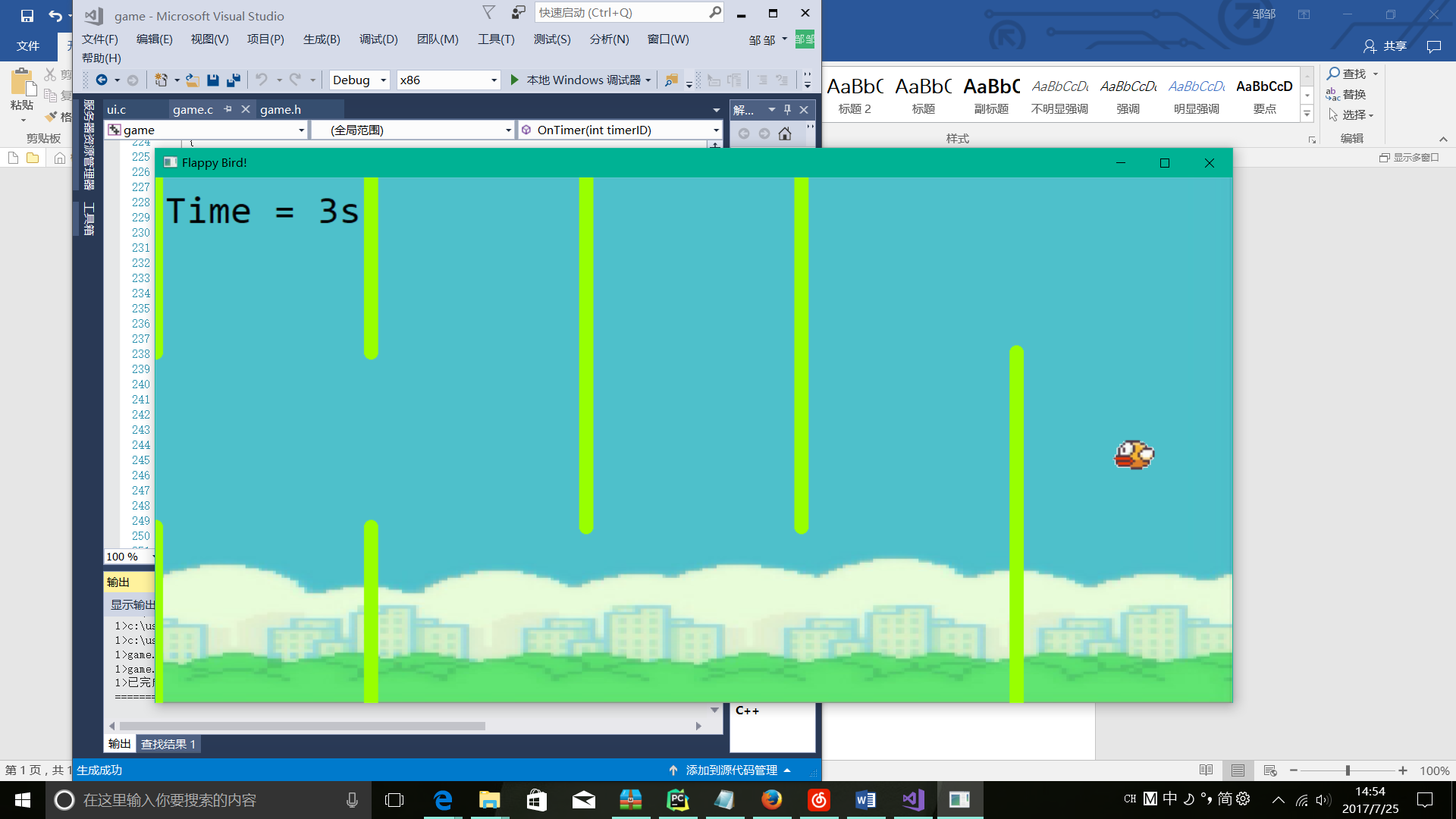
C语言实验报告

一、简介：

Flappy bird 是一款卷轴类的躲避游戏。设计中可用空格操控游戏的开始和暂停，通过控制向上的方向使小鸟不断向上飞，但是不能落到地面和上边界也不能撞到障碍物，否则游戏结束弹出“Game Over！”的窗口。游戏可以通过坚持的时间控制障碍物移动的速度，从而加大游戏难度。



游戏界面

二、障碍物的设计：

障碍物是用Windows下API的笔画出来的，主要的移动和存储是用二维数组实现的。墙的缺口实现是用二维数组将整个窗口纵向的分为了三个部分，用布尔类型有缺口为True，无缺口为False，每个障碍物都随机生成一个True。移动就是坐标的移动，障碍物的横坐标加上移动速度，事实上在整个游戏中只有五根障碍物，当障碍物的横坐标大于窗口的宽度时移到开始位置重新生成缺口。具体算法如下：

if (wallX >= width) //墙超出窗口回到初始位置

{

wallX = width \* (Num - 1) / Num; //最右侧墙的横坐标，距离右边缘N分之一宽度的位置，规定位置

for (i = 0; i < Num - 1; ++i) //障碍物移动后后一个到达前一个的位置

{

for (j = 0; j < 3; ++j)

