3.3.3 得分判断算法 16](#_Toc14423)

[四、系统功能演示与结果分析 16](#_Toc8817)

[4.1系统运行演示 16](#_Toc32256)

[4.2运行结果分析 18](#_Toc9520)

[(1) 功能完整性 18](#_Toc11189)

[(2) 交互性能 18](#_Toc26330)

[(3) 稳定性与可靠性 18](#_Toc16512)

[4.3 系统优化与答辩后改进 18](#_Toc27842)

[五、课程设计总结 18](#_Toc27918)

[概述 : 19](#_Toc2786)

[5.1 设计总体概述 19](#_Toc2926)

[5.2 设计亮点 19](#_Toc6433)

[(1) 界面设计与游戏场景的搭建 19](#_Toc10677)

[(2) 功能实现 19](#_Toc21473)

[(3) 碰撞检测 20](#_Toc26326)

[(4) 游戏难度 20](#_Toc18938)

[(5) 链表的使用 20](#_Toc6964)

[(6) 键盘人物运动的组合 20](#_Toc2954)

[5.3 不足与需完善处 20](#_Toc21165)

[(1) 界面美观度 20](#_Toc10305)

[(2) 游戏内容 20](#_Toc3133)

[(3) 性能优化 20](#_Toc16658)

[5.4 个人收获与总结 21](#_Toc4963)

# 系统需求分析和概要设计

本次课题是基于Qt Creator图形化平台对火柴人打羽毛球这款游戏的复刻，同时搭载了SQLite数据库进行对玩家账户的管理，为玩家提供了双人羽毛球游戏的平台，玩家通过键盘控制火柴人角色进行移动、跳跃和击球，目标是将羽毛球打到对方场地得分。

## 1.1系统需求分析

### 目标 :

(1) 提供一个双人羽毛球比赛的模拟游戏。

(2) 实现用户的基本注册登录操作，同时保存数据。

(3) 实现了难度选择，使得用户有着不同的游戏体验。

(4) 实现火柴人角色的基本运动和动画。

(5) 实现羽毛球的物理运动和碰撞检测。

(6) 提供简单直观的用户界面，包括开始界面、游戏界面和结束界面。

### 1.1.2 功能需求 :

(1) 用户界面：包含注册登录功能，帮助用户管理账号

(2) 开始界面：包含游戏标题、选择按钮（如羽毛球）、说明帮助按钮、帮助按钮和退出按钮。

(3) 游戏界面：选择游戏难度，显示游戏场景、火柴人角色、羽毛球

(4) 结束界面：显示游戏结果（胜利或失败）和继续游戏按钮。

### 1.1.3游戏功能 :

1. 玩家控制：通过键盘控制两个火柴人角色的移动（左右）、跳跃和击球。
2. 羽毛球运动：羽毛球根据物理规则进行运动，包括重力影响、碰撞反弹等。
3. 碰撞检测：检测火柴人角色与羽毛球、羽毛球与地面、羽毛球与网的碰撞。
4. 内置计分系统：根据羽毛球落地位置为双方玩家计分，并更新计分板。
5. 动画效果：实现火柴人角色的运动和击球动画。

