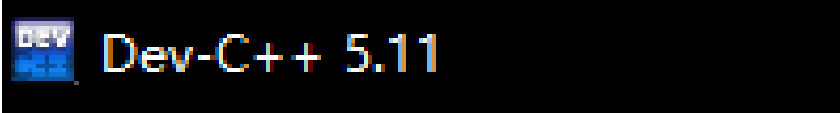

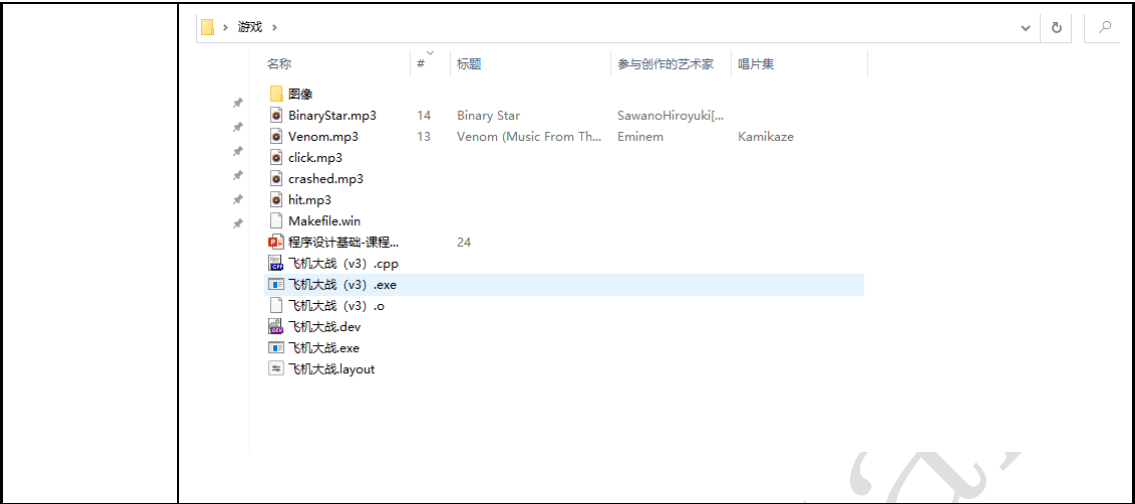


项目名称	飞机大战 (Version 3)																				
项目简介	<p>该项目“小飞机游戏”采用了 EGE 图像库，利用了 C 的整型和字符型数据类型和控制流的分支和循环，同时使用了标准库的 <code><math.h></code>, <code><stdio.h></code>, <code><stdlib.h></code> 函数，实现了以下功能：</p> <ol style="list-style-type: none"> 1. 飞机上下移动 2. 飞机左右移动 3. 飞机斜方向移动 4. 屏幕刷新 5. 输出不同颜色字体 6. 得分判定 7. 飞机间、子弹与飞机之间碰撞检测 8. 飞机暂时升级 9. 失败判定 10. 退回主界面 11. 暂停游戏 12. 重新游戏 13. 扣分判定 14. 生命值判定 																				
实验环境	<ul style="list-style-type: none"> ● 硬件环境： <table> <tr> <td>设备名称</td><td>DESKTOP-BE9JN0S</td></tr> <tr> <td>处理器</td><td>Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz</td></tr> <tr> <td>机带 RAM</td><td>8.00 GB (7.84 GB 可用)</td></tr> <tr> <td>设备 ID</td><td>67E84990-195D-4D41-9798-5EFB528818C2</td></tr> <tr> <td>产品 ID</td><td>00342-35458-46555-AAOEM</td></tr> <tr> <td>系统类型</td><td>64 位操作系统, 基于 x64 的处理器</td></tr> <tr> <td>笔和触控</td><td>没有可用于此显示器的笔或触控输入</td></tr> </table> ● 操作系统： <table> <tr> <td>版本</td><td>Windows 10 家庭中文版</td></tr> <tr> <td>版本号</td><td>20H2</td></tr> <tr> <td>安装日期</td><td>2021/1/3</td></tr> </table> ● 开发环境：Dev-C++ 5.11 	设备名称	DESKTOP-BE9JN0S	处理器	Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz	机带 RAM	8.00 GB (7.84 GB 可用)	设备 ID	67E84990-195D-4D41-9798-5EFB528818C2	产品 ID	00342-35458-46555-AAOEM	系统类型	64 位操作系统, 基于 x64 的处理器	笔和触控	没有可用于此显示器的笔或触控输入	版本	Windows 10 家庭中文版	版本号	20H2	安装日期	2021/1/3
设备名称	DESKTOP-BE9JN0S																				
处理器	Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz																				
机带 RAM	8.00 GB (7.84 GB 可用)																				
设备 ID	67E84990-195D-4D41-9798-5EFB528818C2																				
产品 ID	00342-35458-46555-AAOEM																				
系统类型	64 位操作系统, 基于 x64 的处理器																				
笔和触控	没有可用于此显示器的笔或触控输入																				
版本	Windows 10 家庭中文版																				
版本号	20H2																				
安装日期	2021/1/3																				

