目 录

第一章 概述 1

1.1 研究意义 1

1.2 开发平台介绍 1

第二章 系统整体设计 2

2.1 系统介绍 2

2.2 系统所具有的功能 2

2.3 系统设计 3

2.3.1 系统功能结构 3

2.3.2 程序结构设计 3

2.4 出错处理设计 4

2.5 系统菜单结构设计 4

2.6 游戏关口选择的设计 4

第三章 系统详细设计 6

3.1 游戏界面的实现 6

3.1.1 游戏菜单实现 6

3.1.2 游戏关口选择实现 7

3.1.3 游戏声效的实现 8

3.2 游戏的核心算法设计与实现 8

3.2.1 地图文件读取模块 8

3.2.2 地图绘制模块 9

3.2.3 键盘操作模块 11

3.2.4 游戏规则模块 13

第四章 系统测试 14

4.1 主菜单显示功能的测试演示 14

4.2 键盘操作功能的测试演示 14

4.3 箱子放置到指定位置时变色显示功能的测试演示 17

4.4 游戏胜负判断功能的测试演示 18

第五章 总结 20

参考文献 21

附录 源代码 22

# 概述

## 1.1 研究意义

经典的推箱子是一个来自日本的古老游戏，目的是在训练你的逻辑思考能力。在一个狭小的仓库中，要求把木箱放到指定的位置，稍不小心就会出现箱子无法移动或者通道被堵住的情况，所以需要巧妙的利用有限的空间和通道，合理安排移动的次序和位置，才能顺利的完成任务。

推箱子游戏是非常经典的游戏，类似的游戏你也一定早就玩过。比如后来流行的《切绳子》、《愤怒的小鸟》等游戏，能在生活中给我们带来莫大的快乐，提升我们的大脑运算能力，开发我们的智慧。不仅在PC机上，在手机和很多PSP上都是十分常见而且流行。可以说，这种智力型游戏，给这个社会制造了许多快乐，对儿童、少年的成长起到非常重大的作用。记得几年前，《推箱子》在PC机上刮起了一股不小的益智类游戏的旋风，现在许多资深玩家也都对《推箱子》赞不绝口，可见有深度的益智类游戏是非常受大家欢迎的。

推箱子游戏1981 年由日本人今林宏行首创，是在1982 年12 月由Thinking Rabbit 公司首次发行，名“仓库番”。箱子只可以推, 不可以拉, 而且一次只能推动一个，胜利条件就是把所有的箱子都推到目的地。推箱子游戏是一种老少皆宜的益智游戏，既可以开发青少年学生的智力，又可以防止老年痴呆症，全家一起攻关还可以促进家庭和睦。

## 1.2 开发平台介绍

Visual studio 作为一个主流的开发平台一直深受编程爱好者的喜爱，如果形象点比喻的话，可以 C++看作为一种“工业标准”，而 VC++ 则是某种操作系统平台下的“厂商标准”，而“厂商标准”是在遵循“工业标准”的前提下扩展而来的。Microsoft Visual Studio（简称VS）是美国微软公司的开发工具包系列产品。Visual studio 2017 由Microsoft 于2017年开发, 它不仅是一个C++ 编译器，而且是一个基于Windows操作系统的可视化集成开发环境（integrated development environment ，IDE）。

Visual Studio 2017 正式版不仅添加了实时单元测试、实时架构依赖关系验证等新特性，还对许多实用功能进行了改进，如代码导航、IntelliSense、重构、代码修复和调试等等。无论使用哪种语言或平台，都能节省开发者在日常任务上花费的时间和精力。

此外，该版本还带来了一个新的轻量化和模块化的安装体验，可根据需要量身定制安装。多个增强功能汇集在一起，使 Visual Studio 2017 的启动速度比 Visual Studio 2015 快3倍，解决方案加载时间缩短 2-4倍。

# 第二章 系统整体设计

## 2.1 系统介绍

本系统是一个基于EGE图形库的C风格C++推箱子游戏，游戏使用mmsystem API播放音效，共35关，完美还原经典关卡。

在游戏主界面中，会出现一个小人、若干个箱子和箱子放置点。玩家需要利用方向键控制小人上下左右移动，并推动界面中的箱子到达指定的箱子放置点。

## 2.2 系统所具有的功能

（1）能够显示主菜单和界面

游戏需要提供主菜单让玩家进行游戏设置，同时能够把地图文件中的信息转换成为图像显示到主游戏界面上。

（2）能够实现键盘操作功能

能够接收到键盘输入的W（↑）S（↓）A（←）D（→）等信息，并根据不同的字母方向键把游戏人物移动到相应的位置。例如，当玩家单击W键“上”时，如果向上的位置是可移动的，那么就当把游戏人物向上移动一个方格。

（3）能够把放置到位置上的箱子进行变色显示

当玩家把箱子推到指定位置的格子时，需要把这个箱子进行变色。这样就能明确的显示出该箱子已经放置到指定的位置上。此游戏设计为当推到指定位置的格子时，箱子变为红色。

（4）支持地图扩展功能

管理员可以管理扩展原游戏地图文件，从而生成新的游戏地图。按照Map文件夹中地图文件中数据与图像对照说明进行游戏地图设计。设计完毕后重新打开游戏，选择对应关口即可呈现。

（5）游戏胜负判断功能

在游戏中，当玩家把所有的箱子都推到箱子放置点时，玩家胜利通过当前游戏关口，玩家可以选择进行下一关口的游戏、重新进行当前关口的游戏和退出游戏等操作。

（6）支持关口选择功能

玩家在游戏中可自行选择想要挑战的关口。当玩家选择菜单中“选关（X）、从头开始（A）”，并用W（+）、S（-）按键来选择关口，选择完毕后按ENTER键开始游戏即可打开所选关口。

（7）游戏支持声效功能

通过Boxmove.wav文件，在游戏开始后，玩家在推箱子游戏人物移动过程中会产生声效，充分满足玩家的游戏体验。

（8）游戏撤销步数功能

每关玩家会有三次撤销机会，以返回玩家上次推动箱子前的动作。（注意：撤销不可连续使用，即撤销过一次后再次推动箱子前不可再次撤销！）

## 2.3 系统设计

### 2.3.1 系统功能结构

根据推箱子游戏基本要求以及相关拓展功能，进行游戏结构功能设计，游戏功能结构图如图1 所示。

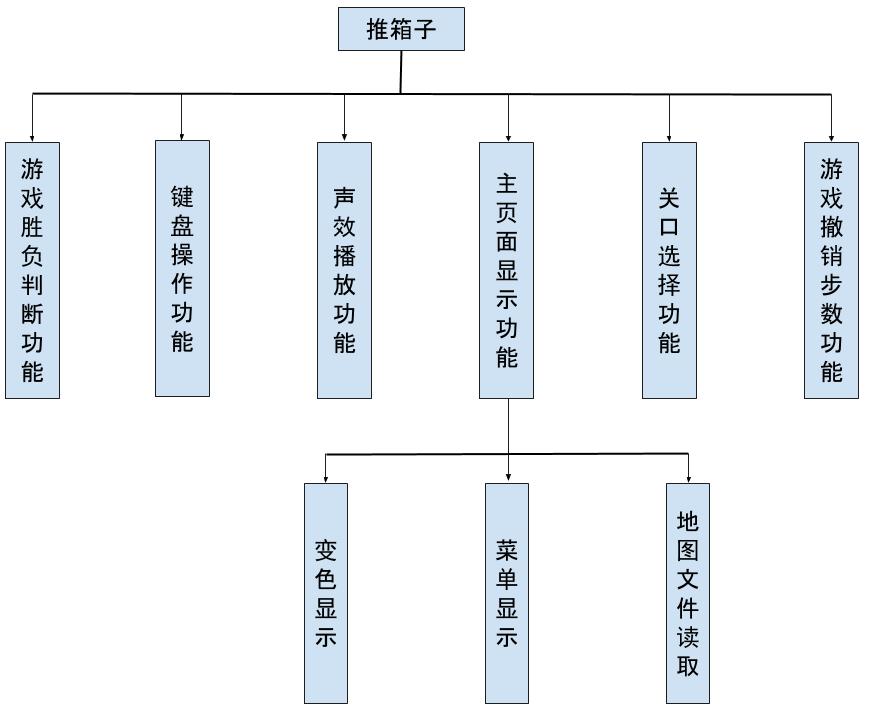


图1 推箱子功能结构图

### 2.3.2 程序结构设计

本设计分为main.cpp，game.cpp，mainloop.cpp三个C++文件和founction.h一个头文件。且分别通过Map、Music、Picture文件夹存放设计所需数据。

