2014-4-12

基于摄像头控制的飞机大战

汇编大作业说明文档

2011013239 文庆福 13681332621 thssvince@gmail.com
2011013248 王 需 18810456160 xu-wang11@mails.tsinghua.edu.cn
2011013256 杨 磊 18810305382 yl93528@gmail.com
清华大学 软件学院

目录

1. 开发环境与工具

开发环境： **Windows8.1** & **Windows8.0** & **Windows7**

运行环境： **opencv 2.48 及以上**

编辑器： **MasmPlus** & **Visual Studio 2012**

代码管理工具： **Git**

依赖库： **winmm.lib** & **gdi32.lib** & **user32.lib** & **kernel32.lib** & **gdiplus.lib**

opencv_core248d

2. 设计与实现

2.1 代码结构

Name	Date modified	Type
.git	4/10/2014 8:04 AM	File folder
images	4/12/2014 10:54 AM	File folder
sound	4/12/2014 10:42 AM	File folder
data.inc	4/10/2014 8:04 AM	INC File
gdiplus.asm	4/6/2014 9:47 AM	ASM File
GUI.APS	4/10/2014 8:04 AM	APS File
GUI.ASM	4/12/2014 10:41 AM	ASM File
GUI.exe	4/12/2014 10:56 AM	Application
GUI.ID	4/12/2014 10:56 AM	ID File
GUI.rc	4/10/2014 8:04 AM	Resource Script
GUI.res	4/12/2014 10:56 AM	Compiled Resourc...
PlaneStriker.app	4/10/2014 8:04 AM	APP File
README.md	3/26/2014 2:56 PM	MD File
struct.inc	4/10/2014 8:04 AM	INC File

程序所有的贴图文件在 **images** 文件夹，程序所有的声音文件在 **sound** 文件夹。

data.inc 定义了所需图片的位置信息

struct.inc 定义了自定义的数据结构

gdiplus.asm 封装了 gdi+加载 png 格式图片的函数。

gui.asm 实现的主体部分。主要包括绘图和碰撞检测部分。

GUI.rc 程序的资源文件

control.dll C++编译来的动态链接库，通过人脸识别对飞机进行控制。

haarcascade_frontalface_alt.xml opencv 人脸识别的训练集

2.2 摄像头控制

2.2.1 摄像头控制飞机运动流程

- 打开摄像头摄像
- 获取一帧图像
- 识别图像中的人脸
- 控制按键

以上是用摄像头控制飞机运动主要流程，下面具体说明下各个环节。

2.2.2 打开摄像头

在 win32 汇编环境下，我们采用调用 VFW 库来打开摄像头获取图像。

其中用到的主要函数有：

- capCreateCaptureWindow
- SendMessage(WM_CAP_DRIVER_CONNECT)
- SendMessage(WM_CAP_SET_PREVIEWRATE)
- SendMessage(WM_CAP_SET_PREVIEW, TRUE)

2.2.3 获取一帧图像

VFW 中若要实时获取并处理图像，需要用到 WM_CAP_SET_CALLBACK_FRAME 消息调用回调函数，在回调函数 FrameCallbackProc 中获取图像数据的指针 lpVHdr。

2.2.4 识别图像中的人脸

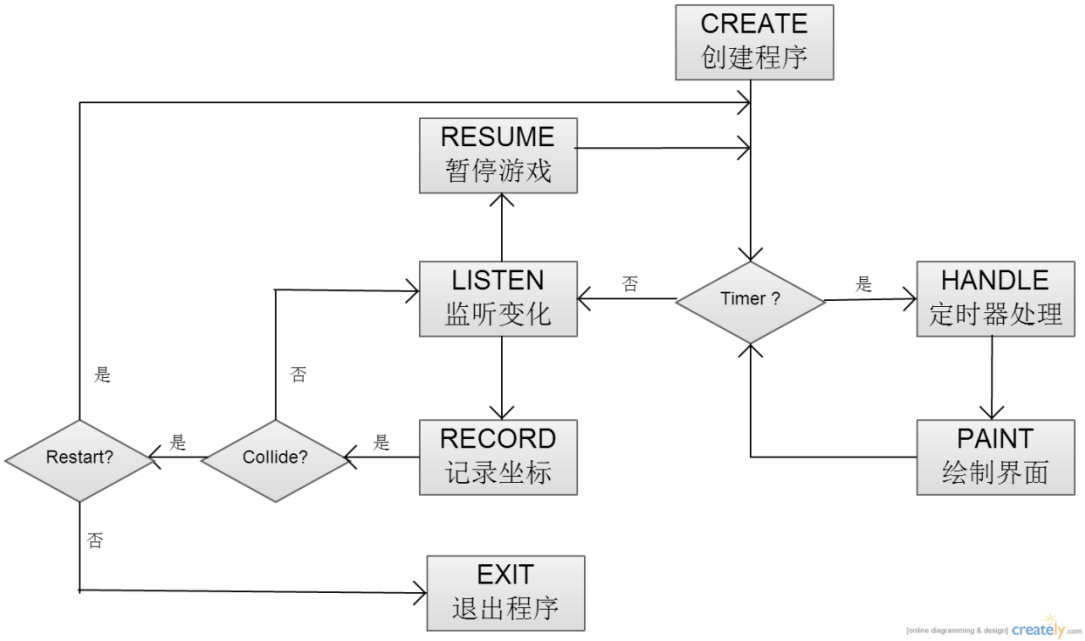
从这一部分开始我们调用 OpenCV 来实现，包括人脸识别，触发相应按键控制。我们并没有在汇编中直接调用 OpenCV 中的函数，而是先用 C++ 写好一个主要的识别与控制函数 dect_and_draw 函数，并将其编译成 dll 和 lib 文件，然后再汇编中调用该动态链接库，正是我们项目中的 control.dll 与 control.lib。