	
实验过程	<p>【步骤描述】</p> <ol style="list-style-type: none">1. 安装 DEV IDE2. 安装 EGE 图形库 <p>参见教程： https://blog.csdn.net/qq_39151563/article/details/100161986</p> <ol style="list-style-type: none">3. 调用 EGE 库里的函数，运行确保安装成功  <ol style="list-style-type: none">4. 录入代码



```

planetextimg = newimage();
gameback = newimage();
gameback1 = newimage();
gameback2 = newimage();
wd_back = newimage();
imgofplane = newimage();
bullet1 = newimage();
bullet3 = newimage();
bullet_enm = newimage();
enemy1 = newimage();
Gameover = newimage();
img_gamestart = newimage();
img_gameover = newimage();
img_update = newimage();

// 循环来获得所有爆炸的图片
char fileofboom[100]; // 用于保存文件名
for (int i = 0; i < 9; i++) {
    boom[i] = newimage();
    sprintf(fileofboom, "图像/爆炸%d.png", i);
    getimage(boom[i], fileofboom);
    setimage(100, 100, boom[i]);
}

// 循环来获得所有加载的图片
char fileofload[100];
for (int i = 0; i < 11; i++) {
    load[i] = newimage();
    sprintf(fileofload, "图像/加载%d.jpg", i);
    getimage(load[i], fileofload);
    setimage(wide, high, load[i]);
}

// 循环来获得所有坠毁的图片
char fileofboom_self[100]; // 用于保存文件名
for (int i = 1; i < 10; i++) {
    boom_self[i] = newimage();
    sprintf(fileofboom_self, "图像/坠毁%d.png", i);
    getimage(boom_self[i], fileofboom_self);
    setimage(100, 100, boom_self[i]);