### 1.1.4非功能需求 :

1. 游戏响应速度快，人物移动跳跃流畅，玩家控制无明显延迟。
2. 羽毛球运动和碰撞检测需实时进行，确保流畅的游戏体验。

### 1.1.5可用性要求 :

1. 用户界面应简单直观，易于操作。
2. 提供清晰的帮助信息，帮助用户快速上手游戏。
3. 为用户打包.exe文件，不需要配置相应的环境即可使用。

## 1.2 系统概要设计

### 1.2.1 系统功能模块

(1) 开始界面：

显示游戏标题、选择按钮（如羽毛球和排球）、帮助按钮和退出按钮。

通过点击按钮，用户可以进入相应的游戏类型或查看帮助信息。

1. 游戏界面：

显示游戏场景，包括火柴人角色、羽毛球和计分板。

提供玩家通过键盘控制火柴人角色进行移动、跳跃和击球的功能。

1. 结束界面：

显示游戏结果（胜利或失败）。

### 1.2.2 游戏控制模块

1. 玩家控制：

处理玩家输入，通过键盘控制火柴人角色的移动、跳跃和击球。

实现火柴人角色的动画效果，包括跑动和击球动画。

1. 羽毛球控制：

处理羽毛球的物理运动，包括重力、反弹和碰撞检测。

根据玩家的击球动作改变羽毛球的运动轨迹。

1. 碰撞检测模块

角色与羽毛球碰撞：

检测火柴人角色与羽毛球的碰撞，并根据碰撞情况处理击球动作。

羽毛球与地面碰撞：

检测羽毛球与地面的碰撞，判断得分并更新计分板。

羽毛球与球网碰撞：

检测羽毛球与球网的碰撞，并处理羽毛球的反弹。

羽毛球与墙壁碰撞:

对于困难模式中，球的速度较快，于是球与墙壁的碰撞增加了游戏的难度，开创新的玩法。

2.4 计分系统模块

根据羽毛球落地点判断得分，并更新计分板。

实现比赛结束的判定逻辑，当一方得分达到预定值时，游戏结束。

### 1.2.3. 数据结构

(1) 玩家状态

位置（X, Y）：存储火柴人角色的当前位置。

速度（vx, vy）：存储火柴人角色的当前速度。

分数（score）：存储玩家的当前得分。

人物的移动处理 : 将人物动作逐帧进行拆分，用链表类存储人物移动时对应帧的图片，在使用相应的键盘控件时，遍历链表，实现人物的移动

(2) 羽毛球状态

位置（x, y）：存储羽毛球的当前位置。

速度（vx, vy）：存储羽毛球的当前速度。

状态（Ball\_is\_Flying, is\_on\_Left, is\_on\_Right）：存储羽毛球是否在飞行状态、是否在左侧或右侧玩家的控制下。

(3) 界面接口设计

主窗口（Widget）：

负责管理开始界面、游戏界面和结束界面的切换。

游戏视图 (View)

依附于主窗口，实现游戏中不同场景的切换与拆分。

游戏场景（PlayScene）：

负责显示游戏场景和处理游戏逻辑，存储每个场景中相应的元素。

场景元素 (Item)

添加于不同的场景当中，作为场景中的元素，实现游戏的目的。

(4) 游戏控制接口

玩家控制接口：

处理键盘输入，控制火柴人角色的运动。

羽毛球控制接口：

处理羽毛球的运动和碰撞检测。

碰撞检测接口

碰撞检测函数：

判断角色与羽毛球、羽毛球与地面、羽毛球与球网的碰撞情况，并进行相应处理。

计分系统接口

计分函数：

根据羽毛球的落地点更新玩家得分，并判断比赛是否结束。

(5) 关键技术与算法设计

Qt框架

使用Qt框架开发用户界面和游戏逻辑，确保跨平台兼容性。

利用QGraphicsView、QGraphicsScene和QGraphicsItem实现游戏场景和动画效果。

碰撞检测算法

使用简单的边界检测算法判断角色与羽毛球、羽毛球与地面、羽毛球与球网的碰撞。

通过调整羽毛球的速度和方向，实现碰撞后的反弹效果。

动画效果

使用QTimer和QPixmap实现火柴人角色的运动和击球动画。

通过链表，逐帧切换图片，实现流畅的动画效果。

# 二．系统总体设计

本节将对火柴人羽毛球游戏系统的总体设计进行详细描述。系统设计旨在通过合理划分功能模块，提高系统的可维护性、可扩展性和易用性。游戏系统包括多个子模块，每个模块负责特定的功能。以下将详细介绍系统的功能模块设计。

