第5章

应用图片与声音素材的游戏开发

本章学习图片、音乐等多媒体素材的导入与使用，进一步提升游戏效果。

5.1 使用图片与声音

本节学习利用 getimage、putimage、mciSendString 等函数在游戏中使用图片与声音，如图 5-1 所示。实现的 flappy bird 游戏原型代码参看“\随书资源\第 5 章\5.1 使用图片与声音\5.1 flappy bird 原型.cpp”，游戏视频“5.1 flappy bird 视频.wmv”。

5.1.1 图片的导入与使用

以下代码利用 loadimage 函数导入一张图片，并将对应的图片对象 img_bk 利用 putimage 函数输出到屏幕，如图 5-2 所示。对于更多图像处理相关函数的使用方法可以查阅安装目录下的 EasyX 帮助文档(EasyX_Help.chm\函数说明\图像处理相关函数)。

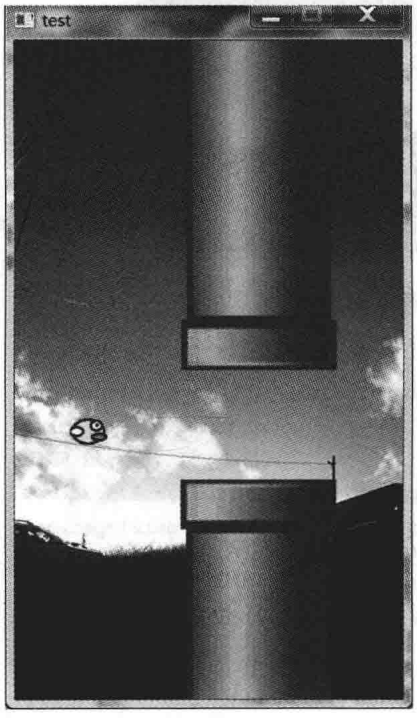

图 5-1 flappy bird 游戏效果

图 5-2 显示背景图片

```
#include <graphics.h>
#include <conio.h>
int main()
{
    initgraph(350, 600);
    IMAGE img_bk;                                    // 定义 IMAGE 对象
    loadimage(&img_bk, "D:\\background.jpg");        // 读取图片到 IMAGE 对象中
    putimage(0, 0, &img_bk);                         // 在坐标 (0, 0) 位置显示 IMAGE 对象
    getch();
    closegraph();
    return 0;
}
```

background.jpg 在“\随书资源\第 5 章\5.1 使用图片与声音\flappy bird 图片音乐素材\”中，可以先将 background.jpg 复制到 D 盘根目录。根据字符串的知识，loadimage()函数中的文件路径需要写成“D:\\background.jpg”。对于某些编译器(例如 Visual Studio 2015)，文件路径字符串需要写成“_T("D:\\background.jpg")”。

另外，可以在游戏背景上绘制小鸟图像 bird2.jpg，如图 5-3 所示。

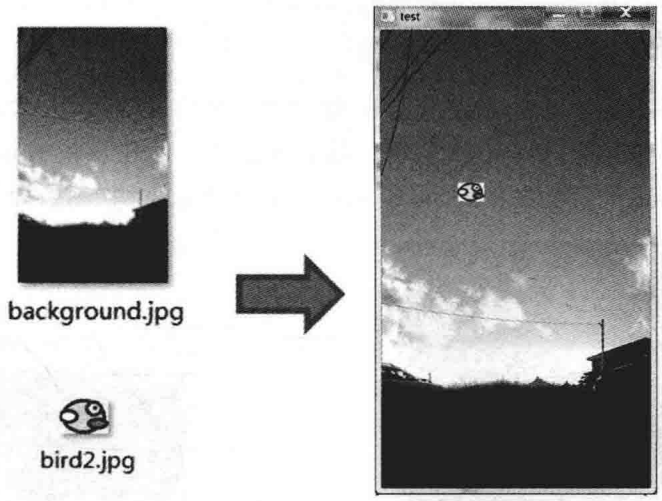

图 5-3　显示小鸟与背景图片

```
#include <graphics.h>
#include <conio.h>
int main()
{
    initgraph(350, 600);
    IMAGE img_bk;                                    // 定义 IMAGE 对象
    loadimage(&img_bk, "D:\\background.jpg");        // 读取图片到 IMAGE 对象中
    putimage(0, 0, &img_bk);                         // 在坐标 (0, 0) 位置显示 IMAGE 对象
    IMAGE img_bd;
    loadimage(&img_bd, "D:\\bird2.jpg");
    putimage(100, 200, &img_bd);
    getch();
    closegraph();
```

```
    return 0;
}
```

5.1.2　遮罩图的使用

以上实现的程序在小鸟的周边会出现明显的白色边框，可以进一步利用小鸟的遮罩图片 bird1.jpg。bird1.jpg 与 bird2.jpg 中的像素一一对应，bird1 中白色的区域将 bird2 中对应的像素显示；bird1 中黑色的区域将 bird2 中对应的像素隐藏。效果如图 5-4 所示。