游戏共由6个模块组成：

◇主菜单模块：主要负责主界面、菜单及各个界面的创建和调用等处理。

◇键盘操作模块：主要负责接收玩家键盘输入并进行箱子移动等处理。

◇关口选择模块：主要负责游戏挑战关口的选择和设置。

◇地图文件读取模块：主要负责读取地图文件并进行相应的解析工作。

◇游戏规则模块：主要负责游戏规则和胜负的判断。

◇声效播放模块：主要负责游戏中声效的播放。

## 2.4 出错处理设计

（1）出错解决方案一

当游戏中用户发觉自己走错步后，可以采用撤销一步的方法来解决。

（2）出错解决方案二

当游戏中出现方案一无法解决时，即撤销步数达到限制或无法弥补的错误，这些情况可以采用暂停当前游戏并重新开始新游戏的方法处理游戏中的错误。

## 2.5 系统菜单结构设计

推箱子的游戏菜单设计如图2 所示。

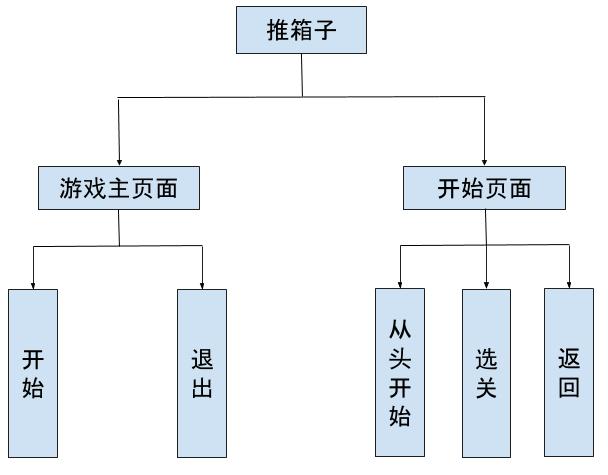


图2 游戏菜单设计图

## 2.6 游戏关口选择的设计

关口选择的设计如图3 所示。

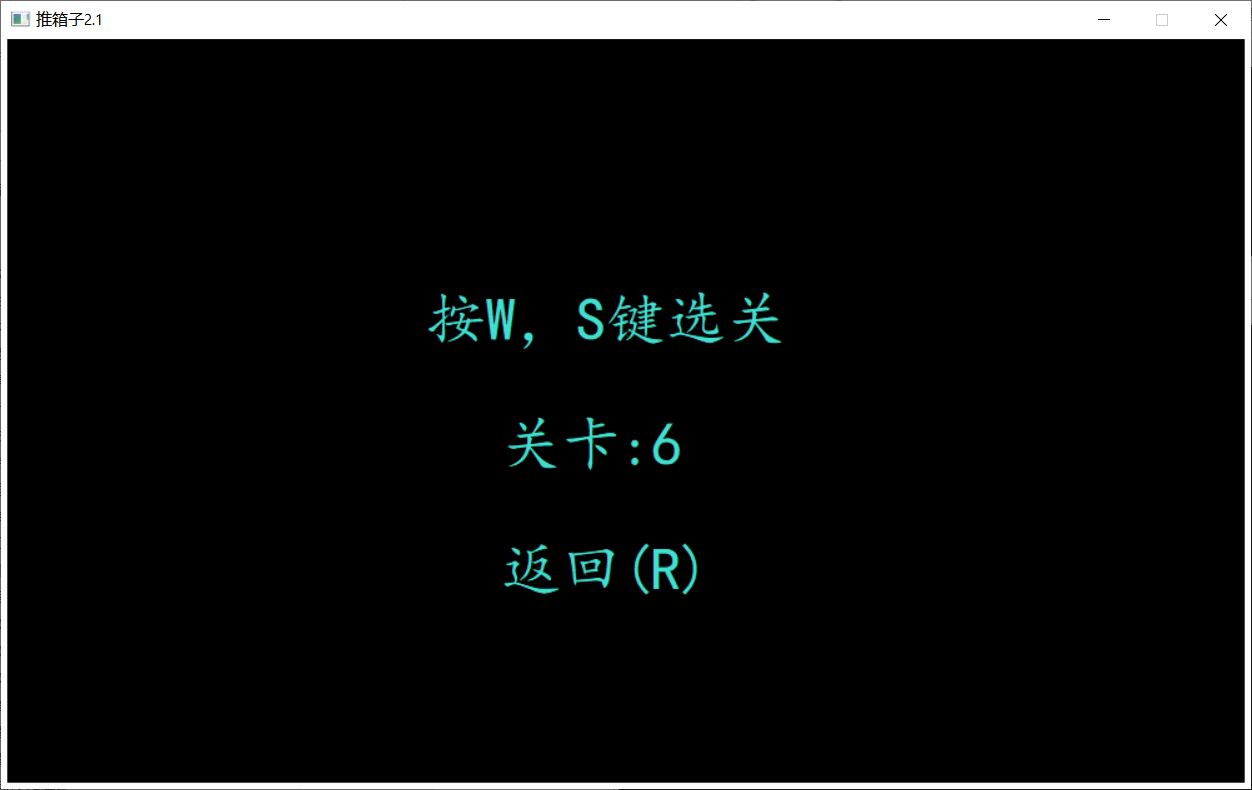


图3 关口选择界面图

当玩家在键盘上输入W（+）S（-）时，并点击Enter键开始游戏时，游戏会自动选择对应关口的地图级别开始新游戏。

# 第三章 系统详细设计

推箱子游戏的Visual C++ 工程采用MFC模式进行开发。本节主要讲解推箱子游戏各个功能模块的代码实现，具体配置在CppProperties文件中。

## 3.1 游戏界面的实现

### 3.1.1 游戏菜单实现

在推箱子游戏中，通过如下几步即可实现游戏的菜单。

1. 在推箱子游戏工程的资源中添加一个菜单资源，其属性如表1 所示

表1 主菜单属性

|  |  |  |
| --- | --- | --- |
| ID | 类别 | 说明 |
| void main\_interface1() | 主菜单 | 游戏的主菜单 |
| void main\_interface2() | 开始菜单 | 主菜单选择开始菜单 |
| void selectLevel(int level) | 选关菜单 | 选关进行游戏 |

1. 给每个菜单添加到mainloop（）函数中。
2. 菜单的实现，各种菜单的调用要运用到的部分代码如下。

void mainloop()

//主程序

while (is\_run())

{

//主菜单1

main\_interface1();

while (is\_run())

{

option = getch();

if (option == 's' || option == 'S' || option == 27)

break;

}

while (is\_run())

{

if (option == 's' || option == 'S')//打印主菜单2

{

main\_interface2();

while (1)

{

option = getch();

if (option == 'a' || option == 'A' || option == 'x' || option == 'X' || option == 'r' || option == 'R')

break;

}

}

if (option == 'n' || option == 'N')//进入下一关

{

system("cls");

level++;

option = gameloop(level);//进入游戏

}

if (option == 'q' || option == 'Q')//重玩本关

{

system("cls");

option = gameloop(level);//进入游戏

}

if (option == 'a' || option == 'A')//从头开始

{

system("cls");

level = 0;

option = gameloop(level);//进入游戏

}

if (option == 'm' || option == 'M' || option == 'r' || option == 'R')//主菜单1

{

break;

}

if (option == 27)//退出程序

break;

}

if (option == 27)//退出程序

break;

}

endmovie();//清屏

Sleep(2000);

}

### 3.1.2 游戏关口选择实现

关口选择界面由W（+）、S（-）来实现，同时要对最大值进行判断。具体代码如下。

if (option == 'x' || option == 'X')//选关

{

level\_option = 1;

while (is\_run())

{

selectLevel(level\_option);

ch = NULL;

ch = getch();

if ((ch == 'w' || ch == 'W') && level\_option < maps\_num)

level\_option++;

if ((ch == 's' || ch == 'S') && level\_option > 1)

level\_option--;

if (ch == 13)

{

option = 'q';

level = level\_option - 1;

break;

}

if (ch == 'r' || ch == 'R')

{

option = 's';

break;

}

}

}

### 3.1.3 游戏声效的实现

声效文件存放在/data/Music/ Boxmove.wav中，实现代码如下：

mciSendString(TEXT("open Boxmove.wav alias Boxmove"), NULL, 0, NULL);

**说明：**MCI(Media Control Interface,媒体控制接口)函数中，有些只适用于一种接口方式，有些则两种接口都可以用。mciSendString只适用于命令串接口方式，用来播放多媒体文件的API指令，可以播放MPEG,AVI,WAV,MP3,等等

## 3.2 游戏的核心算法设计与实现

前面已经对推箱子游戏的菜单的实现进行了阐述，下面的是游戏的核心算法的设计和实现。

### 3.2.1 地图文件读取模块

地图文件读取模块，主要负责将对地图文件进行读取，实现代码如下所示：

void LoadMap(int level,char (\*map)[50])

{

char buffer[256];

FILE \*fp;

sprintf(buffer, "data\\Map\\%d.txt", level+1);

fp = fopen(buffer,"r");

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

{

fgets(map[i], 256, fp);

if (map[i][1] == '|')

break;

}

fclose(fp);