}

getimage(beginimg, "图像/开始界面.jpg"); // 获取开始界面图片
setimage(wide, high, beginimg);

getimage(planetextimg, "图像/飞机大战.png"); // 获取开始游戏文字图片
setimage(450, 150, planetextimg);

getimage(imgofplane, "图像/小飞机1.0.png"); // 获取飞机图片
setimage(sizeofplane, sizeofplane, imgofplane);

getimage(gameback, "图像/背景.jpg"); // 获取游戏背景图
setimage(wide-200, 2*high, gameback);

getimage(gameback1, gameback, 0, 0, wide-200, high); // 将得到的游戏背景图一分为二
getimage(gameback2, gameback, 0, high, wide-200, high); // 将得到的游戏背景图一分为二

getimage(wd_back, "图像/wd_bk.png"); // 获取文字背景图
setimage(200, high+200, wd_back);

getimage(bullet1, "图像/子弹1.png"); // 获取子弹1图片
setimage(50, 50, bullet1);

getimage(bullet3, "图像/子弹3.png"); // 获取子弹3图片
setimage(50, 50, bullet3);

getimage(bullet_enm, "图像/敌机子弹0.png"); // 获取敌机子弹图片
setimage(15, 15, bullet_enm);

getimage(enemy1, "图像/敌机1.png"); // 获取敌机1图片
setimage(100, 100, enemy1);

getimage(Gameover, "图像/gameover.png"); // 获取游戏结束界面
setimage(wide, high, Gameover);

getimage(img_gamestart, "图像/开始游戏.png"); // 开始游戏按钮
setimage(200, 200, img_gamestart);

getimage(img_gameover, "图像/退出游戏.png"); // 结束游戏按钮
setimage(200, 200, img_gameover);

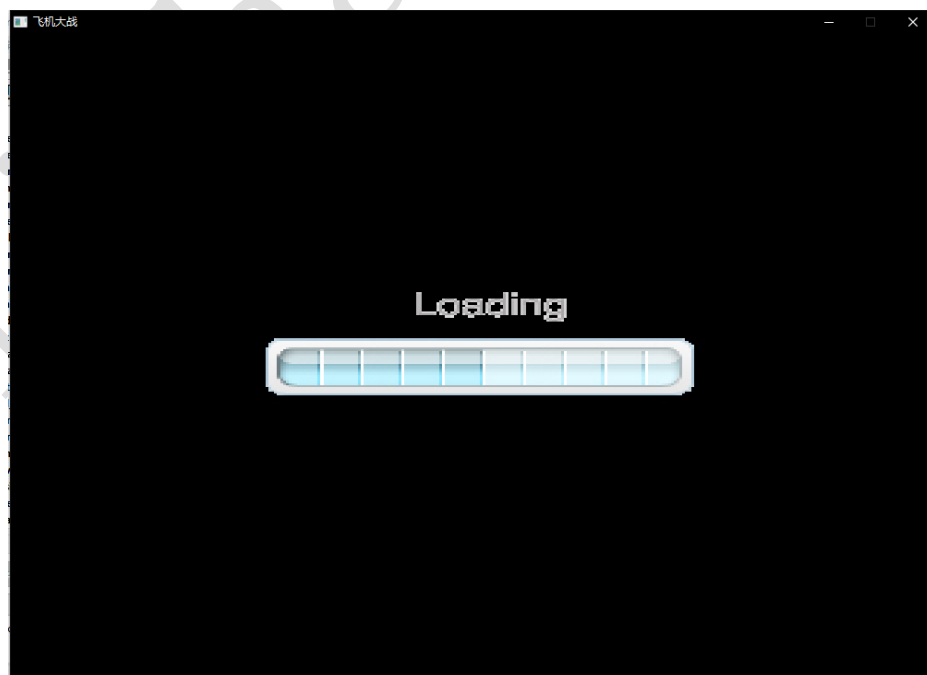
getimage(img_update, "图像/升级.png"); // 升级包图片
setimage(50, 40, img_update);

music_get(); // 音乐
}