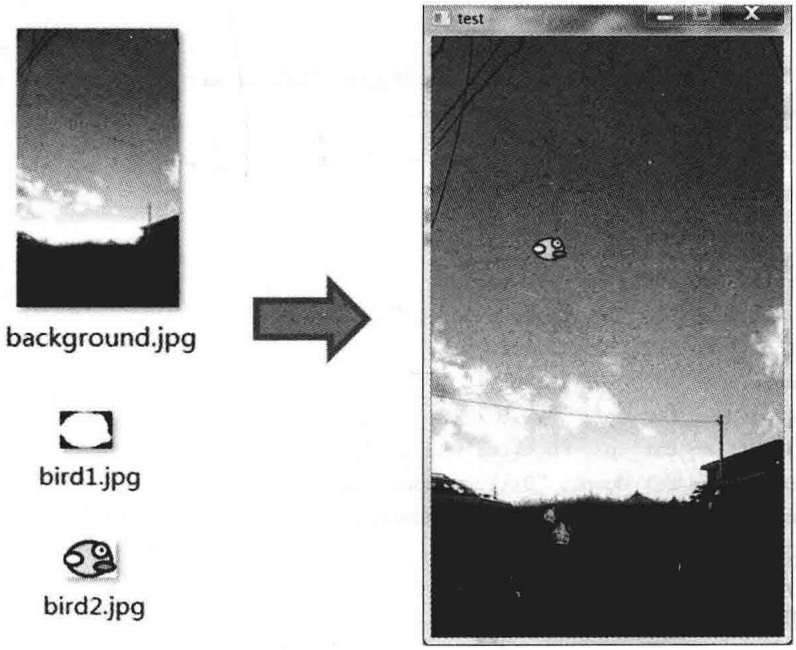

图 5-4　遮罩图的应用效果

```
#include <graphics.h>
#include <conio.h>
int main()
{
    initgraph(350, 600);
    IMAGE img_bk;                                    // 定义 IMAGE 对象
    loadimage(&img_bk, "D:\\background.jpg");        // 读取图片到 IMAGE 对象中
    putimage(0, 0, &img_bk);                         // 在坐标 (0, 0) 位置显示 IMAGE 对象
    IMAGE img_bd1,img_bd2;
    loadimage(&img_bd1, "D:\\bird1.jpg");
    loadimage(&img_bd2, "D:\\bird2.jpg");
    putimage(100, 200, &img_bd1,NOTSRCERASE);
    putimage(100, 200, &img_bd2,SRCINVERT);
    getch();
    closegraph();
    return 0;
}
```

对应的遮罩图片可以用 Photoshop 等软件抠图生成。如果有带透明通道的 png 图片，

本书还提供了自动生成遮罩图片的程序和源代码，参看“\随书资源\第5章\5.1使用图片与声音\png2bmp&mask\”。

5.1.3 flappy bird初步

本节利用游戏框架和导入的图片实现flappy bird的游戏原型，小鸟自由下落，按空格键后向上移动，如图5-1所示。

```
#include <graphics.h>
#include <conio.h>

IMAGE img_bk,img_bd1,img_bd2,img_bar_up1,img_bar_up2,img_bar_down1,img_bar_down2;
int bird_x;
int bird_y;

void startup()
{
    initgraph(350, 600);
    loadimage(&img_bk, "D:\\background.jpg");
    loadimage(&img_bd1, "D:\\bird1.jpg");
    loadimage(&img_bd2, "D:\\bird2.jpg");
    loadimage(&img_bar_up1, "D:\\bar_up1.gif");
    loadimage(&img_bar_up2, "D:\\bar_up2.gif");
    loadimage(&img_bar_down1, "D:\\bar_down1.gif");
    loadimage(&img_bar_down2, "D:\\bar_down2.gif");
    bird_x = 50;
    bird_y = 200;
    BeginBatchDraw();
}

void show()
{
    putimage(0, 0, &img_bk);                                    // 显示背景
    putimage(150, -300, &img_bar_up1,NOTSRCERASE);              // 显示上面一半的障碍物
    putimage(150, -300, &img_bar_up2,SRCINVERT);
    putimage(150, 400, &img_bar_down1,NOTSRCERASE);             // 显示下面一半的障碍物
    putimage(150, 400, &img_bar_down2,SRCINVERT);
    putimage(bird_x, bird_y, &img_bd1,NOTSRCERASE);             // 显示小鸟
    putimage(bird_x, bird_y, &img_bd2,SRCINVERT);
    FlushBatchDraw();
    Sleep(50);
}

void updateWithoutInput()
{
    if (bird_y<580)
        bird_y = bird_y+3;
}

void updateWithInput()
```

```
{
    char input;
    if(kbhit())                                    // 判断是否有输入
    {
        input = getch();
        if (input == ' '&& bird_y>20)
            bird_y = bird_y - 60;
    }
}

void gameover()
{
    EndBatchDraw();
    getch();
    closegraph();
}

int main()
{
    startup();                                     // 数据的初始化
    while (1)                                      // 游戏循环执行
    {
        show();                                    // 显示画面
        updateWithoutInput();                      // 与用户输入无关的更新
        updateWithInput();                         // 与用户输入有关的更新
    }
    gameover();                                    // 游戏结束,进行后续处理
    return 0;
}
```