}

### 3.2.2 地图绘制模块

地图绘制模块主要负责将地图数组中的数据绘制成地图图像。地图格式表示如图4所示。

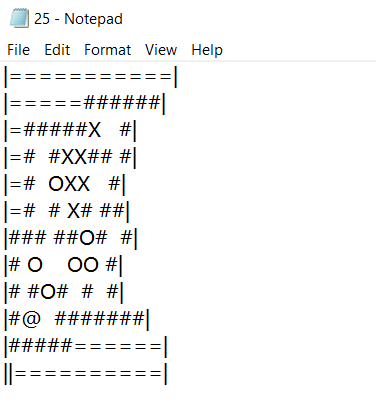


图4 地图TXT文件图

地图绘制模块的实现代码如下。

game\_face(level, step\_num, restep\_num);

for (i = 1; i <= 17; i++)

{

for (t = 0; t < strlen(map[0]); t++)

{

if (map[i][t] == '|' || map[i][t] == '=')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_black);

if (map[i][t] == ' ')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_land);

if (map[i][t] == 'O')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxYellow);

if (map[i][t] == 'Q')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxRed);

if (map[i][t] == '#')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_wall);

if (map[i][t] == 'X')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_aim);

if (map[i][t] == '@')

if (ch == 'd' || ch == 'D')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT2);

}

else if (ch == 'a' || ch == 'A')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT2);

}

else if (ch == 'w' || ch == 'W')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP2);

}

else

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN2);

}

}

if (map[i][1] == '|')

break;

}

### 3.2.3 键盘操作模块

键盘操作模块主要负责接收玩家键盘输入并进行箱子移动、撤销处理、暂停界面、中途退出或重玩等处理，部分实现代码如下。

*//箱子向下移动*

if (kbhit())-+

{

ch = getch();

ch = tolower(ch);

switch (ch)

{

//向下移动

case 's':

if (map[x + 1][y] == ' ' || map[x + 1][y] == 'X')

{

map[x][y] = ' ';

x++;

map[x][y] = '@';

step\_num++;

}

else if ((map[x + 1][y] == 'O' || map[x + 1][y] == 'Q') && map[x + 2][y] != 'O' && map[x + 2][y] != 'Q' && map[x + 2][y] != '#')

{

map[x][y] = ' ';

x++;

map[x][y] = '@';

map[x + 1][y] = 'O';

lstep = 1;

x\_box = x;

y\_box = y;

rerestep\_num = step\_num;

step\_num++;

last\_step = ch;//存储上一步信息

mciSendString(TEXT("seek Boxmove to 0"), NULL, 0, NULL);

mciSendString(TEXT("play Boxmove"), NULL, 0, NULL);//声效设置

}

break;

*//暂停游戏*

case 27://暂停界面

pause();

while (ch != 'r')

{

ch = getch();

if (ch == 'x' || ch == 'q' || ch == 'n' || ch == 'm' || ch == 'X' || ch == 'Q' || ch == 'N' || ch == 'M')

{

option = ch;

ch = 27;

break;

}

}

break;

*//退出&重玩游戏*

//中途退出或重玩

if (ch == 27 || ch == 'q')

{

pass = 0;

break;

}

*//撤销处理*

//撤销一步

if (lstep == 1 && ch == 'b'&&restep\_num < 3)

{

switch (last\_step)

{

//返还向下移动

case 's':

map[x][y] = ' ';

map[x\_box + 1][y\_box] = ' ';

map[x\_box][y\_box] = 'O';

x\_box--;

x = x\_box;

y = y\_box;

map[x][y] = '@';

step\_num = rerestep\_num;

restep\_num++;

lstep = 0;

break;

### 3.2.4 游戏规则模块

游戏规则模块：主要负责游戏规则的判断。其设计方法比较简单，需要在每次玩家移动主角后，对当前地图数组进行判断。如果当前地图中存在单独的“点”或者“主角站在点上”两种情况，就说明玩家未胜利；反之就是胜利。

游戏规则模块的实现，代码如下所示：

for (i = 0; X\_num[i] != 0; i++)

{

if (map[X\_num[i]][Y\_num[i]] == 'Q')

pass++;

}

if (pass == xnum)

{

pass = 1;

ch = 27;

Sleep(1000);

break;

}else

pass = 0;

}

# 第四章 系统测试

## 4.1 主菜单显示功能的测试演示

这个测试用例主要是测试游戏的菜单和界面显示是否成功，根据测试用例文档，其测试步骤如下所示。

1. 程序启动后，其菜单及主界面如图5所示。



图5 推箱子游戏主界面和菜单图

判断结果：游戏的菜单界面显示成功。

## 4.2 键盘操作功能的测试演示

键盘操作功能测试，测试步骤如下。

1. 游戏开始后按下‘S’键进入开始菜单界面，如图6所示。
2. 玩家可以选择‘A’键从头开始闯关，直接进入游戏界面，也可选择‘X’键手动选择关卡。按下‘R’键返回主菜单。如图6所示。
3. 玩家可以选择‘W’键和‘S’键进行关卡的选择，按下‘R’键返回开始菜单界面。选择关卡之后按下‘ENTER’键进入游戏。如图7所示。
4. 游戏开始后的主角位置，如图8 所示。
5. 单击键盘上的“上”方向按键，来控制主角向上移动。
6. 查看主角是否向上移动，如图9 所示。
7. 暂停游戏并测试继续（R），选关（X），重玩（Q），下一关（N），菜单（M）分别进行测试。