```

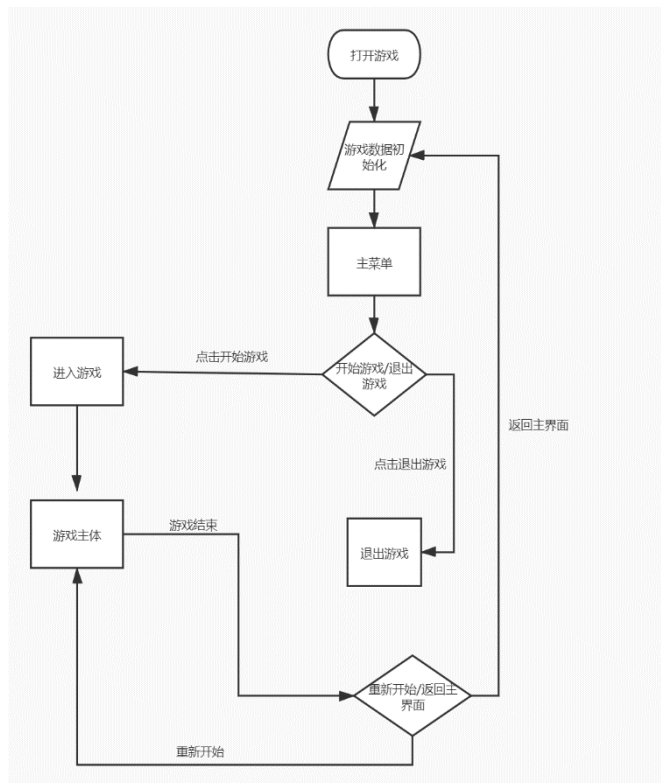
6. 编译

7. 运行：运行结果见下面的截图



代码分析及
实验结果

【流程图】



关键代码:

【显示】

所有关键图像及文字均由 show 函数来执行, 在 show 函数外对各个图像的 x, y 坐标进行改变, 在 show 函数中通过它们的坐标来进行显示。

```

//显示画面
void show() {

    cleardevice(); //清屏
    //抓图并在程序中放置
    putimage(200, yofback1, gameback1); //显示游戏背景1
    putimage(200, yofback2, gameback2); //显示游戏背景2
    putimage_withalpha(NULL, wd_back, 0, -50); //显示文字背景
    putimage_withalpha(NULL, imgofplane, ximgofplane, yimgofplane); //显示飞机

    //显示子弹
    for ( int i = ammo; i>=0; i-- ) {
        if( bullet[i].s==1 ) {
            if( bullet[i].up ) putimage_withalpha(NULL, bullet3, bullet[i].x, bullet[i].y);
            else putimage_withalpha(NULL, bullet1, bullet[i].x, bullet[i].y);
        }
    }

    //显示敌机
    for ( int i = 0; i<numofenm_now; i++ ) {
        //这个位置有敌机且未被消灭,显示敌机
        if( enemy[i].s==1 && enemy[i].iskilled == 0 ) {
            putimage_withalpha(NULL, enemy1, enemy[i].x, enemy[i].y);
            //显示在画布中的敌机的子弹
            for ( int j = 0; j<ammo_enm; j++ ) {
                if ( enemy[i].blt[j].s_blt == 1 ) putimage_withalpha(NULL, bullet_enm, enemy[i].blt[j].x, enemy[i].blt[j].y);
            }
            //这个位置有敌机但已被消灭,显示爆炸动画
            if( enemy[i].s==1 && enemy[i].iskilled == 1 ) {
                boom_gif( enemy[i].x, enemy[i].y, i );
            }
        }
    }

    //显示升级包
    if ( isupexist ) putimage_withalpha(NULL, img_update, x_up, y_up);

    //文字相关的显示
    setbkmode(TRANSPARENT); //设置文字背景色为透明
    setcolor(BLACK);
    setfont(30, 0, "黑体");
    xyprintf(0, high*0.95, "得分: %d", score);
    setfont(30, 0, "黑体");
    setcolor(RED);
    outtextxy(5, 0, "控制 :");
    outtextxy(5, 35, "W : 上");
    outtextxy(5, 70, "S : 下");
    outtextxy(5, 105, "A : 左");
    outtextxy(5, 140, "D : 右");
    outtextxy(5, 175, "P: 暂停游戏");
    outtextxy(5, 210, 200, 70, "M: 打开/关闭音乐");
    setfont(35, 0, "楷体");
    outtextxy(5, 290, "血量");

    //血量显示
    setfillcolor(YELLOW);
    bar(10, 340, 180, 360);
    setfillcolor(BLACK);
    bar(blood_x, 340, 180, 360);
}

```

【GIF】


```

//加载中
void load_gif() {

    double t3 = fclock();
    int i = 0;
    while(1) {
        if ( (fclock()-t3) >= 0.15 ) {
            cleardevice();
            putimage(0,0,load[i]);
            i++;
            t3 = fclock();
            delay_fps(60);
        }
        if ( i > 10 ) break;
    }
}

//显示爆炸的动图
void boom_gif( int x, int y, int i) {

    static double t1 = fclock();
    if ( (fclock()-t1)>0.03 ) {
        putimage_withalpha(NULL, boom[enemy[i].pboom], x, y);
        enemy[i].pboom++;
        t1 = fclock();
    }
    //爆炸动画结束后, 初始化
    if( enemy[i].pboom > 8 ) {
        enemy[i].s = 0;
        enemy[i].iskilled = 0;
        enemy[i].blood = 3;
        enemy[i].pboom = 0;
    }
}