## 2.1 系统功能模块设计

系统功能模块设计包括以下几个主要模块：

### 2.1.1 主窗口模块 (Widget)

主窗口模块负责整体游戏界面的显示和控制。该模块的主要功能包括：

用户的登录注册界面。

初始化游戏界面，包括背景、角色、球网、按钮等元素的加载。

处理用户的输入，如键盘事件和按钮点击事件。

切换不同的游戏场景，如主菜单、游戏场景、结束场景等。

维护游戏的整体逻辑，如开始游戏、暂停游戏、重置游戏等。

### 2.1.2 角色模块 (Player\_Left 和 Player\_Right)

角色模块负责火柴人角色的控制和动画显示。该模块的主要功能包括：

角色的移动控制，包括左右移动和跳跃。

角色的动画处理，不同动作的切换（如行走、跳跃、击球等）。

角色的碰撞检测，与球和场地边界的交互处理。

### 2.1.3 球模块 (Ball)

球模块负责游戏中羽毛球的行为控制和物理效果。该模块的主要功能包括：

初始化球的位置和状态。

控制球的移动和速度，包括重力和反弹效果。

实现球与墙壁、球网和角色的碰撞检测和反弹逻辑。

记录和更新球的位置，处理发球逻辑。

### 2.1.4 场景管理模块 (QGraphicsScene)

场景管理模块负责不同游戏场景的管理和切换。该模块的主要功能包括：

初始化不同的场景（主菜单、游戏场景、结束场景等）。

添加和管理场景中的图形元素，如背景、角色、球等。

处理场景之间的切换逻辑，根据游戏进程显示相应的场景。

### 2.1.5 用户界面模块 (UI)

用户界面模块负责显示游戏的用户界面元素和信息。该模块的主要功能包括：

显示登录，注册等按钮。

显示控制按钮，如开始按钮、继续按钮、退出按钮等。

提供游戏帮助和菜单选项的访问入口。

### 2.1.6 音效模块 (QSoundEffect)

音效模块负责管理和播放游戏中的音效。该模块的主要功能包括：

加载和播放背景音乐和游戏音效。

在特定事件触发时播放相应的音效。

通过以上模块的协同工作，系统实现了一个完整的火柴人羽毛球游戏。每个模块各司其职，保证了系统的清晰结构和高效运行。

## 2.2 用例分析与设计

### 2.2.1 主要用例

(1) 参与者：

玩家1（左侧玩家）

玩家2（右侧玩家）

(2) 用例：

登录注册: 玩家在启动游戏前进行登录注册。

启动游戏：玩家启动游戏应用程序，进入主菜单。

选择难度：玩家选择游戏的难度级别（简单或困难）。

开始游戏：玩家从主菜单进入游戏界面，开始比赛。

移动角色：玩家通过键盘控制角色的移动和跳跃。

发球：玩家发球开始游戏。

击球：玩家通过击球与对手对抗。

得分：系统根据球的落点判断得分，并更新计分板。

暂停游戏：玩家暂停游戏，系统显示暂停菜单。

继续游戏：玩家继续游戏，系统恢复游戏状态。

结束游戏：玩家结束游戏，系统显示比赛结果。

### 2.2.2 用例描述

每个用例可以通过用例描述来详细说明其执行过程、参与者、前置条件和后置条件等信息。以下是部分用例的描述：

1. 用例：启动游戏

参与者：玩家1，玩家2

描述：玩家启动游戏应用程序，进入主菜单。

前置条件：玩家进行了登录注册操作。

后置条件：系统显示主菜单，等待玩家选择操作。

1. 用例：选择难度

参与者：玩家1，玩家2

描述：玩家选择游戏的难度级别（简单或困难）。

前置条件：系统显示主菜单

后置条件：系统设置游戏难度级别，准备开始游戏。

1. 用例：开始游戏

参与者：玩家1，玩家2

描述：玩家从主菜单进入游戏界面，开始比赛。

前置条件：玩家已选择难度级别

后置条件：系统显示游戏界面，初始化比赛状态。

1. 用例：移动角色

参与者：玩家1，玩家2

描述：玩家通过键盘控制角色的移动和跳跃。

前置条件：游戏正在进行

后置条件：角色根据玩家输入移动或跳跃，更新角色位置。

1. 用例：发球

参与者：玩家1，玩家2

描述：玩家发球开始游戏。

前置条件：游戏初始化状态或得分后重置状态

后置条件：球开始运动，游戏进入对抗阶段。

1. 用例：击球

参与者：玩家1，玩家2

描述：玩家通过击球与对手对抗。

前置条件：球在场地内飞行

后置条件：球根据击球方向和力量改变运动轨迹。

1. 用例：得分

参与者：系统

描述：系统根据球的落点判断得分，并更新计分板。

前置条件：球落地

后置条件：更新得分，重置球的位置和状态

### 2.2.3用例设计图示



## 2.3 系统运行流程

### 2.3.1启动阶段

玩家启动游戏应用程序，系统显示主菜单。

玩家可以选择“开始游戏”或“退出”。

### 2.3.2主菜单阶段

玩家在主菜单中选择游戏难度（简单或困难）。

玩家点击“开始游戏”按钮进入游戏界面。

### 2.3.3 游戏初始化阶段

系统根据选择的难度级别初始化游戏参数。

系统显示游戏界面，玩家1和玩家2的角色和球初始化到起始位置。

### 2.3.4游戏进行阶段

玩家通过键盘控制角色移动和跳跃。

玩家通过按键发球，球开始运动。

玩家通过击球与对手对抗，球在场地内飞行。

系统根据球的运动和碰撞检测更新球的位置和状态。

### 2.3.5得分判断阶段

系统检测球是否落地或出界。

如果球落地，系统根据球的落点判断得分。

系统更新得分板，重置球的位置和状态，准备下一回合。

### 2.3.6游戏暂停阶段

玩家可以按键暂停游戏，系统显示暂停菜单。

玩家可以选择继续游戏或退出游戏。

### 2.3.7游戏继续阶段

玩家选择继续游戏，系统恢复游戏状态，继续比赛。

### 2.3.8游戏结束阶段

当一方玩家得分达到设定的目标分数时，系统结束游戏。

系统显示比赛结果，玩家可以选择重新开始或退出游戏。

### 2.3.9 图例演示



# 三、系统详细设计

系统详细设计阶段是在系统总体设计的基础上，对系统各个模块进行详细的设计，包括数据结构设计、类的设计和核心算法的设计。本章节将详细描述介绍火柴人羽毛球游戏系统的数据结构、类的层次结构和核心算法，确保系统的各个模块能够按照预期功能正常运行。

