Saamliu

BikeGraphix

A computer graphics learning project.

Samliu

2022-12-28

目录

一、前言

（一）项目简介

1.项目名称：运动的自行车

2.项目简介：项目选题来源于教材第 3 章习题 4。编写一辆自行车在一公路上由右至左快速行驶的程序。

（二）涉及到的知识

本项目涉及到以下知识点：安装 Turbo C++ 4.0、图形系统初始化与关闭、图形颜色设置、图形模式下文本处理、图形存取处理、常用画图函数。

二、环境说明

（一）硬件环境

机型	Legion R7000
CPU	AMD Ryzen 7 4800H with Radeon Graphics 2.90 GHz
显卡	NVIDIA GeForce GTX 1650Ti
操作系统	Windows 10 64 位

表 1 硬件环境说明图

（二）软件环境

DosBox	DOSBox 是一个 DOS 模拟程序，由于它采用的是 SDL 库，所以可以很方便的移植到其他的平台。DOSBox 的最新版本已经支持在 Windows、Linux、Mac OS X、BeOS 、palmOS、Android 、webOS、os/2 等系统中运行。
Turbo C++ 4.0	Turbo C 2.0 版不仅是一个快捷、高效的编译程序，同时还有一个易学、易用的集成开发环境。使用 Turbo C 2.0 版无需独立地编辑、编译和连接程序，就能建立并运行 C 语言程序。

表 2 软件环境说明图

（三）使用的编程语言

C 语言

三、分析与设计

（一）图形设计

1. 背景

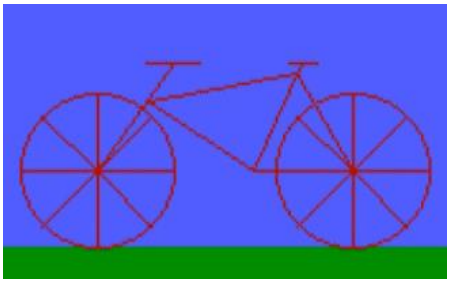
由代码：int far getmaxx(void)与 int far getmaxy(void)获取 x 轴与 y 轴的最大值，从而确定显示区域的大小为 640*480。背景设计如下示意图所示。

2. 自行车框架

接下来要设计自行车的大小，结构与位置。一辆简单的自行车，可以仅由线段和圆两种简单图形组成。首先，确定后轮的圆心位置与半径大小，圆心位置与右边界的距离应该略微大于半径，与下边界（即公路上边）的距离应该等于半径。然后利用 circle 函数画出后轮，再确定前轮与后轮的间距，利用 circle 函数画一个与后轮 x 坐标不同，y 坐标与半径相同的前轮。然后在两个车轮的基础上确定自行车其他结构的相对大小与位置。确定好每个直线的两个端点的坐标，利用 line 函数画出 8 个不同的直线，从而与两个车轮构成一个自行车框架。

3.自行车车轮

由于自行车框架为从右至左平行移动，而车轮滚轴为从右至左旋转移动。因此图形结构设计分开设计。车轮滚轴的设计采用由四个半圆等分一个圆来实现，半圆的颜色为背景色填充。因此，确定半圆（扇形）的圆心位置（即车轮的圆心位置）、半径、起始角与终止角的大小，利用 pieslice 函数勾画出车轮滚轴，前后轮仅圆心位置不同。最终由自行车框架和车轮滚轴构成完整的自行车。示意图如下图所示。



（二）运动设计

1.自行车的运动

自行车框架的运动采用图形存取来实现。图形存取的基本原理是：把屏幕上某个区域的信息存入一个缓冲区，然后在另一个区域把它的内容显示出来。

首先确定要存入与复制的区域，由于需要实现自行车框架的移动，因此确定的区域为包含自行车框架的矩形区域。然后利用 imagesize 函数确定存储图形区域所需的内存大小。接下来利用 getimage 函数将指定区域的图形从屏幕拷贝到内存区域。最后利用 putimage 函数即可将 getimage 函数保存的图形重新在屏幕上显示。参数（x,y）是恢复显示图形左上角的位置。利用 for 循环，即可实现不同位置的重复拷贝，从而具有“移动”的视觉效果。

2.车轮的转动

车轮的转动效果，需要依靠组成车轮的滚轴的旋转来实现。而要想实现旋转效果，需要不断改变滚轴的角度，即扇形的起始角与终止角的角度。扇形的起始角和终止角改变相同的角度，在视觉上就有滚轴旋转的效果。每个扇形都做出

相同的角度改变，在整体上就有车轮旋转的视觉效果。而车轮的从右至左移动只需改变扇形圆心位置即可，改变的大小与自行车框架的移动一致，在视觉上就是自行车的整体移动。因此，同样利用 for 循环来实现车轮的不断转动与车轮的移动。

（三）文字设计

1.提示文字的显示

为了丰富最终效果，可以在屏幕上添加提示文字。利用图形模式下的文本处理的知识来实现。在输出提示文字在屏幕前，需要设置一系列“准备工作”。首先是确定字体类型与字体大小，Turbo C 提供了两种字符类型，分别是点阵字符（位映像字符）与笔画字符（矢量字符）。当然，在这之前，必须保证系统文件夹中有对应的字体文件。然后利用 settextstyle 函数确定字符类型，文本输出方向与字符大小。接下来利用 settextjustify 函数来确定文本对齐方式。然后，利用 setcolor 函数确定文字颜色。最后利用 outtextxy 函数在指定位置输出字符串“Press any key to start the bike.”。