由于此次作业的重点不在人脸识别算法，人脸识别的算法在此就不赘述了。

2.2.5 控制按键

通过判断相邻两帧图像中人脸的相对位移，可以得知人脸移动的方向，从而触发对应按键来实现控制。

2.3 消息处理



2.4 界面绘制

2.4.1 Gdi+绘图

界面绘制采用了 win32 api 中的 gdi+来进行绘图。之所以不使用常用的 gdi,是因为 gdi+在可以加载透明的 png 图片。在创建程序时将各个图片资源加载进来。玩家、子弹、爆炸效果以及各种敌机都以数组的形式存储，在绘制时依次在相应的位置绘制各个形状。

2.4.2 双缓冲减少屏幕闪烁

双缓冲为了避免屏幕闪烁，使用双缓冲技术。改变以往直接重绘各个元素的做法，先将下一帧要绘制的图像先绘制好存在 buffer 中，在替换下一帧图像时，直接将存在 buffer 中的整幅图像替换进来。

2.4.3 背景滚动

背景滚动使用一张较长的图，只要保证该图片的尾部和头部完全相同，在图片滚动到尾部时在次从头部开始显示，就可以形成无限滚动的效果。

2.4.4 键盘控制

在 keydown 时赋予飞机移动速度，在 keyup 时将飞机速度置为 0，保证了飞机移动的流程，避免了仅在 keydown 时移动飞机造成的卡顿。

2.4.5 随机生成敌机

以时间为种子生成敌机，具体做法是获取当前计数的时间，乘以一个素数（程序中取 61），然后交换 al 和 ah，再将此数除以屏幕的宽度，得到的余数为敌机的坐标。

2.5 碰撞检测

每张图片都是从一个起始点开始分别绘制长宽，在检测两个物体是否碰撞时，实质是检测两个矩形是否相交。碰撞检测主要分为子弹和敌机的碰撞检测以及玩家和敌机的碰撞检测。

对每辆敌机，根据其类型不同，判断其在该帧画面中的位置周围是否有子弹，子弹的纵坐标需大于敌机，即子弹是从前面飞来的。如果检测到子弹，子弹消失，并判断敌机被子弹击中的数目是否已满，如果已满则绘制敌机爆炸。

对于玩家控制的飞机，在每帧画面中检测其周围是否有敌机，若存在敌机则玩家飞机爆炸，游戏结束。

3. 技术难点

3.1 图像数据传递

之前说过，我们在做人脸识别过程调用了基于 OpenCV 的动态链接库 control.dll。那么显然我们需要将图像数据传递到 control.dll 中的 dect_and_draw 函数中。

通过 google 我们很容易知道，回调函数 FrameCallbackProc 中的 lpVHdr 变量里存储了我们需要的图像数据。lpVHdr 是一个结构体指针 LPVIDEOHDR：

```
typedef struct videohdr_tag {
    LPBYTE        lpData;                /* pointer to locked data buffer */
    DWORD         dwBufferLength;        /* Length of data buffer */
    DWORD         dwBytesUsed;           /* Bytes actually used */
    DWORD         dwTimeCaptured;        /* Milliseconds from start of stream */
    DWORD_PTR     dwUser;                /* for client's use */
    DWORD         dwFlags;               /* assorted flags (see defines) */
    DWORD_PTR     dwReserved[4];         /* reserved for driver */
    } VIDEOHDR, NEAR *PVIDEOHDR, FAR * LPVIDEOHDR;
```

而图像数据就存储在 lpData 域，这又是一个 unsigned char * 类型。我们的做法是直接将 lpVHdr 作为参数传入 dect_and_draw 函数。