5.1.4　声音的导入与使用

声音对于游戏也是一个非常重要的元素,可以使用 mciSendString 函数播放 mp3 文件。在程序前加上 # pragma comment(lib,"Winmm. lib"),以下语句可以循环播放背景音乐:

```
mciSendString("open D:\\background. mp3 alias bkmusic", NULL, 0, NULL);  // 背景音乐
mciSendString("play bkmusic repeat", NULL, 0, NULL);                     // 循环播放
```

以下语句可以播放一次音乐:

```
mciSendString("open D:\\Jump. mp3 alias jpmusic", NULL, 0, NULL);        // 打开音乐
mciSendString("play jpmusic", NULL, 0, NULL);                            // 仅播放一次
```

如果需要多次播放某一音乐,则需要先关闭再打开播放:

```
mciSendString("close jpmusic", NULL, 0, NULL);                  // 先把前面一次的音乐关闭
mciSendString("open D:\\Jump. mp3 alias jpmusic", NULL, 0, NULL);        // 打开音乐
mciSendString("play jpmusic", NULL, 0, NULL);                            // 仅播放一次
```

5.1.5 带音效的 flappy bird

这里导入背景音乐 background.mp3 循环播放，并且每按一次空格键播放一次音乐 Jump.mp3，需要先将 mp3 文件复制到对应的目录。

```
#include <graphics.h>
#include <conio.h>
// 引用 Windows Multimedia API
#pragma comment(lib,"Winmm.lib")
IMAGE img_bk,img_bd1,img_bd2,img_bar_up1,img_bar_up2,img_bar_down1,img_bar_down2;
int bird_x;
int bird_y;

void startup()
{
    initgraph(350, 600);
    loadimage(&img_bk, "D:\\background.jpg");
    loadimage(&img_bd1, "D:\\bird1.jpg");
    loadimage(&img_bd2, "D:\\bird2.jpg");
    loadimage(&img_bar_up1, "D:\\bar_up1.gif");
    loadimage(&img_bar_up2, "D:\\bar_up2.gif");
    loadimage(&img_bar_down1, "D:\\bar_down1.gif");
    loadimage(&img_bar_down2, "D:\\bar_down2.gif");
    bird_x = 50;
    bird_y = 200;
    BeginBatchDraw();

    mciSendString("open D:\\background.mp3 alias bkmusic", NULL, 0, NULL);  // 打开背景音乐
    mciSendString("play bkmusic repeat", NULL, 0, NULL);  // 循环播放
}

void show()
{
    putimage(0, 0, &img_bk);                              // 显示背景
    putimage(150, -300, &img_bar_up1,NOTSRCERASE);        // 显示上面一半的障碍物
    putimage(150, -300, &img_bar_up2,SRCINVERT);
    putimage(150, 400, &img_bar_down1,NOTSRCERASE);       // 显示下面一半的障碍物
    putimage(150, 400, &img_bar_down2,SRCINVERT);
    putimage(bird_x, bird_y, &img_bd1,NOTSRCERASE);       // 显示小鸟
    putimage(bird_x, bird_y, &img_bd2,SRCINVERT);
    FlushBatchDraw();
    Sleep(50);
}

void updateWithoutInput()
{
    if (bird_y<500)
        bird_y = bird_y+3;
}
```

```
void updateWithInput()
{
    char input;
    if(kbhit())                                          // 判断是否有输入
    {
        input = getch();
        if (input == ''&& bird_y>20)
        {
            bird_y = bird_y - 60;
            mciSendString("close jpmusic", NULL, 0, NULL);        // 先把前面一次的音乐关闭
            mciSendString("open D:\\Jump.mp3 alias jpmusic", NULL, 0, NULL);    // 打开音乐
            mciSendString("play jpmusic", NULL, 0, NULL); // 仅播放一次
        }
    }
}

void gameover()
{
    EndBatchDraw();
    getch();
    closegraph();
}

int main()
{
    startup();                                           // 数据的初始化
    while (1)                                            // 游戏循环执行
    {
        show();                                          // 显示画面
        updateWithoutInput();                            // 与用户输入无关的更新
        updateWithInput();                               // 与用户输入有关的更新
    }
    gameover();                                          // 游戏结束,进行后续处理
    return 0;
}
```

5.1.6　小结

带图片、音乐效果的 flappy bird 是不是很帅？

思考题：实现完整的 flappy bird 游戏。

5.2　飞机大战

本节继续应用图片、音乐素材实现一个鼠标控制的飞机大战游戏，如图 5-5 所示。游戏代码和素材参看“\随书资源\第 5 章\5.2 飞机大战\”。

5.2.1　用鼠标控制飞机移动

第一步实现鼠标控制飞机移动，如图 5-6 所示。首先需要将 background.jpg、planeNormal_1.jpg、planeNormal_2.jpg 复制到对应目录。