## 3.1 数据结构设计

系统的数据结构设计涉及游戏中的主要对象及其属性和行为。在火柴人羽毛球游戏中，主要的数据结构包括玩家、球、场地、得分等。以下是主要的数据结构设计：

### 3.1.1Player（玩家）

属性：

位置（Position）：表示玩家在游戏界面中的位置（x, y）。

速度（Speed）：表示玩家的移动速度。

分数（Score）：表示玩家的当前得分。

状态（State）：表示玩家的当前状态，如移动、跳跃、击球等。

行为：

移动（Move）：根据键盘输入控制玩家的移动。

跳跃（Jump）：根据键盘输入控制玩家的跳跃。

击球（Hit）：根据键盘输入控制玩家的击球动作。

### 3.1.2Ball（球）

属性：

位置（Position）：表示球在游戏界面中的位置（x, y）。

速度（Velocity）：表示球的当前速度（vx, vy）。

状态（State）：表示球的当前状态，如飞行中、跟随玩家等。

重力加速度（Gravity）：模拟球的下落速度。

行为：

移动（Move）：更新球的位置和速度，实现球的飞行和碰撞检测。

重置位置（ResetPosition）：根据得分情况重置球的位置。

### 3.1.3 Court（场地）

属性：

大小（Size）：表示场地的大小。

网（Net）：表示场地中间的球网位置和大小。

行为：

碰撞检测（CollisionDetection）：检测球与场地边界和球网的碰撞。

### 3.1.4 ScoreBoard（计分板）

属性：

左侧玩家分数（LeftScore）：表示左侧玩家的得分。

右侧玩家分数（RightScore）：表示右侧玩家的得分。

行为：

更新分数（UpdateScore）：更新计分板上的分数显示。

## 3.2 类的设计

### 3.2.1 类层次结构设计

火柴人羽毛球游戏系统主要包括以下几个类：Widget、Player、Ball、Menu、Lnlog、

Regis、Instruction、Mynet、Player\_Right、Player\_Left。类层次结构如下图所示：

Widget ----- Regis Lnlog

├ Player Player\_Right Player\_Left

├ Ball

├ Menu

├ Instruction

├ Mynet



### 3.2.2 类的描述

(1) Widget 类

描述：游戏主窗口类，负责游戏的初始化和控制游戏的主要逻辑。

属性：

GameView：游戏视图。

PlayScene：游戏场景。

Player\_Left：左侧玩家对象。

Player\_Right：右侧玩家对象。

Ball：球对象。

ScoreBoard：计分板对象。

Difficulty：游戏难度。

方法：

keyPressEvent：处理键盘按键事件。

keyReleaseEvent：处理键盘释放事件。

Player\_Left\_Move：控制左侧玩家的移动。

Player\_Right\_Move：控制右侧玩家的移动。

resetPlayerPositions：重置玩家位置。

updateScoreBoard：更新计分板。

checkBallPosition：检查球的位置。

checkGameOver：检查游戏是否结束。

1. Player 类

描述：表示游戏中的玩家。

属性：

Position：玩家位置。

Speed：玩家移动速度。

Score：玩家得分。

State：玩家状态。

方法：

