高级语言程序设计实验报告

南开大学 计算机类

颜铭

2013365

2021年5月8日

目录：

高级程序大作业实验报告………………………………1

1. 作业简介……………………………………………………1
2. 立项过程……………………………………………………1
3. 开发流程……………………………………………………2
4. 整体概要
5. 算法/核心代码：

**关键算法**：

游戏基础功能：简单物理模型算法，随机更新地图算法，游戏功能交互算法；

游戏模式算法：spfa算法，二阶动态规划算法，滑动窗口算法

1. 开发问题综述
2. 用户测试体验效果
3. 开发感悟与收获………………………………………………8
4. 程序终端运行…………………………………………………9

高级语言程序设计大作业实验报告

1. 作业简介：

游戏名称：runlegend;

一款基于qt平台开发的平面传奇类跑酷小游戏，通过槽函数关联实现主界面与游戏主体窗口的交互，在游戏时计时函数信号触发物体本身参数函数，主体窗口重绘与键盘事件达到使物体运动的目的，基本游戏模式为玩家通过操控角色躲避障碍物和寻宝加分。游戏结束时弹出画面交互玩家可以选择回到首页或者重新开始，整体互动性较强。

1. 立项过程：
2. 受qt项目特色功能槽函数和计时器线程启发，初步定向为一个灵活且开放的动态游戏。
3. 个人此前对传奇冒险画风的游戏有过涉猎，选择该方向方便在设计过程中融入游戏元素。
4. 为锻炼代码水平，应该使得编写过程中对象明确逻辑清晰可执行高，由于跑酷游戏基础模式只需有角色，道具，背景等要素。采用随机刷新地图的方式也可以增加代码利用效率。
5. 基于以上综合考虑，确定项目为开发一款传奇类跑酷小游戏。
6. 开发流程：
7. 整体概要：

主界面配置有四个按钮，分别是最高纪录、操作、玩法、与开始键。因而游戏设有实时计分功能与保存当前最高分功能。

操作按键简明扼要介绍了游戏所有关键键盘操作与鼠标操作，玩法特别说明了游戏背景使得道具设置更加合理。开始按键通过槽函数触发计时器关联游戏主界面；

游戏界面的障碍物主要是随机刷新的三种墙体，羽箭及陨石。

其中陨石实现逆时针自转外还能自动伸缩，增大游戏难度。

碰到障碍物会扣分、减速和减血。

人物可以自由上下左右移动，往上移动时可以连续点击w键实现二段跳，同时人物能在特定范围瞬移，以此跨越不可直接躲避的障碍物。

游戏的道具有：

血瓶：可以提供回血；

神杖：可以让界面的所有障碍物短暂消失；

圣者遗迹：继承神杖功能外，对玩家实行奖励，玩家在特定运动范围内拾取能量，获得最大能量触发额外奖励。算法为滑动窗口算法。

哥布林怪盗：游戏npc角色，移速快频率高，通过动态规划算法实现类似田忌赛马的最佳博弈策略，会使玩家丧失大量分数；

天书：通过spfa算法计算树形结构上强连通子图失效的单位时间，实现奖励和惩罚相结合的功能。较好地将游戏背景神秘元素以及编程数据的抽象化结合在一起。

背景图由五张图片长周期循环组成，地面也会随着前进不断绘制。

1. 算法/核心代码：

·实现简单物理跳跃模型：

在人物类里设置加速度和初速度等参数并单独创建计时器进行累加运算实现速度变化，再通过布尔变量关联键盘按下事件反馈到坐标累加变化。

if(up){

if(this->y==610){

jumptimer.start();

jumponce=false;

}

if(this->y<610&&twicerdy&&jumponce){

jumptimer.start();

fallspeed=0;

twicerdy=false;

}

if(jumptimer.isActive()){

this->y-=jumpspeed\*jumptimer.remainingTime()/400;

}

}

·实现陨石障碍物的自转滚动伸缩物理模型：

在圆形类中设置半径，起始角，角频率，广角标等参数，在构造函数中根据图片宽高设定常数，同时限制角度变化在360度内关联倾斜与渐变函数。

circle::**circle**(int x,int y,int width,int height)