{

walls[i][j] = walls[i + 1][j];

}

}

for (j = 0; j < 3; ++j) //移动后的随机缺口

{

walls[Num - 1][j] = TRUE;

}

walls[Num - 1][rand() % 3] = FALSE;

}

}

三、小鸟的设计：

画小鸟是用贴位图实现的，小鸟的跳动需要计时器，其实就是设置了初始速度和加速度，通过时间段内的位移改变小鸟的纵坐标实现的。小鸟事实上只能上下移动，它的飞翔是通过障碍物的移动造成错觉实现的。具体算法如下：

SetTimer(hwnd, JUMP\_TIMER, TimerInitInterval, NULL);

dh = v \* t + 1 / 2 \* g \* t \* t;

v += g \* t;

point\_bird.y += dh;

四、碰撞问题：

碰撞问题就是判断小鸟和障碍物的坐标是否重合，但是由于障碍物是由二维数组存储的而小鸟的纵坐标为位移，所以把窗口分为上中下三个区域，再用坐标判断，具体算法如下：

//分屏，存小鸟，判断小鸟的区域

if (point\_bird.y < 0)

{

bird\_pos = 0;

dead = TRUE;

}

else if (point\_bird.y > 0 && point\_bird.y < height / 3)

{

bird\_pos = 0;

}

else if (point\_bird.y < height \* 2 / 3)

{

bird\_pos = 1;

}

else if (point\_bird.y < height)

{

bird\_pos = 2;

}

else

{

bird\_pos >= 2;

dead = TRUE;

}

for (i = 0; i < Num; ++i) //判断有没有撞到障碍物（坐标重合）

{

if (walls[i][bird\_pos] && BirdX <= wallX - i \* wallDistance && BirdX + BirdWidth >= wallX - i \* wallDistance)

{

dead = TRUE;

}

}

五、游戏界面设计

游戏的界面设计主要参考了飞机大战的ui.c，背景图和小鸟的图片是从网上找的，画背景时用了缩放的函数

StretchBlt(hdcMem, rect.left, rect.top,

rect.right - rect.left, rect.bottom - rect.top,

hdcmem\_BK,

0, 0,

bmp\_bk.bmWidth,

bmp\_bk.bmHeight,

SRCCOPY);

画障碍物时是用for循环和poly数组画直线

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

{

for (j = 0; j < 3; ++j)

{

//从右往左第i列，从上往下第j个

if (walls[i][j])

{

poly[0].x = poly[1].x = rect.left + wallX - i \* wallDistance; //表示竖直线段

poly[0].y = rect.top + j \* wallLength;

poly[1].y = rect.top + (j + 1) \* wallLength;

Polyline(hdcMem, poly, 2);

}

}

}

画小鸟时用了TransparentBlt(hdcMem, BirdX, point\_bird.y,

BirdWidth, BirdWidth,

hdcmem\_Bird,

0, 0,

bmp\_bird.bmWidth,

bmp\_bird.bmHeight,

RGB(255, 255, 255));函数去除小鸟周围的白色背景，并用CreateFont(48, 0, 0, 0, FW\_DONTCARE, 0, 0, 0, DEFAULT\_CHARSET, OUT\_OUTLINE\_PRECIS,

CLIP\_DEFAULT\_PRECIS, CLEARTYPE\_QUALITY, VARIABLE\_PITCH, TEXT("Consolas"));显示了游戏的时间用于增加游戏的难度。

六、总结：

这个游戏是自己第一次完全自己一个人编C语言的大程序，总的来说自己的逻辑思维能力还是有些欠缺，一开始写游戏的时候有点乱，分不清应该先写什么后写什么，前面写的东西和后面要写的有冲突，只能重写。对API一开始不能很好的理解和运用。改的最痛苦的是针对游戏运行一段时间的黑屏问题。一开始以为是delete有问题但是检查后没有忘记delete的hdc，然后以为是把loadimage放在paint里的问题，专门分开写发现连背景图都不见了，最后是由于头文件的重名问题句柄没有传入正确的数值，还好最后解决了这个问题。然后难写一点的是碰撞问题，总是会有bug出现，一开始是情况判断不准确，忘记把小鸟的宽度加在判断条件里。不管怎样游戏最终是做出来了，难忘的一次经历，当然也发现了自己当时学习时的一些纰漏和不足，还是很有意义的。通过课程设计，普遍感到自己应用基础知识及专业知识解决问题的能力有了很大的提高，以前所学的许多课程觉得很零散，也不知道有什么用及怎么用，现在也都找到了用武之地。