Move：控制玩家移动。

Jump：控制玩家跳跃。

Hit：控制玩家击球。

1. Ball 类

描述：表示游戏中的球。

属性：

Position：球的位置。

Velocity：球的速度。

State：球的状态。

Gravity：重力加速度。

方法：

Move：更新球的位置。

resetBallPosition：重置球的位置。

followPlayer：球跟随玩家。

getCollisionRect：获取球的碰撞矩形。

getBallPosition：获取球的位置。

1. Menu 类

描述：表示游戏的菜单界面。

属性：

StartButton：开始按钮。

SettingsButton：设置按钮。

ExitButton：退出按钮。

方法：

showMenu：显示菜单界面。

handleButtonClick：处理按钮点击事件。

1. Lnlog 类

描述：表示登录界面。

属性：

UsernameField：用户名输入框。

PasswordField：密码输入框。

LoginButton：登录按钮。

方法：

showLogin：显示登录界面。

handleLogin：处理登录事件。

1. Regis 类

描述：表示注册界面。

属性：

UsernameField：用户名输入框。

PasswordField：密码输入框。

RegisterButton：注册按钮。

方法：

showRegister：显示注册界面。

handleRegister：处理注册事件。

1. Instruction 类

描述：表示游戏说明界面。

属性：

InstructionText：说明文字。

BackButton：返回按钮。

方法：

showInstructions：显示游戏说明。

handleBack：处理返回事件。

1. Mynet 类

描述：表示球网对象。

属性：

Position：球网位置。

Size：球网大小。

方法：

getCollisionRect：获取球网的碰撞矩形。

1. Player\_Right 类

描述：表示右侧玩家。

继承：继承自 Player 类。

属性：

SpecificAttributes：右侧玩家的特定属性。

方法：

SpecificMethods：右侧玩家的特定方法。

1. Player\_Left 类

描述：表示左侧玩家。

继承：继承自 Player 类。

属性：

SpecificAttributes：左侧玩家的特定属性。

方法：

SpecificMethods：左侧玩家的特定方法。

## 3.3核心算法的分析与设计

本章节将详细描述火柴人羽毛球游戏系统中的核心算法，包括球的移动算法、碰撞检测算法和得分判断算法。

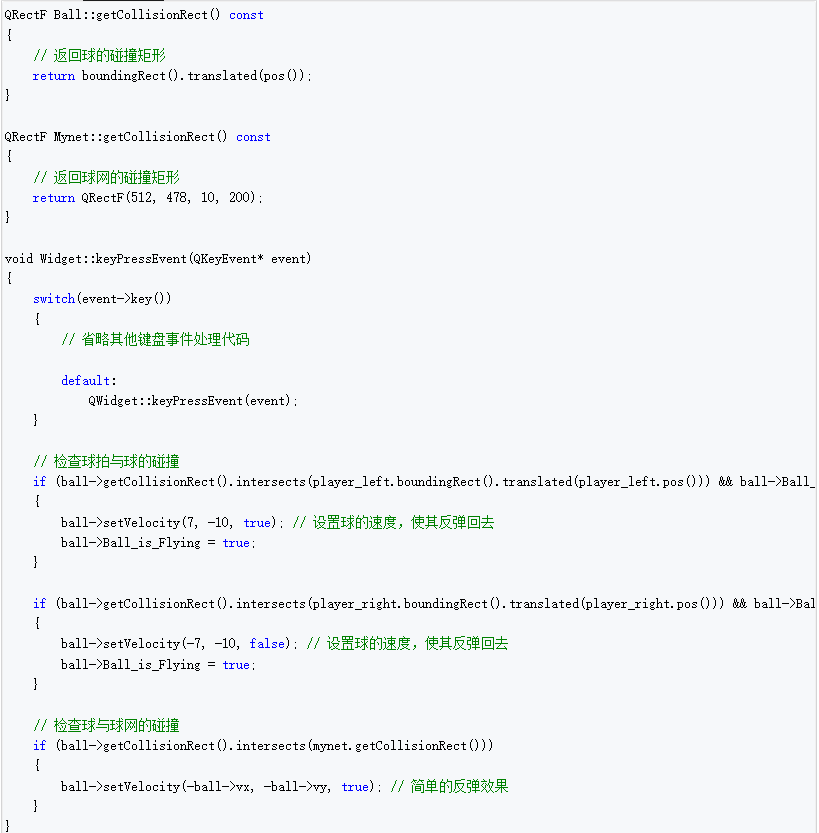
### 3.3.1 球的移动算法

球的移动算法负责更新球的位置和速度，模拟球在场地中的运动轨迹。算法考虑了球的初速度、重力加速度和碰撞反弹效果



### 3.3.2 碰撞检测算法

碰撞检测算法负责检测球与球网、场地边界和玩家之间的碰撞，并根据碰撞情况更新球的速度和方向。



### 3.3.3 得分判断算法

得分判断算法负责检测球是否落地，并更新得分情况。根据得分情况，重置球的位置并检查游戏是否结束。

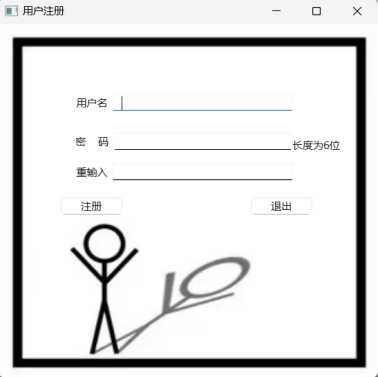
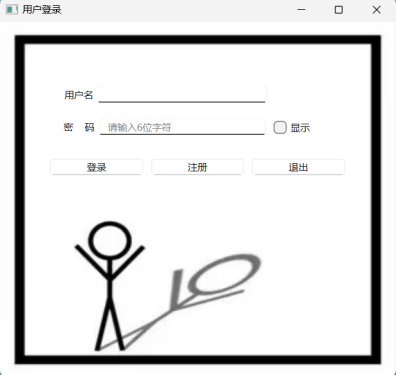
### 