```

这里借用 main 函数里的循环来进行刷新与显示

显

示

结

果

:



点击播放 gif

【飞机上下左右移动】：

```

//关于飞机位置及移动的量
float ximgofplane = wide*0.5 - sizeofplane/2 + 100, yimgofplane = high - sizeofplane; //浮点型, 精确小飞机位置, 这里初始化为小飞机出生点
float speed = 10.0f; //移动速度, 浮点型能任意控制速度快慢
float xspeeds[4] = { -speed, 0, speed, 0 };
float yspeeds[4] = { 0, -speed, 0, speed };
int keys[4] = {'A', 'W', 'D', 'S' }; //0, 1, 2, 3分别代表左, 上, 右, 下
int directkeys[4] = {key_left, key_up, key_right, key_down };

```

```
// 飞机移动WASD/上下左右
```

```
void moving () {  
    float xNext = ximgofplane; // 获得小飞机的初始位置  
    float yNext = yimgofplane;  
    for(int i = 0; i < 4; i++) {  
        if( keystate(keys[i]) || keystate(directkeys[i]) ) {  
            xNext += xspeeds[i];  
            yNext += yspeeds[i];  
        }  
    }  
    // 如果移动了  
    if (xNext != ximgofplane || yNext != yimgofplane) {  
        if ( 200 <= xNext && xNext <= wide-sizeofplane && 0 <= yNext && yNext <= high-sizeofplane ) {  
            ximgofplane = xNext;  
            yimgofplane = yNext;  
        }  
        // 改变飞机坐标  
    }  
}
```

将 A, W, D, S 的按键状态存入两个数组中，按下时即进行飞机坐标的改变，从而达到移动的效果

操作说明：可以利用键盘的“a”、“d”、“w”和“s”键来控制飞机的上下左右移动：

- “A” 键：左移
- “D” 键：右移
- “W” 键：上移
- “S” 键：下移
- M： 打开/关闭音乐

【射击】

```

void bulletget() {
    // 获取当前时间
    static double t2 = fclock();

    // 获取当前子弹的初始位置并存储在结构数组中
    for (int i = 0; i < ammo; i++) {
        // i%btbullet 用于控制子弹生成的速度
        if ( bullet[i].s == 0 && (fclock()-t2) >= gapofbullet ) {
            // 每一枚子弹的初始位置总是飞机头部
            bullet[i].x = xingofplane+10;
            bullet[i].y = yingofplane-50;
            bullet[i].s = 1; // 该数组位置存储了一枚子弹
            if (isupdated) bullet[i]._up = 1;
            else bullet[i]._up = 0;
            t2 = fclock();
            break;
        }
    }

    // 每一枚子弹的位置不停改变
    for (int j = 0; j < ammo; j++) {
        if ( bullet[j].s == 1 ) {
            if( bullet[j].y > 0 ) {
                bullet[j].y -= spdofblt; // 子弹移动
            } else {
                bullet[j].y = high + 50; // 子弹到达极限位置后移出屏幕
                bullet[j].s = 0; // 该数组位置重置
            }
        }
    }
}

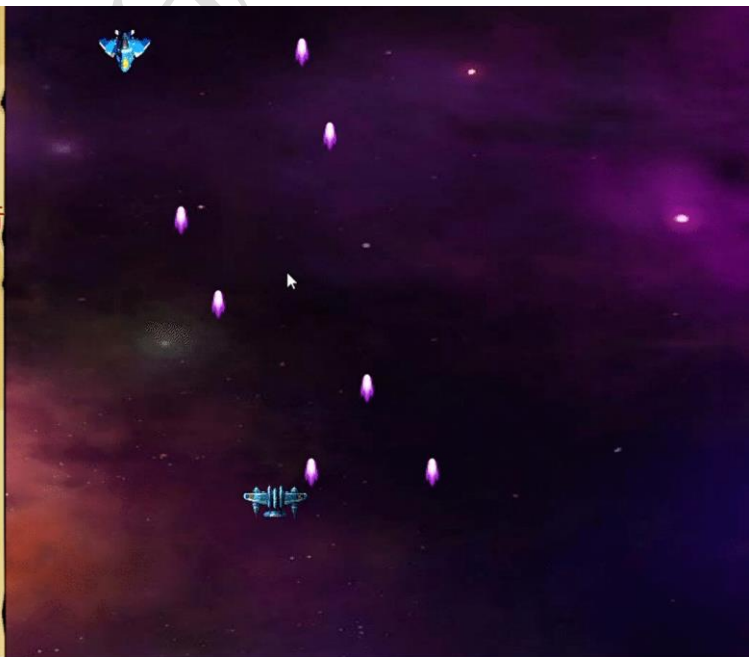
```