一开始我们并不理解 lpVHdr->lpData 域里面图像数据的格式，我们直接输出发现就是一堆乱码，甚至怀疑这个参数错了，并不是图像数据。后面尝试转为 int 输出发现都是一堆介于 0 到 255 之间的数字，我们确定这个就是我们要的图像数据。

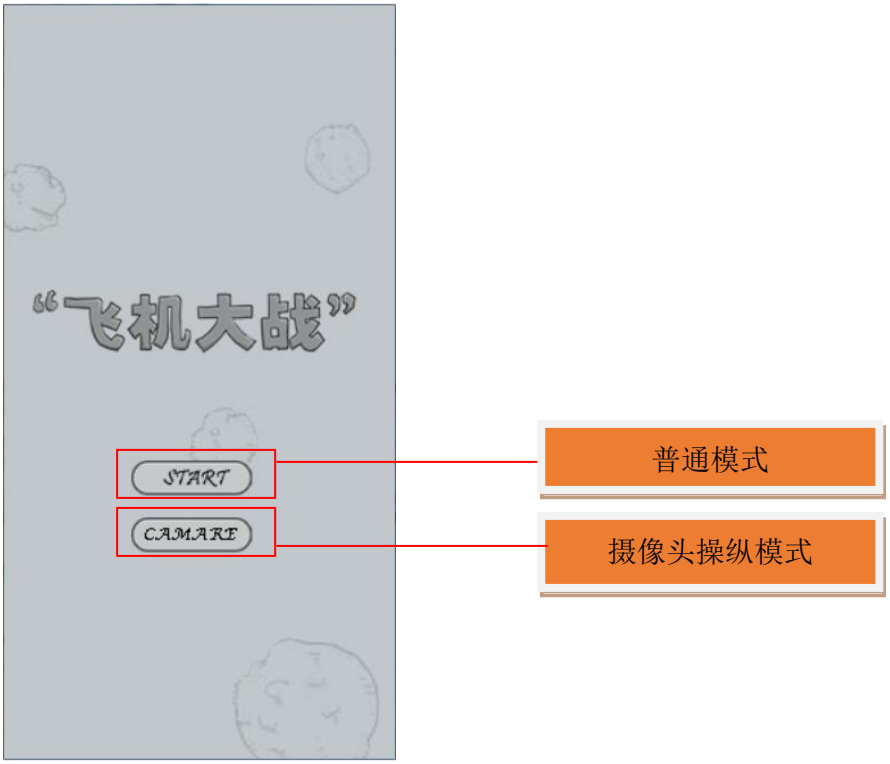
然后我们就直接去寻找如何将 lpVHdr->lpData 这个 unsigned char * 类型的数据转化为 OpenCV 所需要的 IplImage 图像数据类型。费劲周折，又开始怀疑是拿到的数据不是我们要的图像。

最后找了大量资料，无意中发现图像数据原来是有格式问题的。这里的

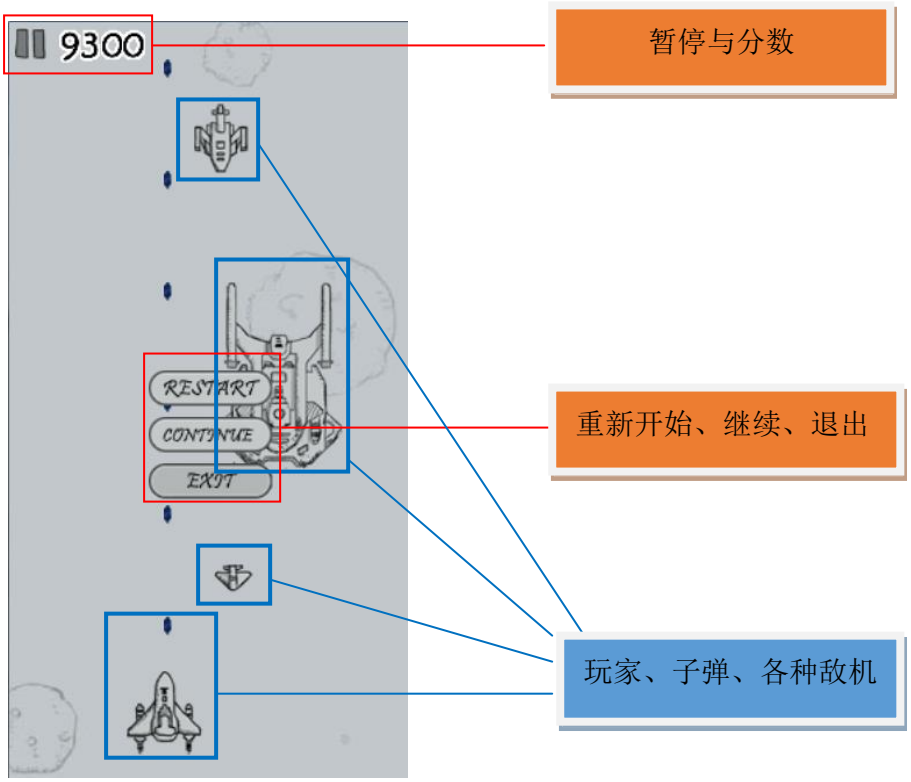
lpVHdr->lpData 图像是 VFW 直接获取的是 YUY2 格式的图像数据，而主流所需要的是 RGB 数据，所以我们先得将其转化为 RGB 格式的图像，然后再将其转化为 IplImage 类型。代码中 YUY2_RGB 和 Byte2IplImg 就是做这两件事情的。得到 IplImage 类型的数据，那么后面的识别就好做了。