# 四、系统功能演示与结果分析

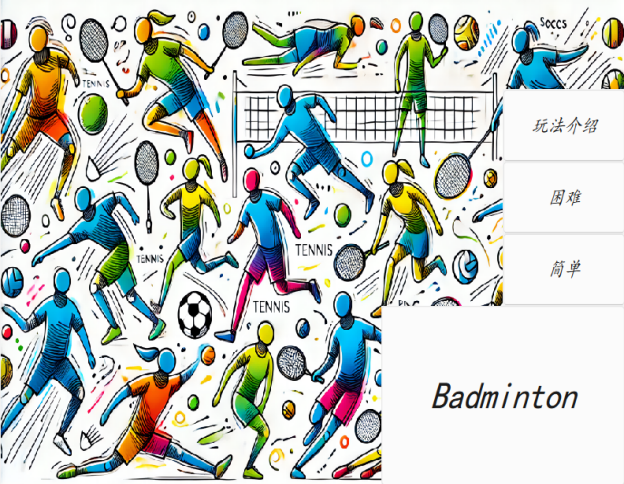
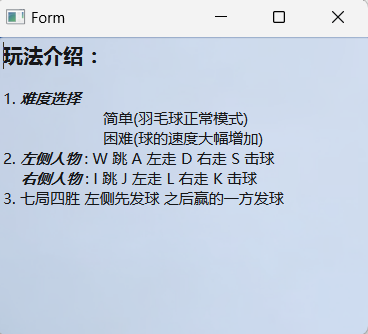
系统功能演示与结果分析阶段旨在展示火柴人羽毛球游戏系统的实际运行效果，分析其运行结果，以评估系统的功能完整性和性能表现。本章节将通过具体的系统运行演示和结果分析，验证系统设计和实现的有效性。