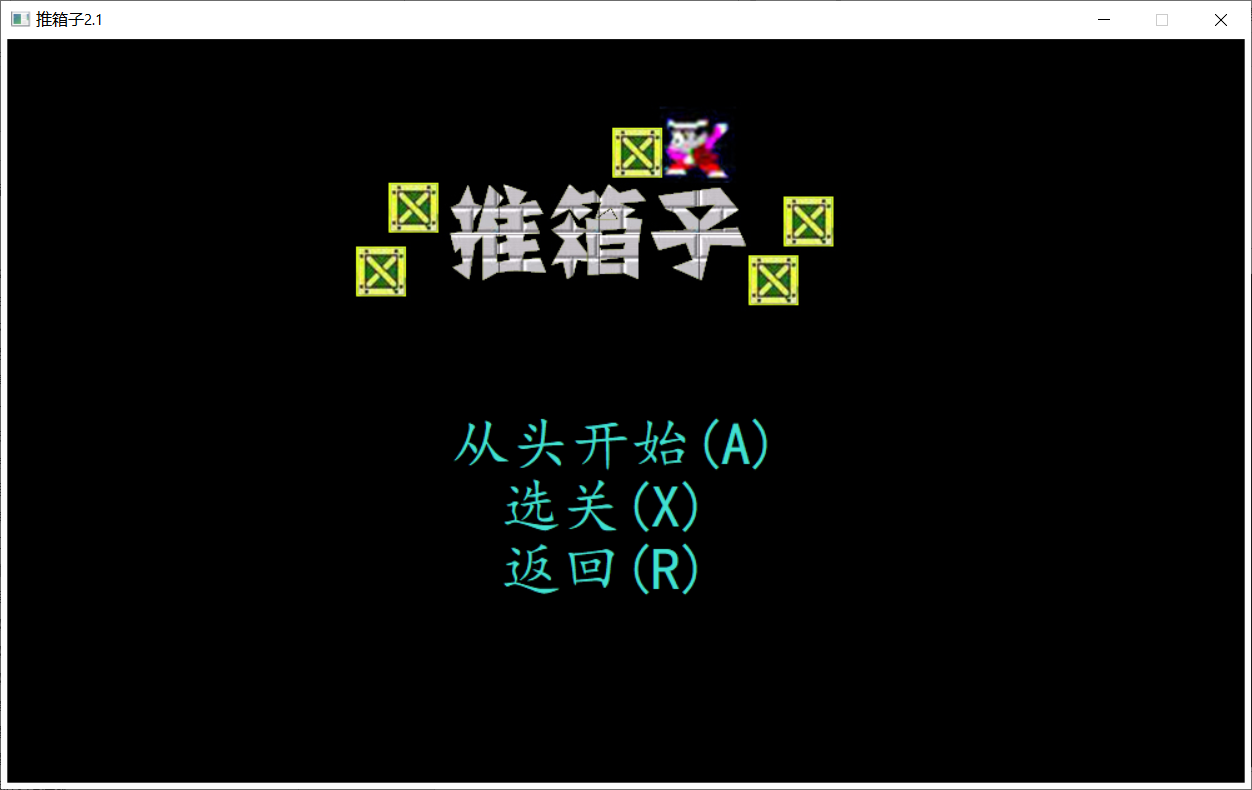


图6 选择菜单界面图

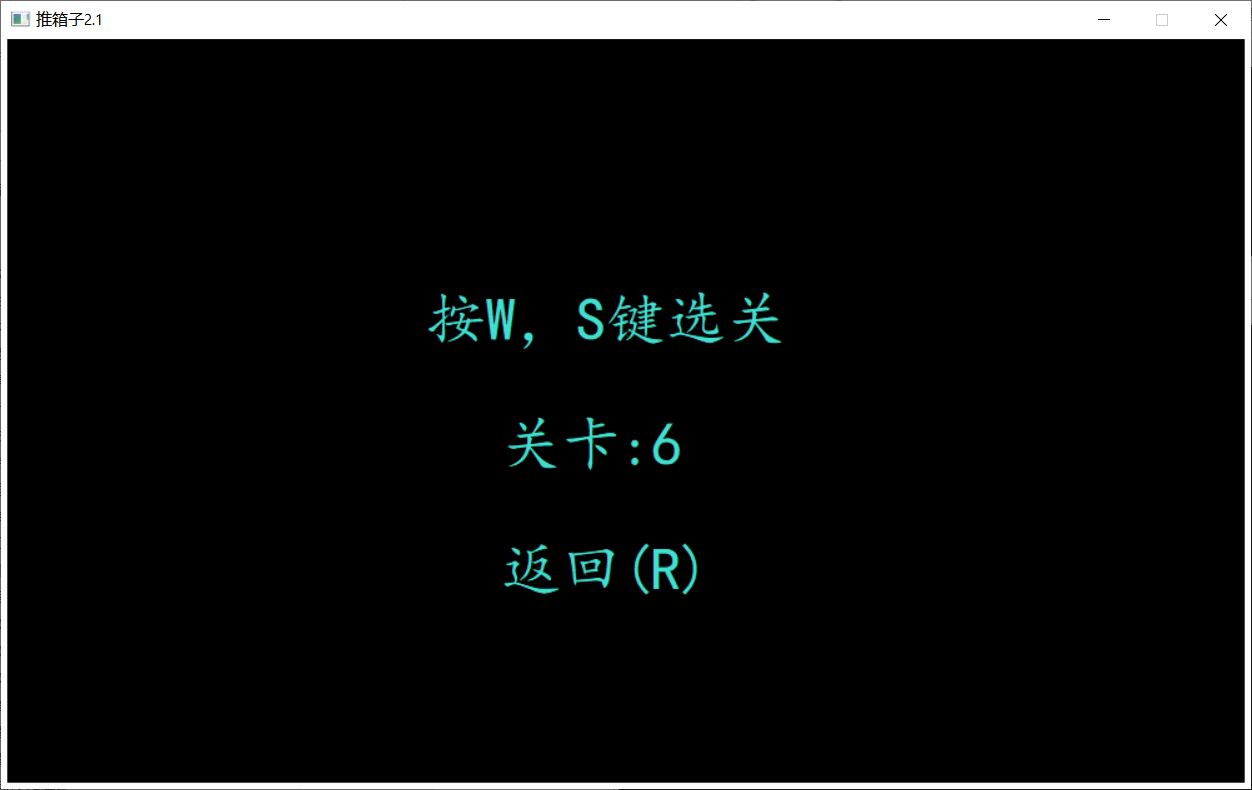


图7 选关菜单界面图

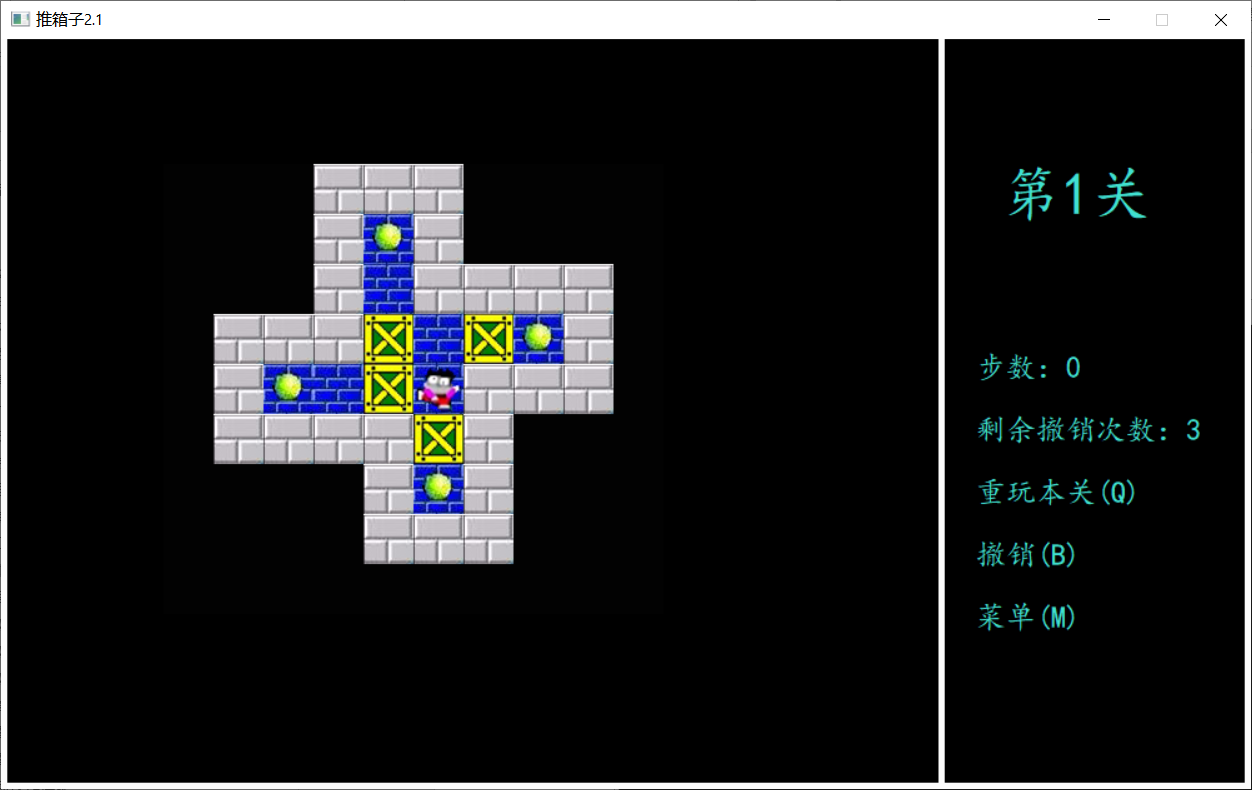


图8 游戏开始界面图

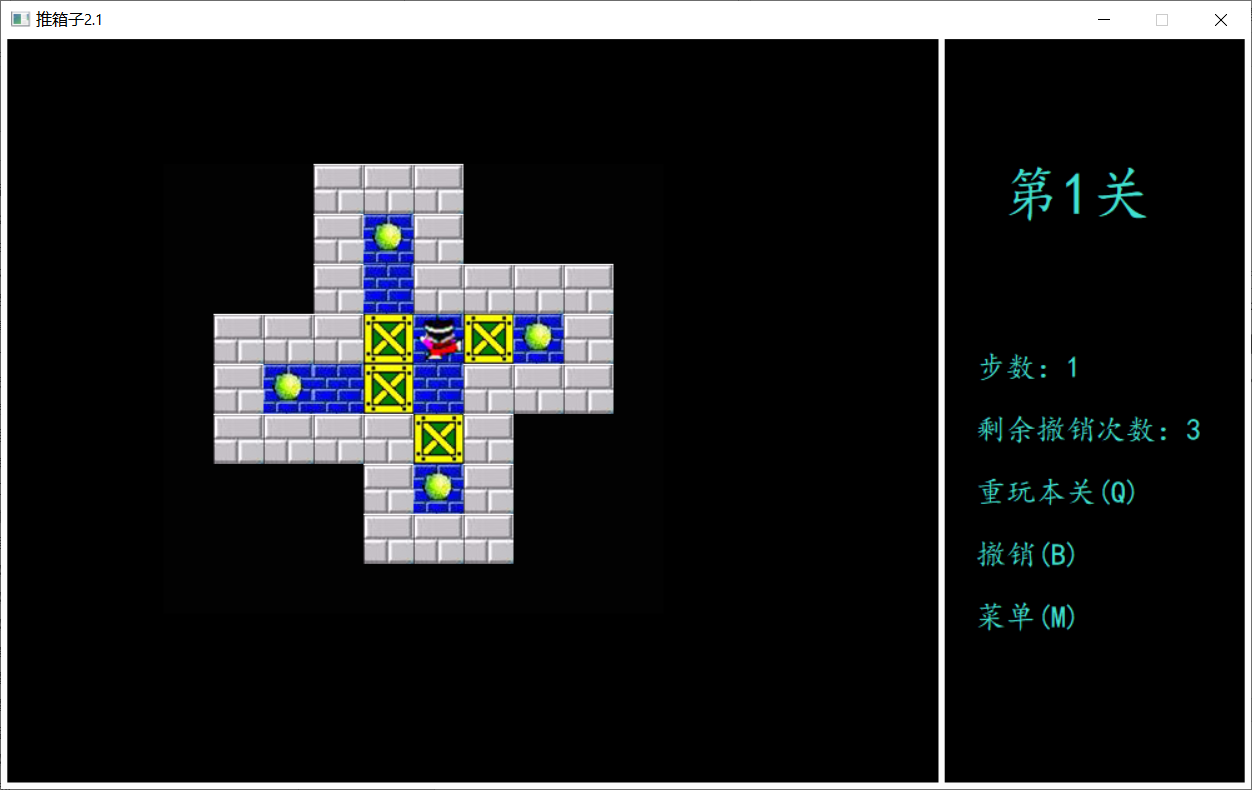


图9 主角向上移动后界面图

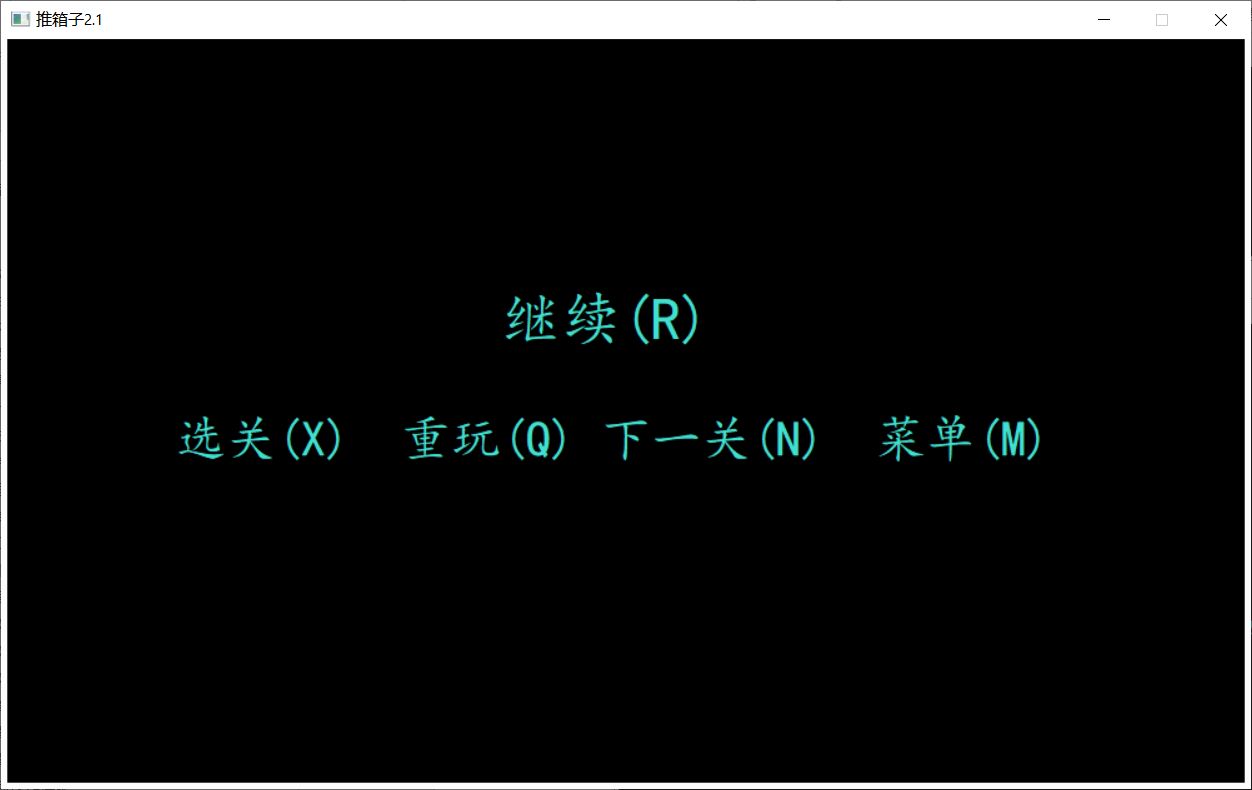


图10 暂停游戏界面图

判断结果：键盘操作功能测试成功。

## 4.3 箱子放置到指定位置时变色显示功能的测试演示

测试将箱子放置到指定位置时，箱子会变色的功能。具体测试步骤如下：

（1） 推箱子游戏已经开始，如图11 所示。

（2） 推动箱子到指定位置（地图中的点位置）后，显示如图12 所示。



图11 关卡初始状态图



图12 箱子到达指定点界面图

判断结果：箱子放置到指定位置时变色显示功能测试成功。

## 4.4 游戏胜负判断功能的测试演示

测试推箱子游戏中游戏胜负判断功能。由于推箱子游戏的特殊性，所以只测试其在玩家胜利时，能否进入下一关，具体测试步骤如下。

（1） 游戏开始后，玩家把将要推的箱子推到指定位置，只剩下最后一个箱子，如图13 所示。

（2） 将最后一个箱子推到指定位置。显示效果如图14 所示。

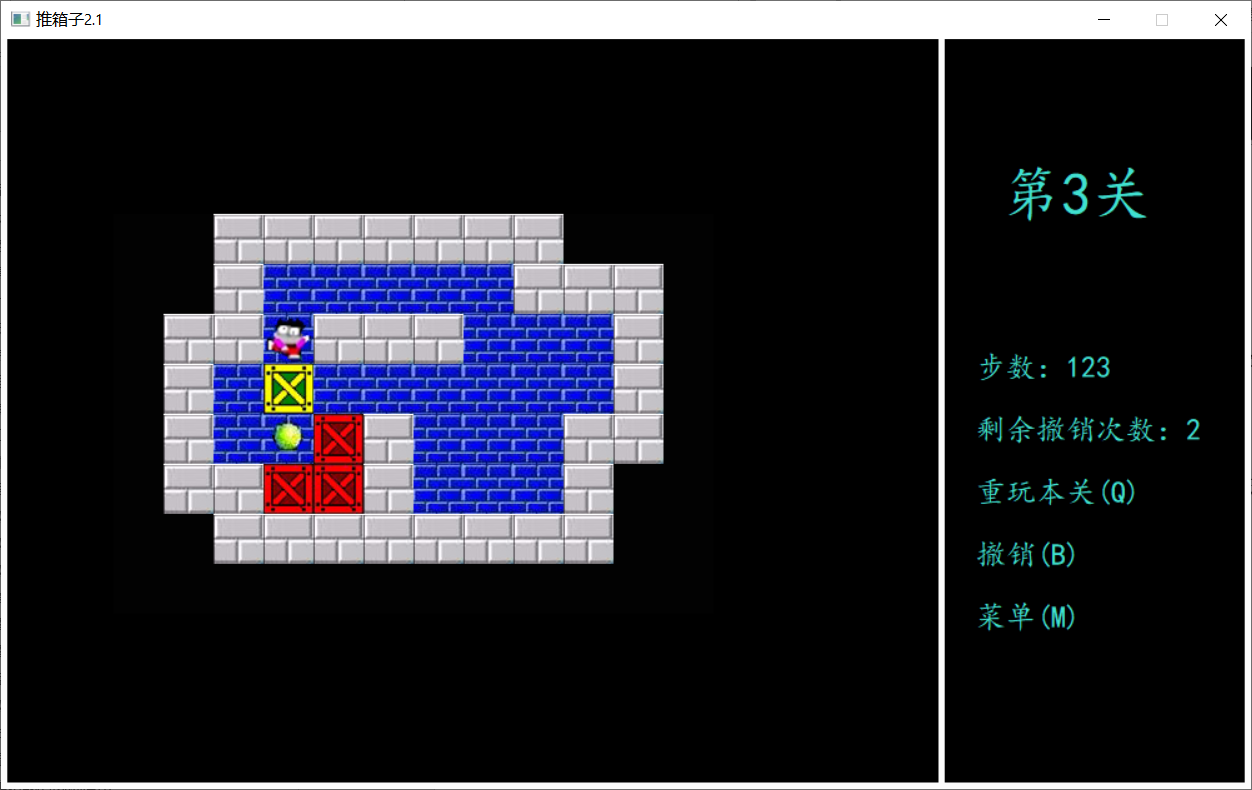


图13 剩余最后一个箱子图

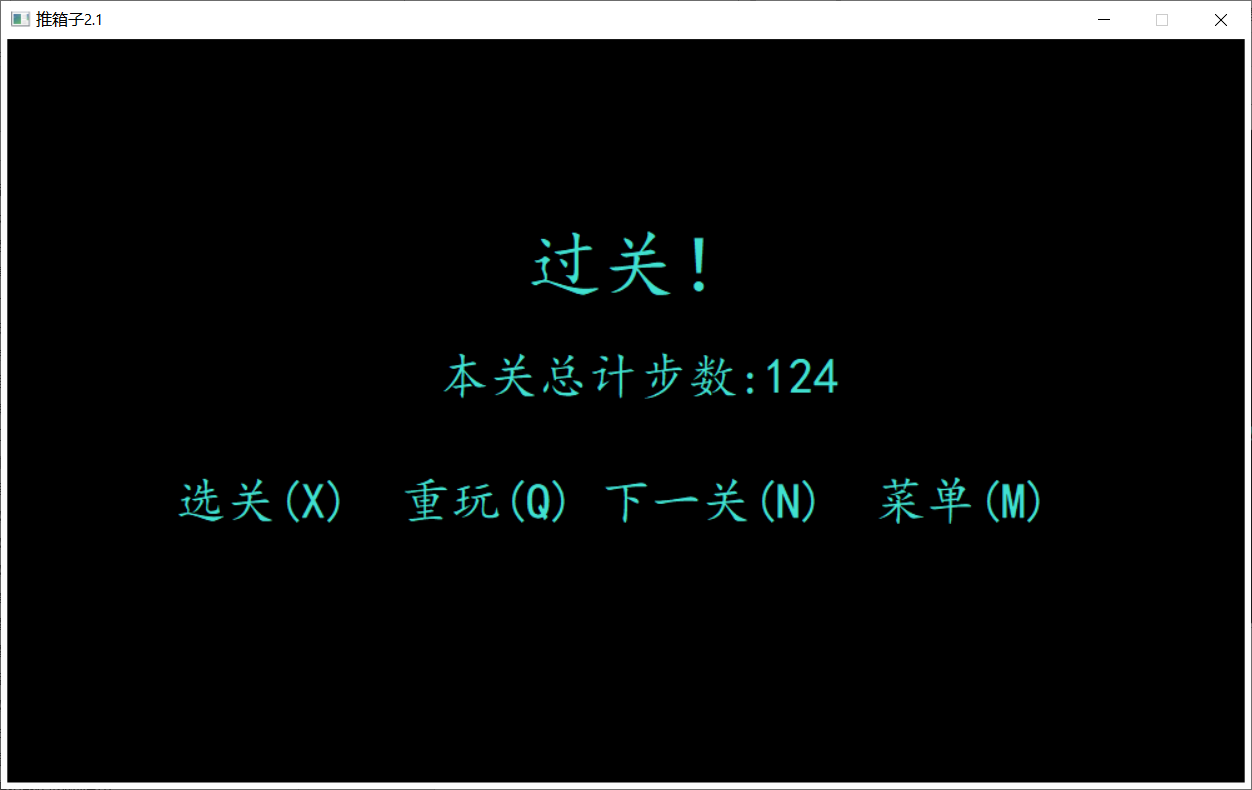


图14 通关界面图

判断结果：推箱子游戏中游戏胜负判断功能正常

# 第五章 总结

本论文阐述了推箱子游戏的设计与实现全过程，并在论文中相应的位置插入了图片、系统设计流程图以及一些编程思想代码，更加清晰的描述了该系统的实现过程。

通过这次课题设计，我查阅了网上大量有关资料以及与同学交流经验，学到了不少知识。虽然经历了不少艰辛，但收获同样巨大，大大提高了动手的能力，充分体会到了在创造过程中探索的艰难和成功时的喜悦。虽然这个设计的功能还有很多进步空间，但是在设计过程中所学到的东西是这次课程设计的最大收获和财富，使我们终身受益。这次设计不仅增强了我的实践动手能力，也让我对课堂上所学到的理论知识的理解加深了许多，这给我提供了一个在学习生活中很难得的理论结合实际的机会。我也会在今后的学习生活中多进行实践操作，积累经验以达到更好的效果。

# 参考文献

[1] 王浩. Visual C++游戏开发案例实战. 北京. 清华大学出版社. 2014.

[2] 刘晓华. Visual C++游戏编程基础. 北京. 电子工业出版社. 2005.

[3] 斯文克. Visual C++ MFC编程实例. 北京. 机械工业出版社. 2000.

[4] 侯俊杰. 深入浅出MFC （第二版）. 武汉. 华中科技大学出版社.2001.

[5] 谭浩强. C++程序设计（第2版）. 北京. 清华大学出版社. 2011.

[6] Stephen Prata. C++ Primer Plus (6th Edition). America. Addison-Wesley Professional. 2011.

[7] Mike Dickheiser. C++ For Game Programmers (Second Edtion). America. Charles River Media. 2006.

[8] Michael Dawson. Beginning C++ Through Game Programming (Fourth Edtion). America.Cengage Learning. 2014.