4. 功能描述

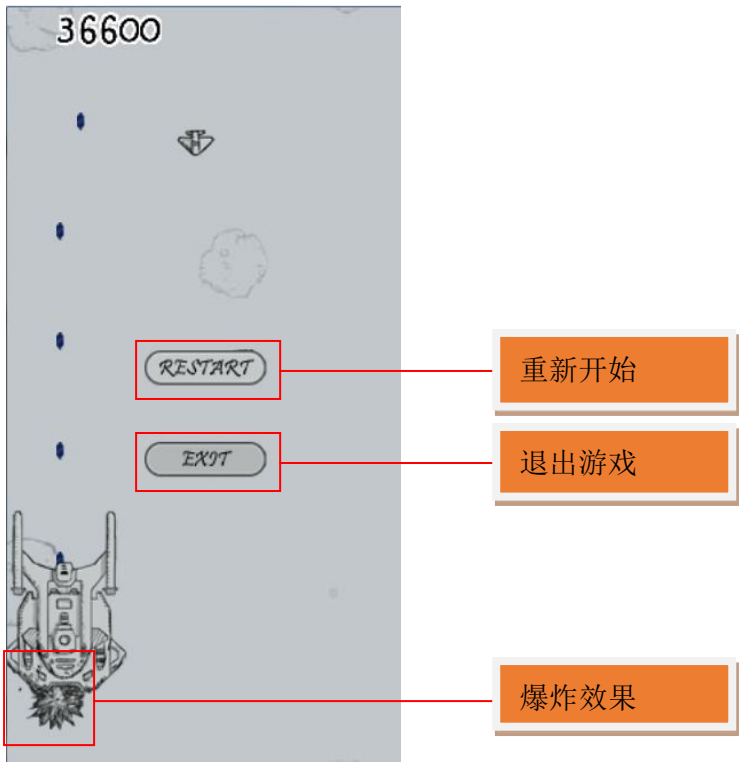
开始游戏



暂停游戏


9300
RESTART
CONTINUE
EXIT
暂停与分数
重新开始、继续、退出
玩家、子弹、各种敌机

结束游戏


36600
RESTART
EXIT
重新开始
退出游戏
爆炸效果

5. 致谢

非常感谢王朝坤老师孜孜不倦地传授我们 windows 汇编程序设计的知识，感谢陈俊助教呕心沥血辛苦批阅我们的作业！

感谢这一路相伴一同奋斗的战友们！

6. 参考资料

1. http://devpinoy.org/blogs/cvega/archive/2008/10/26/load-png-jpeg-from-resource-using-gdi-masm.aspx
2. 《Intel 汇编程序设计》
3. http://msdn.microsoft.com/en-us/library/windows/desktop/dd757161(v=vs.85).aspx
4. http://www.aogosoft.com/downpage.asp?mode=viewtext&id=197

7. 附录

7.1 BUG 跟踪与解决

BUG	解决方法
Win7 下在 LOOP 循环中调用 mciSendString 播放音乐界面卡死	Win7 下调用 mciSendString 播放音乐时，会修改 ECX，如果直接用 ECX 进行循环则会造成卡死（在 Win8 下运行正常）。解决方法是用一个内存变量来进行循环。
敌机的爆炸效果不定期绘制不出	设置的重绘定时器为 10ms，而动画制作中 1s 播放 24 帧，人眼大概能看到 30ms 一帧的画面，由于刷新过快造成一些爆炸效果无法被人眼看到。解决方法是每个爆炸效果绘制多帧。
Camera 模式下玩家飞机爆炸时程序崩溃	在调试中发现会造成内存越界，在实际中发现在玩家和敌机碰撞时音乐被播放了多次，将播放爆炸音强制只播放一次。该问题暂时得到解决，可能留有隐患。

7.2 文档编写修订日志

版本	时间	参与人员	主要工作
Version 1.0	2014 年 4 月 12 日	文庆福、王需、杨磊	完成开发环境与工具、代码设计与实现、技术难点和功能描述的编写