## 4.1系统运行演示

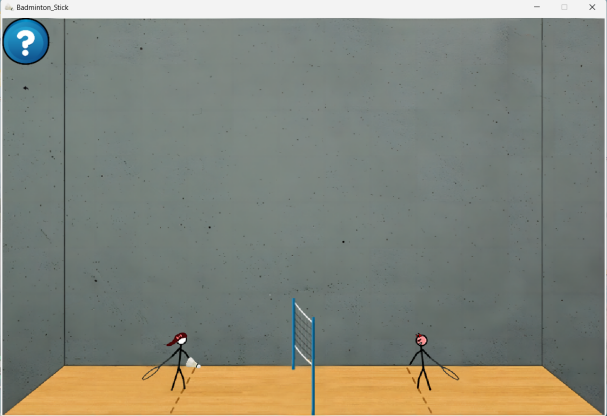
1. 注册界面 (2) 登录界面

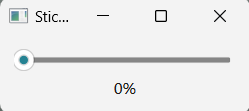
(3) 功能选择界面 包含 玩法介绍 难度选择 选择游戏 (4) 玩法介绍

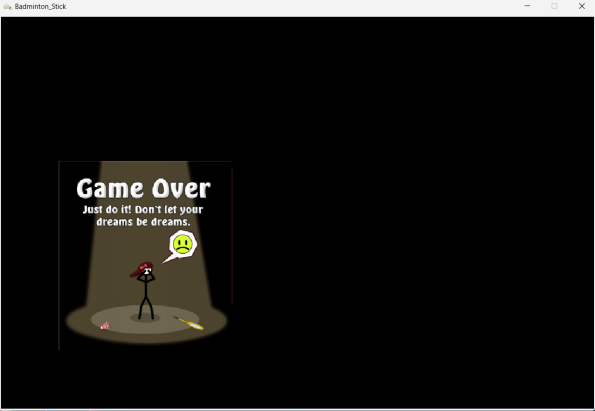
(5) 开始游戏界面 (6) 游戏界面



(7) 音量控制界面



1. 胜利与失败界面



## 4.2运行结果分析

通过对系统的运行演示，可以进行以下结果分析：

1. 功能完整性

系统的主要功能模块，包括登录与注册、主菜单、游戏界面、游戏结束等，能够正常运行。

各功能模块之间的切换流畅，用户体验良好。

1. 交互性能

用户通过键盘控制玩家的移动和操作，响应速度快，无明显延迟。

碰撞检测和球的反弹效果真实，符合预期。

1. 稳定性与可靠性

系统运行过程中，无明显的崩溃或错误现象。

各个模块的功能实现稳定可靠。

## 4.3 系统优化与答辩后改进

(1) 为注册界面提供了注册提示，在游戏功能选择界面增添了游戏玩法说明控件，使得用户玩家体验更好，能够迅速掌握该款游戏。

(2) 将原先的发球方式 : 单数左侧发球，双数右侧发球改为开局默认左侧发球，后续赢方继续发球，使得该款游戏更贴近于生活。

(3) 为功能选择界面提供了游戏玩法的创新，包含游戏的简单与困难模式，使游戏形式更加新颖。

# 五、课程设计总结

第五章节将对火柴人羽毛球游戏系统的课程设计进行总结，概述设计的总体情况，分析设计中的亮点以及存在的不足之处，并提出需要完善的方面。

## 概述 :

本课程设计旨在通过开发一个简单但具有趣味性的火柴人羽毛球游戏系统，提升学生对软件开发全过程的理解和实践能力。设计过程涵盖了需求分析、系统设计、代码实现、测试与验证等多个环节，通过实际操作，培养了学生的编程能力和解决问题的能力。

## 5.1 设计总体概述

火柴人羽毛球游戏系统的设计总体概述如下：

项目背景

该项目旨在为学生提供一个实践平台，通过开发一个简易的游戏系统，综合运用所学的编程知识和开发技能。

系统功能

系统主要包括登录与注册、主菜单、游戏界面、游戏结束等功能模块。

用户可以选择不同的游戏难度，通过键盘控制火柴人进行羽毛球对战，系统实时更新比分并判断胜负。

技术实现

使用C++语言和Qt框架进行开发，Qt提供了丰富的界面和事件处理功能。