:barriers(x,y,width,height){

img.load(circle1);

img.scaled(width,width);

this->r=width/2;

this->collisionqwidth=width\*0.7;

this->collisionangle=width\*0.15;

}

QPixmap circle::**getimg**(){

QTransform m;

degree+=1;

if(degree>=360)

degree=0;

m.rotate(degree);

return this->img.transformed(m,Qt::FastTransformation);

}

·游戏功能交互算法：

矩形碰撞检测，子窗口信号槽传参父窗口，全局布尔变量参数开关；

·随机更新地图算法：

在构造赋值时按要求随机生成物体x、y坐标，if语句特判生成条件。通过随计时器触发超时信号递增的difficult难度参数打乱物体生成分布。

·圣者遗迹模式下，玩家只能按规定的最小和最大步数行走。通常情况下要解的最优解可以穷举所有路径，但是n方级复杂度并不理想。

只要注意到每次行进范围一定，就可以转化为一个子循环访问区间极值的问题。最小值从对头出列最大值从队尾入列并预处理最值的所有插头，不断更新区间状态和维护单调队列。最后再遍历择优即可。

int bequest::**quefind**(int x){

while (que[head] + right< x)

head++;

return que[head];

}

int bequest::**prize**(){

int ans=-5;

memset(*que*,0,sizeof(que));

memset(*f*, 128, sizeof(f));

f[0]=0;

for (int i = left; i <= n; i++) {

while( f[i-left] >= f[que[tail]] && tail >= head)

tail--;

que[++tail] = i-left;

int from = quefind(i);

f[i] = f[from] + eng[i];

if (i + right> n){

ans=std::max(ans,f[i]);

}

}

return ans;

}

哥布林怪盗角色，通过动态规划可以为电脑选择最佳博弈策略；

由于数据不确定性，优选考虑贪心思想。为使电脑掌握主动权，电脑出点顺序将是离散的不是最大就是最小的即时策略，那么就不用关心我们人物的出点方法。假设人物按强到弱出点，就可以用二维数组f[i][j]表示电脑出j较大点数和i-j较小点数与玩家i个点数匹配后能获得最大收益，用g[i][j]储存电脑方出点所得相应收益。

易得状态转移方程：

f[i][j] = max(f[i - 1][j] + g[num - (i - j) + 1][i], f[i - 1][j - 1] + g[j][i]);

bool **cmp**(int a, int b){

return a>b;

}

int goblin::**steal**(){

std::sort( role, role + num,cmp);

std::sort(burger ,burger + num,cmp);

memset(*g*, 0, sizeof(g));

for (int i = 1; i <=num; i++) {

for (int j = 1; j <= num; j++) {

if (role[i] >burger[j])

g[i][j] = -100;

if (role[i] < burger[j])

g[i][j] = +150;

}

}

memset(*f*, -0x7f, sizeof(f));

f[0][0] = 0;

for (int i = 1; i <= num; i++) {

f[i][0] = f[i - 1][0] + g[num - i + 1][i];

f[i][i] = f[i - 1][i - 1] + g[i][i];

for (int j = 0; j < i; j++)

f[i][j] = std::max(f[i - 1][j] + g[num - (i - j) + 1][i], f[i - 1][j - 1] + g[j][i]);

}

int ans=f[num][0];

for (int i = 2; i <= num; i++) {

ans = std::max(ans, f[num][i]);

}

return ans;

}

·天书模式中计算了这样一个故事背景问题：

总计n只猴子，第一只猴子尾巴挂在树上，剩下的猴子左右手或拉着其他猴子或放空。给定m个单位时间，记录0到m-1时刻，每个单位时间给出任意一只猴子的放手信息，计算每只猴子掉落时间，由于错综复杂的拉手结构很有可能到了m时刻都没掉下来，输出“-1”；

