南开大学 计算机大类

姓名：侯佳辛

学号：2413724

班级：3205

2025年5月13日

**高级语言程序设计2—2**

**实验报告**

目录

[高级语言程序设计大作业实验报告 1](#_Toc21286)

[一. 作业题目 1](#_Toc29969)

[二. 开发软件 1](#_Toc18364)

[三. 课题要求 1](#_Toc742)

[四. 主要流程 1](#_Toc25240)

[1． 整体流程 2](#_Toc20452)

[2． 算法或公式 2](#_Toc25354)

[五. 测试与优化 6](#_Toc28265)

[六. 实验总结 6](#_Toc23116)

**高级语言程序设计大作业实验报告**

1. **作业题目**

基于飞机大战的麦克莫顿（第五人格游戏角色）捕获爆米花游戏

1. **开发软件**

Visual Studio 2017

QT5.15.2

1. **课题要求**

1. 功能要求

- 实现玩家麦麦操控与黑球发射功能

- 生成敌方爆米花并具备移动能力

- 完成黑球与爆米花的碰撞检测，处理碰撞后的逻辑

2. 技术要求

- 基于Qt框架开发，使用C++语言

- 采用模块化设计，合理划分游戏对象类

- 利用定时器实现动画效果

1. **主要流程**
   1. **整体流程**

实现思路：

1. 项目初始化与环境搭建

- 创建Qt项目：使用Qt Creator新建一个基于QWidget的Qt Widgets Application项目，配置好编译器与构建环境。

- 资源准备：收集或设计游戏所需图片资源，将资源文件添加到Qt项目的资源文件（ .qrc ）中。

2. 加载游戏资源（图片、音效等）。

3. 创建并显示Mainscene主窗口。

4. 进入事件循环，等待用户交互（如鼠标点击、定时器触发）。

程序启动

├── 初始化Qt应用

├── 创建Mainscene窗口

│ ├── 显示"开始游戏"按钮

│ └── 等待用户点击

│

└── 用户点击按钮

├── 隐藏按钮

├── 初始化游戏参数

├── 启动定时器（游戏循环）

│ ├── 生成爆米花

│ ├── 更新所有元素位置

│ ├── 重绘界面

│ ├── 检测碰撞

│ └── 重复执行（每8ms一次）

│

└── 处理用户交互

├── 鼠标移动控制麦麦

└── 自动发射红球

* 1. 算法或公式

**地图滚动算法**

双背景循环滚动：

1. 两张背景图片垂直排列，一张在屏幕内（map2），一张在屏幕上方（map1）。 2. 两张图片同时向下滚动（posy += scroll\_speed）。

3. 当第一张图片完全移出屏幕（posy >= 0），将其重置到屏幕上方（posy = -game\_height），实现无缝循环。

void map::**mapPosition**()

{

//对第一张背景处理

map1\_posy += scroll\_speed;

if(map1\_posy >=0)

{

map1\_posy = -game\_height;

}

//处理第二张背景

map2\_posy += scroll\_speed;

if(map2\_posy >= game\_height)

{

map2\_posy = 0;

}

}

**玩家控制**：通过监听鼠标事件，实现玩家的左右移动与发射黑球功能。

void Mainscene::***mouseMoveEvent***(QMouseEvent \* event)

{

int x= event->x()-m\_maimai.m\_rect.width()\*0.5;

int y= event-// map::mapPosition()

m\_maimai.m\_rect.height()\*0.5;

//防止飞机出边界，设置边界检测

if(x<=0)

{

x=0;

}

if(x>=game\_width-m\_maimai.m\_rect.width())

{

x=game\_width-m\_maimai.m\_rect.width();

}

if(y<=0)

{

y=0;

}

if(y>=game\_height-m\_maimai.m\_rect.height())

{

y=game\_height-m\_maimai.m\_rect.height();

}

m\_maimai.position(x,y);

}

**爆炸检测**

**矩形碰撞检测（AABB 算法）： ◦ 检测两个矩形是否重叠。 ◦**

**数学条件：**

**rect1.right > rect2.left &&**

**rect1.left < rect2.right &&**

**rect1.bottom > rect2.top &&**

**rect1.top < rect2.bottom ◦ Qt 中使用QRect::intersects()直接实现。**

//红球爆米花相碰

if(p\_popcorns[i].p\_rect.intersects(m\_maimai.m\_balls[j].b\_rect))

{

p\_popcorns[i].p\_free=true;

m\_maimai.m\_balls[j].b\_free=true;

//进行爆炸

for (int k=0;k<bombnum;k++)

{

if(bombs[k].b\_free)

{

bombs[k].b\_free=false;

//更新爆炸坐标

bombs[k].b\_x=p\_popcorns[i].p\_x;

bombs[k].b\_y=p\_popcorns[i].p\_y;

break;

}

}

}

**. 随机数生成**

// Mainscene::popcornappear()

p\_popcorns[i].p\_x = rand() % (game\_width - p\_popcorns[i].p\_rect.width());

均匀分布随机数：公式：min + rand() % (max - min + 1) ◦ 例如：生成 0 到game\_width-物体宽度之间的随机数。

**五．测试与优化**

- 功能测试：检查游戏各功能是否正常运行，包括麦麦移动、黑球发射、碰撞检测等。

- 性能优化：优化碰撞检测算法，

**六．实验总结**

本次Qt项目通过结合Qt的图形界面框架与事件处理机制，实现了一个完整的2D射击游戏。开发过程中深入学习了Q碰撞检测算法及游戏逻辑设计，提升了C++编程与界面开发能力。未来可进一步扩展游戏功能，如增加关卡系统、多人对战模式等，增强游戏趣味性与可玩性。

}