通过面向对象的设计方法，合理划分系统模块，保证代码的结构清晰和可维护性。

## 5.2 设计亮点

本次课程设计的亮点包括：

1. 界面设计与游戏场景的搭建

界面简洁美观，用户操作方便。通过Qt框架提供的图形界面功能，实现了友好的用户交互界面。通过元素 -> 场景 -> 视图 -> 窗口 这一流程，完整的搭建了我们的游戏窗口。

1. 功能实现

系统功能完备，包括用户登录注册、游戏控制、实时计分、胜负判断等。

通过键盘事件处理，实现了玩家对火柴人的灵活控制，增强了游戏的趣味性。

1. 碰撞检测

实现了球与玩家、球网、地面和墙壁的碰撞检测，使游戏逻辑更加合理和真实。

通过模拟重力效果，使球的运动更加贴近现实。

1. 游戏难度

提供了不同的游戏难度选择，满足不同用户的需求。

根据难度调整球的速度，增加了游戏的挑战性。

1. 链表的使用

利用Qt中的List实现小量数据的遍历与存储，对于人物移动的帧数播放这样多次少量的操作，List的实现极大的减少了内存的损耗，同时增加了人物的流畅性。

1. 键盘人物运动的组合

利用容器原件，将键盘反馈的数据，同时存储在容器当中，实现了跳与击球与移动的同时进行，使得人物更加灵活，增加了游戏的流畅性与可玩性。

## 5.3 不足与需完善处

1. 界面美观度

当前界面设计相对简单，可以进一步优化界面的美观度和细节处理，增加更多动画效果和音效。

1. 游戏内容

游戏内容较为单一，可以增加更多的游戏模式和玩法，如单人模式、多人联机模式等。

增加更多的玩家和球的皮肤选择，提升用户的个性化体验。

1. 性能优化

进一步优化核心算法，提高系统的运行效率，减少资源消耗。

增强系统的稳定性，避免在复杂操作下出现崩溃或卡顿现象。

## 5.4 个人收获与总结

本次面向对象课程设计，我认为不仅仅是单独对我代码能力与面向对象能力的考验，本次课程设计从多个方面提升了我自己。

通过课程设计，额外学习了Qt Create，SQL的语法语句，数据库的使用，git在周期代码管理的应用，甚至于图像音频的处理，我认为这些都是本次课程设计给我带来的额外的巨大的收获。

将整个大的游戏分拆成一个个小小的类，一个个场景，一个个元素，在对这些类与其他内容的包装，封装成一个完整的游戏。

面对困难与问题时，我会不断尝试新的方法，面对新的程序平台，我认识到了指导文档的重要性，对一些自己不知道的头文件与方法，帮助文档极大的帮助了我完成本次课设。

这些思想方法与技能，我相信在今后的学习生活中会给我带来巨大的帮助。