分析：由于无向子图强连通，入手点因是逆序连边遍历也就是特判边层关系；本题固定由第一只猴子出发，可以转化为单源最短路径问题，模板spfa算法，把放手时间当做边权，不放手边权赋值为大数，若m时还没有变化输出“-1”，就可以解决问题；

void magicalbook::**spfa**(){

std::queue<int> q;

q.push(1);

while (!q.empty()) {

int u = q.front(); q.pop();

for (int i = a[u]; i; i = net[i]) {

int v = to[i], wd = std::min(ans[u], w[i]);

if (wd > ans[v]) {

ans[v] = wd;

q.push(v);

}

}

}

}

void magicalbook::**monktime**(){

ans[1]=m;

for (int i = 0; i < m; i++) {

int x=rand()%n+1;

int y=rand()%2+1;

monout[0][i]=x;

monout[1][i]=y;

if (y == 1)

l[0][x] = std::min(i, l[0][x]);

if (y == 2)

r[0][x] = std::min(i, r[0][x]);

}

int t = 0;

for (int i = 1; i <= n; i++) {

int x = l[1][i], y = r[1][i];

if (x != -1) {

net[++t] = a[i], to[t] = x, w[t] = l[0][i], a[i] = t;

net[++t] = a[x], to[t] = i, w[t] = l[0][i], a[x] = t;

}

if (y != -1) {

net[++t] = a[i], to[t] = y, w[t] = r[0][i], a[i] = t;

net[++t] = a[y], to[t] = i, w[t] = r[0][i], a[y] = t;

}

}

spfa();

以上就是项目核心算法。

1. 开发问题综述：

首先，由于官方的qt creator6.0版本还在市场投放环节，再开发过程中移除了原来版本的一些动态库，比如multimedia多媒体库，这就导致无法加入音频。通过重新下载目前最成熟的5.14版本得到解决；

在实现程序基本功能后发现无法运行，在网上查阅资料后得知是系统环境变量配置缺少qmake的dll脚本导致无法编译，重新加入mingqw\_64跨平台资源后再关闭影子构建实现运行。

然后在运行调试过程中遇到了奇怪的程序异常崩溃问题，检查了空指针未初始化引用和数组越界等没有出现问题。一开始怀疑是图片像素在程序窗口外加载引发线程负载。但是了解qt准静态图片保护原理后排除。在F5调试下，发现是线程处理时由于冲突导致循环语句执行时没有及时清空数据导致访问操作空指针。通过break强制截断加强程序逻辑问题得到解决。

最后值得一提，在父窗口运算传参到子类等待反馈，执行临时对话容易引发资源死锁问题，因注意动态对象数据初始化不要超线程。

1. 用户测试体验：

操作较为灵活，游戏有新意，分数奖励与惩罚机制较合理，且代码逻辑清晰方便今后开发过程继续载入新内容。

1. 开发感悟收获：

C++语言跟接近底层，在现今互联网工业界较少在应用层运用。在使用qt进行C++开发的过程中接触到了动态静态编译，项目头文件宏定义集成，工作线程与代码匹配等底层问题，不仅能拓宽知识面巩固课堂指针，类，继承与派生，虚函数等知识点，还能不断锻炼程序调试问题，突破传统c++作业一百行代码的局限，在大量代码查阅中能提升程序逻辑理解能力。此外，把典型的c++作业题的控制台程序加入到游戏元素中也起到了发散思维的作用。总体而言，独立开发游戏是一个有趣享受的过程。

1. 程序终端运行：

版本后记：

1.0基本场景和程序基本逻辑构建，实现人物躲避和障碍物扣血，存档与加分功能一同实现；

2.0 添加道具神杖、血瓶；

3.0修复界面按钮显示Bug；

4.0 修复道具数组越界问题导致程序异常结束问题；

5.0 添加新道具圣者遗迹，哥布林怪盗与天书；

6.0 完善天书功能，实现恶魔降临的运动和落地爆炸特效；

7.0 补充添加背景图片，实现滚动播放切换场景体现时间变化；

8.0 添加音频特效

9.0 完善详细玩法介绍操作介绍等策划，重置存档；

10.0 debug 和 release 版本；

**经多次调试，程序debug版本基本不会出现异常退出问题；**

**Release版本偶尔会卡顿；**

视频网址： <https://www.bilibili.com/video/BV1Xh411m7en>

项目网址：https://gitee.com/yan-guo-ming/running-legend