# 附录 源代码

**# founction.h #**

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#include<math.h>

#include<ege.h>

#include <graphics.h>

#include <mmsystem.h>

#include<string>

#pragma comment(lib,"winmm.lib")

#define maps\_num 35

void mainloop();

char gameloop(int level);

void main\_interface1();

void main\_interface2();

void endmovie();

void selectLevel(int level);

void LoadMap(int level, char(\*map)[50]);

**# game.cpp #**

#include"founction.h"

void game\_face(int level, int step\_num, int restep\_num);

void pause();

void lastlevel();

void passlevel(int step\_num);

char gameloop(int level)

{

mciSendString(TEXT("open Boxmove.wav alias Boxmove"), NULL, 0, NULL);

PIMAGE pimg\_wall = newimage();

PIMAGE pimg\_land = newimage();

PIMAGE pimg\_BoxYellow = newimage();

PIMAGE pimg\_BoxRed = newimage();

PIMAGE pimg\_aim = newimage();

PIMAGE pimg\_ManUP1 = newimage();

PIMAGE pimg\_ManUP2 = newimage();

PIMAGE pimg\_ManDOWN1 = newimage();

PIMAGE pimg\_ManDOWN2 = newimage();

PIMAGE pimg\_ManRIGHT1 = newimage();

PIMAGE pimg\_ManRIGHT2 = newimage();

PIMAGE pimg\_ManLEFT1 = newimage();

PIMAGE pimg\_ManLEFT2 = newimage();

PIMAGE pimg\_black = newimage();

getimage(pimg\_wall, "data\\Picture\\wall.jpg", 0, 0);

getimage(pimg\_land, "data\\Picture\\land.jpg", 0, 0);

getimage(pimg\_BoxYellow, "data\\Picture\\BoxYellow.jpg", 0, 0);

getimage(pimg\_BoxRed, "data\\Picture\\BoxRed.jpg", 0, 0);

getimage(pimg\_aim, "data\\Picture\\aim.jpg", 0, 0);

getimage(pimg\_ManUP1, "data\\Picture\\ManUP1.jpg", 0, 0);

getimage(pimg\_ManUP2, "data\\Picture\\ManUP2.jpg", 0, 0);

getimage(pimg\_ManDOWN1, "data\\Picture\\ManDOWN1.jpg", 0, 0);