显示结果:

控制：
W：上
S：下
A：左
D：右
P：暂停游戏
M：打开/关闭音乐

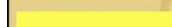
血量

得分： 0



控制：
W：上
S：下
A：左
D：右
P：暂停游戏
M：打开/关闭音乐

血量



得分： 0



- P：暂停游戏如图所示



- M: 关闭/打开音乐
- P: 返回游戏

【受伤扣血，零血死亡】

```
//检测是否被敌机子弹攻击
void isattacked() {
    //对存在的敌机进行检测
    for (int i = 0; i < numofenm_now; i++) {
        if (enemy[i].s == 1 && enemy[i].iskilled == 0) {
            for (int j = 0; j < ammo_enm; j++) {
                if (enemy[i].blt[j].s_blt == 1) {
                    if ((abs(xingofplane-enemy[i].blt[j].blt_enm_x) + abs(yingofplane-enemy[i].blt[j].blt_enm_y)) <= 40) {
                        enemy[i].blt[j].s_blt = 0;
                        if (blood_x > 10) blood_x -= 10;
                        if (blood_x <= 10) isExploded = 1;
                        break;
                    }
                }
            }
        }
    }
}
```

显示结果:

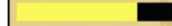
控制：
W：上
S：下
A：左
D：右
P：暂停游戏
M：打开/关闭音乐
血量

得分： 0



控制：
W：上
S：下
A：左
D：右
P：暂停游戏
M：打开/关闭音乐

血量



得分： 0



控制：
W：上
S：下
A：左
D：右
P：暂停游戏
M：打开/关闭音乐



血量



得分： 10



【与敌机相撞】

	<div data-bbox="329 196 515 852"><p>控制： W：上 S：下 A：左 D：右 P：暂停游戏 M：打开/关闭音乐</p><p>血量</p><div data-bbox="333 513 496 533" style="background-color: yellow; width: 117px; height: 10px;"></div></div> <div data-bbox="329 813 454 852"><p>得分： 10</p></div> <div data-bbox="515 196 1262 852"></div> <div data-bbox="329 862 515 1519"><p>控制： W：上 S：下 A：左 D：右 P：暂停游戏 M：打开/关闭音乐</p><p>血量</p><div data-bbox="333 1179 496 1199" style="background-color: black; width: 117px; height: 10px;"></div></div> <div data-bbox="329 1479 454 1519"><p>得分： 10</p></div> <div data-bbox="515 862 1262 1519"></div>
<p>实验代码</p>	<p>见附的源文件“飞机大战（v3）.cpp”</p>

参考资料和 相关网站	<p>1. EGE 教程专栏:</p> <p>https://blog.csdn.net/qq_39151563/category_9311717.html</p> <p>2. EGE 帮助文档:</p> <p>https://xege.org/manual/index.htm</p> <p>3. 游戏素材: 爱给网</p> <p>https://www.aigei.com/</p>
---------------	--