getimage(pimg\_ManDOWN2, "data\\Picture\\ManDOWN2.jpg", 0, 0);

getimage(pimg\_ManRIGHT1, "data\\Picture\\ManRIGHT1.jpg", 0, 0);

getimage(pimg\_ManRIGHT2, "data\\Picture\\ManRIGHT2.jpg", 0, 0);

getimage(pimg\_ManLEFT1, "data\\Picture\\ManLEFT1.jpg", 0, 0);

getimage(pimg\_ManLEFT2, "data\\Picture\\ManLEFT2.jpg", 0, 0);

getimage(pimg\_black, "data\\Picture\\black.jpg", 0, 0);

//定义及初始化地图

char map[50][50];

LoadMap(level, map);

//变量申明

//i,t 循环计数器, x 纵坐标, y 横坐标, pass 通关判断条件

//restep\_num 记录每局撤销次数,step\_num 记录本局总步数,wide 地图宽度

int i, t, x, y, pass, restep\_num, step\_num, wide, high, count;

//(ch 游戏过程中读取键盘信息, option暂停过程中读取键盘信息及游戏结束返还值,last\_step 记录上一步数据)

char ch, option, last\_step;

//申明两数组，记录终点坐标

int X\_num[50] = { 0 }, Y\_num[50] = { 0 }, xnum, ynum;

//lstep 记录是否撤销过,x\_box 记录上一次推箱子前箱的纵坐标子,y\_box 记录上一次推箱子前箱的横坐标子, rerestep\_num 记录上一次推箱子前所走的步数

int lstep, x\_box, y\_box, rerestep\_num;

//初始化变量

high = wide = step\_num = restep\_num = pass = 0;

ch = option = last\_step = NULL;

xnum = ynum = 0;

lstep = count = 0;

//初始化起始位置

for (i = 0; i < 20; i++)

{

for (t = 0; map[i][t] != '\0'; t++)

if (map[i][t] == '@')

{

x = i;

y = t;

i = 99;

break;

}

}

//寻找终点坐标

for (i = 2; i < 20; i++)

{

for (t = 2; map[i][t] != '\0'; t++)

{

if (map[i][t] == 'X' || map[i][t] == 'Q')

{

X\_num[xnum] = i;

Y\_num[ynum] = t;

xnum++;

ynum++;

}

if (map[i][1] == '|')

{

i = 99;

break;

}

}

}

//计算地图高度

for (high = 0; high < 20; high++)

if (map[high][0] != '|')

break;

high = (13 - high) / 2;

//计算地图宽度

wide = (16 - strlen(map[0])) / 2;

//打印地图

game\_face(level, step\_num, restep\_num);

for (i = 1; i <= 17; i++)

{

for (t = 0; t < strlen(map[0]); t++)

{

if (map[i][t] == '|' || map[i][t] == '=')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_black);

if (map[i][t] == ' ')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_land);

if (map[i][t] == 'O')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxYellow);

if (map[i][t] == 'Q')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxRed);

if (map[i][t] == '#')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_wall);

if (map[i][t] == 'X')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_aim);

if (map[i][t] == '@')

if (ch == 'd' || ch == 'D')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT2);

}

else if (ch == 'a' || ch == 'A')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT2);

}

else if (ch == 'w' || ch == 'W')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP2);

}

else

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN2);

}

}

if (map[i][1] == '|')

break;

}

//游戏主程序

for (; is\_run(); delay\_fps(60))

{

fflush(stdin);

Sleep(100);

// todo: 逻辑更新

if (kbhit())

{

ch = getch();

ch = tolower(ch);

switch (ch)

{

//向下移动

case 's':

if (map[x + 1][y] == ' ' || map[x + 1][y] == 'X')

{

map[x][y] = ' ';

x++;

map[x][y] = '@';

step\_num++;

}

else if ((map[x + 1][y] == 'O' || map[x + 1][y] == 'Q') && map[x + 2][y] != 'O' && map[x + 2][y] != 'Q' && map[x + 2][y] != '#')

{

map[x][y] = ' ';

x++;

map[x][y] = '@';

map[x + 1][y] = 'O';

lstep = 1;

x\_box = x;

y\_box = y;

rerestep\_num = step\_num;

step\_num++;

last\_step = ch;//存储上一步信息

mciSendString(TEXT("seek Boxmove to 0"), NULL, 0, NULL);

mciSendString(TEXT("play Boxmove"), NULL, 0, NULL);

}

break;

//向上移动

case 'w':

if (map[x - 1][y] == ' ' || map[x - 1][y] == 'X')

{

map[x][y] = ' ';

x--;

map[x][y] = '@';

step\_num++;

}

else if ((map[x - 1][y] == 'O' || map[x - 1][y] == 'Q') && map[x - 2][y] != 'O' && map[x - 2][y] != 'Q' && map[x - 2][y] != '#')

{

map[x][y] = ' ';

x--;

map[x][y] = '@';

map[x - 1][y] = 'O';

lstep = 1;

x\_box = x;

y\_box = y;

rerestep\_num = step\_num;

step\_num++;

last\_step = ch;//存储上一步信息

mciSendString(TEXT("seek Boxmove to 0"), NULL, 0, NULL);

mciSendString(TEXT("play Boxmove"), NULL, 0, NULL);

}

break;

//向左移动

case 'a':

if (map[x][y - 1] == ' ' || map[x][y - 1] == 'X')

{

map[x][y] = ' ';

y--;

map[x][y] = '@';

step\_num++;

}

else if ((map[x][y - 1] == 'O' || map[x][y - 1] == 'Q') && map[x][y - 2] != 'O' && map[x][y - 2] != 'Q' && map[x][y - 2] != '#')

{

map[x][y] = ' ';

y--;

map[x][y] = '@';

map[x][y - 1] = 'O';

lstep = 1;

x\_box = x;

y\_box = y;

rerestep\_num = step\_num;

step\_num++;

last\_step = ch;//存储上一步信息

mciSendString(TEXT("seek Boxmove to 0"), NULL, 0, NULL);

mciSendString(TEXT("play Boxmove"), NULL, 0, NULL);

}

break;

//向右移动

case 'd':

if (map[x][y + 1] == ' ' || map[x][y + 1] == 'X')

{

map[x][y] = ' ';

y++;

map[x][y] = '@';

step\_num++;

}

else if ((map[x][y + 1] == 'O' || map[x][y + 1] == 'Q') && map[x][y + 2] != 'O' && map[x][y + 2] != 'Q'&&map[x][y + 2] != '#')

{

map[x][y] = ' ';

y++;

map[x][y] = '@';

map[x][y + 1] = 'O';

lstep = 1;

x\_box = x;

y\_box = y;

rerestep\_num = step\_num;

step\_num++;

last\_step = ch;//存储上一步信息

mciSendString(TEXT("seek Boxmove to 0"), NULL, 0, NULL);

mciSendString(TEXT("play Boxmove"), NULL, 0, NULL);

}

break;

case 27://暂停界面

pause();

while (ch != 'r')

{

ch = getch();

if (ch == 'x' || ch == 'q' || ch == 'n' || ch == 'm' || ch == 'X' || ch == 'Q' || ch == 'N' || ch == 'M')

{

option = ch;

ch = 27;

break;

}

}

break;

case 'q'://重玩

option = ch;

break;

}

}

//撤销一步

if (lstep == 1 && ch == 'b'&&restep\_num < 3)

{

switch (last\_step)

{

//返还向下移动

case 's':

map[x][y] = ' ';

map[x\_box + 1][y\_box] = ' ';

map[x\_box][y\_box] = 'O';

x\_box--;

x = x\_box;

y = y\_box;

map[x][y] = '@';

step\_num = rerestep\_num;

restep\_num++;

lstep = 0;

break;

//返还向上移动

case 'w':

map[x][y] = ' ';

map[x\_box - 1][y\_box] = ' ';

map[x\_box][y\_box] = 'O';

x\_box++;

x = x\_box;

y = y\_box;

map[x][y] = '@';

step\_num = rerestep\_num;

restep\_num++;

lstep = 0;

break;

//返还向左移动

case 'a':

map[x][y] = ' ';

map[x\_box][y\_box - 1] = ' ';

map[x\_box][y\_box] = 'O';

y\_box++;

x = x\_box;

y = y\_box;

map[x][y] = '@';

step\_num = rerestep\_num;

restep\_num++;

lstep = 0;

break;

//返还向右移动

case 'd':

map[x][y] = ' ';

map[x\_box][y\_box + 1] = ' ';

map[x\_box][y\_box] = 'O';

y\_box--;

x = x\_box;

y = y\_box;

map[x][y] = '@';

step\_num = rerestep\_num;

restep\_num++;

lstep = 0;

break;

}

}

//if (lstep = 1)

// last\_step = ch;//存储上一步信息

//还原终点标记

for (i = 0; X\_num[i] != 0; i++)

{

if (map[X\_num[i]][Y\_num[i]] == ' ')

map[X\_num[i]][Y\_num[i]] = 'X';

if (map[X\_num[i]][Y\_num[i]] == 'O')

map[X\_num[i]][Y\_num[i]] = 'Q';

}

//中途退出或重玩

if (ch == 27 || ch == 'q')

{

pass = 0;

break;

}

count++;

if (count == 99)

count = 0;

// todo: 图形更新

// 清屏

cleardevice();

//打印地图

game\_face(level, step\_num, restep\_num);

for (i = 1; i <= 17; i++)

{

for (t = 0; t < strlen(map[0]); t++)

{

if (map[i][t] == '|' || map[i][t] == '=')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_black);

if (map[i][t] == ' ')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_land);

if (map[i][t] == 'O')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxYellow);

if (map[i][t] == 'Q')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_BoxRed);

if (map[i][t] == '#')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_wall);

if (map[i][t] == 'X')

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_aim);

if (map[i][t] == '@')

if (ch == 'd' || ch == 'D')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManRIGHT2);

}

else if (ch == 'a' || ch == 'A')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManLEFT2);

}

else if (ch == 'w' || ch == 'W')

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManUP2);

}

else

{

if (count % 5 > 0)

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN1);

else

putimage(50 + t \* 40 + wide \* 40, 20 + i \* 40 + high \* 40, pimg\_ManDOWN2);

}

}

if (map[i][1] == '|')

break;

}

//通关条件判定

for (i = 0; X\_num[i] != 0; i++)

{

if (map[X\_num[i]][Y\_num[i]] == 'Q')

pass++;

}

if (pass == xnum)

{

pass = 1;

ch = 27;

Sleep(1000);

break;

}

else

pass = 0;

}

//判断能否继续下一关

cleardevice();

if (option != NULL)

{

if (option == 'n')

{

if (level == maps\_num - 1)

{

lastlevel();

option = 'p';

Sleep(1500);

system("cls");

}

}

delimage(pimg\_wall);

delimage(pimg\_land);

delimage(pimg\_BoxYellow);

delimage(pimg\_BoxRed);

delimage(pimg\_aim);

delimage(pimg\_ManUP1);

delimage(pimg\_ManUP2);

delimage(pimg\_ManDOWN1);

delimage(pimg\_ManDOWN2);

delimage(pimg\_ManRIGHT1);

delimage(pimg\_ManRIGHT2);

delimage(pimg\_ManLEFT1);

delimage(pimg\_ManLEFT2);

delimage(pimg\_black);

return option;

}

//通关界面

if (pass == 1)

{

passlevel(step\_num);

}

while (pass == 1)

{

option = getch();

if (option == 27)

option = 'm';

if (option == 'n' || option == 'N')

{

if (level == maps\_num - 1)

{

lastlevel();

option = 'p';

Sleep(1500);

cleardevice();

}

}

if (option == 'x' || option == 'q' || option == 'n' || option == 'p' || option == 'm' || option == 'X' || option == 'Q' || option == 'N' || option == 'P' || option == 'M')

{

cleardevice();

delimage(pimg\_wall);

delimage(pimg\_land);

delimage(pimg\_BoxYellow);

delimage(pimg\_BoxRed);

delimage(pimg\_aim);

delimage(pimg\_ManUP1);

delimage(pimg\_ManUP2);

delimage(pimg\_ManDOWN1);

delimage(pimg\_ManDOWN2);

delimage(pimg\_ManRIGHT1);

delimage(pimg\_ManRIGHT2);

delimage(pimg\_ManLEFT1);

delimage(pimg\_ManLEFT2);

delimage(pimg\_black);

return option;

}

}

delimage(pimg\_wall);

delimage(pimg\_land);

delimage(pimg\_BoxYellow);

delimage(pimg\_BoxRed);

delimage(pimg\_aim);

delimage(pimg\_ManUP1);

delimage(pimg\_ManUP2);

delimage(pimg\_ManDOWN1);

delimage(pimg\_ManDOWN2);

delimage(pimg\_ManRIGHT1);

delimage(pimg\_ManRIGHT2);

delimage(pimg\_ManLEFT1);

delimage(pimg\_ManLEFT2);

delimage(pimg\_black);

return option;

}

void game\_face(int level, int step\_num, int restep\_num)

{

cleardevice();

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfillcolor(WHITE);

bar(750, 0, 755, 700);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

setfont(48, 0, "楷体");

setbkmode(TRANSPARENT);

char str[20];

sprintf(str, "第%d关", level + 1);

outtextxy(800, 100, str);

setfont(24, 0, "楷体");

sprintf(str, "步数：%d", step\_num);

outtextxy(780, 250, str);

sprintf(str, "剩余撤销次数：%d", 3 - restep\_num);

outtextxy(780, 300, str);

outtextxy(780, 350, "重玩本关(Q)");

outtextxy(780, 400, "撤销(B)");

outtextxy(780, 450, "菜单(M)");

}

void pause()

{

cleardevice();

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

setfont(48, 0, "楷体");

setbkmode(TRANSPARENT);

outtextxy(400, 200, "继续(R)");

setfont(40, 0, "楷体");

outtextxy(140, 300, "选关(X)");

outtextxy(320, 300, "重玩(Q)");

outtextxy(480, 300, "下一关(N)");

outtextxy(700, 300, "菜单(M)");

}

void lastlevel()

{

cleardevice();

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

setfont(48, 0, "楷体");

setbkmode(TRANSPARENT);

outtextxy(280, 250, "这已经是最后一关了！");

}

void passlevel(int step\_num)

{

cleardevice();

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

setfont(60, 0, "楷体");

setbkmode(TRANSPARENT);

outtextxy(420, 150, "过关！");

setfont(40, 0, "楷体");

char str[20];

sprintf(str, "本关总计步数:%d", step\_num);

outtextxy(350, 250, str);

setfont(40, 0, "楷体");

outtextxy(140, 350, "选关(X)");

outtextxy(320, 350, "重玩(Q)");

outtextxy(480, 350, "下一关(N)");

outtextxy(700, 350, "菜单(M)");

}

void LoadMap(int level,char (\*map)[50])

{

char buffer[256];

FILE \*fp;

sprintf(buffer, "data\\Map\\%d.txt", level+1);

fp = fopen(buffer,"r");

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

{

fgets(map[i], 256, fp);

if (map[i][1] == '|')

break;

}

fclose(fp);

}

**# mainloop.cpp #**

#include"founction.h"

void mainloop()

{

int level, level\_option;

char option, ch;

level = 0;

option = 'M';

ch = NULL;

//主程序

while (is\_run())

{

//主菜单1

main\_interface1();

while (is\_run())

{

option = getch();

if (option == 's' || option == 'S' || option == 27)

break;

}

while (is\_run())

{

if (option == 's' || option == 'S')//打印主菜单2

{

main\_interface2();

while (1)

{

option = getch();

if (option == 'a' || option == 'A' || option == 'x' || option == 'X' || option == 'r' || option == 'R')

break;

}

}

if (option == 'n' || option == 'N')//进入下一关

{

system("cls");

level++;

option = gameloop(level);//进入游戏

}

if (option == 'q' || option == 'Q')//重玩本关

{

system("cls");

option = gameloop(level);//进入游戏

}

if (option == 'a' || option == 'A')//从头开始

{

system("cls");

level = 0;

option = gameloop(level);//进入游戏

}

if (option == 'x' || option == 'X')//选关

{

level\_option = 1;

while (is\_run())

{

selectLevel(level\_option);

ch = NULL;

ch = getch();

if ((ch == 'w' || ch == 'W') && level\_option < maps\_num)

level\_option++;

if ((ch == 's' || ch == 'S') && level\_option > 1)

level\_option--;

if (ch == 13)

{

option = 'q';

level = level\_option - 1;

break;

}

if (ch == 'r' || ch == 'R')

{

option = 's';

break;

}

}

}

if (option == 'p' || option == 'P')//通关重置数据

{

option = 'm';

level = 0;

break;

}

if (option == 'm' || option == 'M' || option == 'r' || option == 'R')//主菜单1选择

{

break;

}

if (option == 27)//退出程序

break;

}

if (option == 27)//退出程序

break;

}

endmovie();

Sleep(2000);

}

void main\_interface1()

{

PIMAGE pimg\_logo = newimage();

getimage(pimg\_logo, "data\\Picture\\logo.jpg", 0, 0);

//清屏

cleardevice();

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

//文字背景色（注意setbkcolor函数也会同时改变文字背景色）

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

setbkmode(TRANSPARENT);

putimage(280, 50, pimg\_logo);

setfont(48, 0, "楷体");

outtextxy(400, 300, "开始(S)");

outtextxy(380, 400, "退出(ESC)");

delimage(pimg\_logo);

}

void main\_interface2()

{

PIMAGE pimg\_logo = newimage();

getimage(pimg\_logo, "data\\Picture\\logo.jpg", 0, 0);

//清屏

cleardevice();

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

//文字背景色（注意setbkcolor函数也会同时改变文字背景色）

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

//设置字体，第一个参数是字体的高度（像素），第二个参数是字体的宽度，第二个参数如果为0，就使用默认比例值

setfont(80, 0, "楷体");

//要使用特殊格式化字符请用outtextrect

//设置文字背景填充方式为透明，默认为OPAQUE不透明

setbkmode(TRANSPARENT);

putimage(280, 50, pimg\_logo);

setfont(48, 0, "楷体");

outtextxy(360, 300, "从头开始(A)");

outtextxy(400, 350, "选关(X)");

outtextxy(400, 400, "返回(R)");

delimage(pimg\_logo);

}

void endmovie()

{

//清屏

cleardevice();

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

//文字背景色（注意setbkcolor函数也会同时改变文字背景色）

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

//设置字体，第一个参数是字体的高度（像素），第二个参数是字体的宽度，第二个参数如果为0，就使用默认比例值

//如果高度为12，即相当于小五号字，或者9磅字，实际的换算就自己完成吧

setfont(200, 0, "楷体");

//写文字，注意：outtextxy不支持\t \n这类格式化用的特殊字符，这类字符会被忽略

//要使用特殊格式化字符请用outtextrect

//设置文字背景填充方式为透明，默认为OPAQUE不透明

setbkmode(TRANSPARENT);

outtextxy(330, 200, "SIRI");

}

void selectLevel(int level)

{

//清屏

cleardevice();

setcolor(EGERGB(0x40, 0xE0, 0xD0));

setfillcolor(WHITE);

bar(0, 595, 1000, 600);

bar(0, 0, 5, 600);

bar(995, 0, 1000, 600);

//文字背景色（注意setbkcolor函数也会同时改变文字背景色）

setfontbkcolor(EGERGB(0x80, 0x00, 0x80));

//设置字体，第一个参数是字体的高度（像素），第二个参数是字体的宽度，第二个参数如果为0，就使用默认比例值

//如果高度为12，即相当于小五号字，或者9磅字，实际的换算就自己完成吧

setfont(48, 0, "楷体");

//写文字，注意：outtextxy不支持\t \n这类格式化用的特殊字符，这类字符会被忽略

//要使用特殊格式化字符请用outtextrect

//设置文字背景填充方式为透明，默认为OPAQUE不透明

setbkmode(TRANSPARENT);

outtextxy(340, 200, "按W，S键选关");

char str[20];

sprintf(str, "关卡:%d", level);

outtextxy(400, 300, str);

outtextxy(400, 400, "返回(R)");

}

**# main.cpp #**

#include"founction.h"

//窗口尺寸

#define Win\_L 1000

#define Win\_H 600

int main(void)

{

ege::setinitmode(ege::INIT\_ANIMATION);

// 图形初始化

ege::initgraph(Win\_L, Win\_H);

setcaption("推箱子2.1"); //修改窗口名称

// 随机数初始化，如果需要使用随机数的话

ege::randomize();

// 程序主循环

mainloop();

// 关闭绘图设备

ege::closegrap

();

return 0;

}