2.提示文字的消失

利用 getch()函数接收用户按下的任意键后要实现提示文字的消失。这里巧妙地采用遮盖的方法实现文字消失的“视错觉”。即再次在同样位置输出同样的字符串，但是字符串颜色与背景色相同。因此，再次输出的字符串与背景融为一体，从而实现字符串的“消失”。

四、实现与结果

（一）具体实现

1.头文件

```
#include"graphics.h"
```

graphics.h 头文件是整个程序中必不可少的头文件。该头文件包含 Turbo C 中 70 多个图形库函数，凡是在程序中要调用图形函数都必须包含该头文件。

```
#include"conio.h"
```

代码中使用 getch()函数需包含该库。getch()函数用以接收用户的任意一个键盘输入。通常用以“暂停”程序。

```
#include"stdlib.h"
```

使用 malloc()函数需包含该库。在把图形存入内存时，需要使用 malloc()函数，用以取得指向图形区域的地址指针。

2.定义变量

```
int gdriver=VGA,gmode=VGAHI;
```

定义显示器类型与显示模式。

```
int bgcolor=9,roadcolor=2,bikecolor=4;
int textcolor1=62,textcolor2=57;
```

bgcolor 为背景颜色变量，roadcolor 为公路颜色变量，bikecolor 为自行车颜色变量，textcolor1 与 textcolor2 为文字颜色变量。不同数值对应不同的颜色。在背景色与数值的对应关系中，9 对应于浅蓝色。在包括 EGA 和 VGA 图形适配器的 EGA 调色板中，2 代表绿色，4 代表红色，62 代表黄色，57 代表浅蓝色。

```
int bikespeed=3,rollspeed=1,time=0;
```

bikespeed 为控制自行车行驶速度变量，即每次复制图形区域的横坐标改变量。rollspeed 为控制自行车车轮转动速度变量，即车轮扇形的角度改变量。time 为延迟函数 delay()的参数，控制程序延迟时间。

```
int i,j;
```

i 为控制自行车移动的循环变量，j 为控制车轮滚轴旋转角度循环变量。

```
void *w;
```

指针，指向图形存储的数组。

3. 图形系统初始化

```
initgraph(&gdriver,&gmode,"");
```

该语句为图形系统初始化代码。前两个参数为图形驱动程序与图形显示模式，与前面的定义相对应。最后的参数表明在当前路径下查找图形驱动程序。

4. 画背景

```
setbkcolor(bgcolor);
```

该语句设置背景颜色。背景色区别前景色，是指整个显示屏的颜色。

```
setfillstyle(1,roadcolor);
```

设置公路填充颜色，setfillstyle 函数的第一个参数为 1，代表实填充模式，第二个参数为填充的颜色。

```
bar(0,400,639,479);
```

画公路。bar 函数的作用是画一个矩形条。四个参数为左上角与右下角的坐标。且用当前填充模式和填充色填充。

5. 画自行车框架

```
setcolor(bikecolor);
```

设置画自行车的画笔颜色。

```
circle(600,370,30);
```

画后轮。

```
circle(500,370,30);
```

画前轮。

```
line(500,370,530,328);
line(519,328,540,328);
line(600,370,561,370);
line(561,370,580,328);
line(586,328,575,328);
```

```
line(600,370,578,332);
line(578,332,520,343);
line(561,370,520,343);
```

用 8 个线条绘制自行车车身。最后与前后轮共同构成自行车框架。

6. 图形存取

```
w=malloc(imagesize(465,325,635,400));
```

确定好存取区域后，利用 imagesize 检测区域所占内存，再利用 malloc 函数取得存取区域的地址指针。

```
getimage(465,325,635,400,w);
```

将图像，即自行车框架存入内存。前四个参数 imagesize 函数中的参数相同。w 是一个 void 类型指针，需要复制的屏幕区域将保存在由它指向的数组中。

7. 画车轮滚轴

```
setfillstyle(0,bgcolor);
```

设置画车轮滚轴填充颜色，setfillstyle 函数的第一个参数为 0，代表背景颜色填充模式，第二个参数为填充的背景颜色。

```
pieslice(600,370,90,270,30);
pieslice(600,370,135,315,30);
pieslice(600,370,180,360,30);
pieslice(600,370,225,45,30);
pieslice(500,370,90,270,30);
pieslice(500,370,135,315,30);
pieslice(500,370,180,360,30);
pieslice(500,370,225,45,30);
```

分别用 4 个扇形绘制前后车轮滚轴。前两个参数为圆心坐标，后面两个参数为扇形的起始角与终止角的角度，最后的参数为扇形半径大小。

8. 提示文字

```
settextstyle(0,0,2);
```

设置字体类型与大小。settextstyle 的第一个参数为字体类型，0 代表 8×8 点阵字符（默认值）；第二个参数为文本输出方向，0 代表从左至右输出（默认）；第三个参数为字符大小，2 代表 16×16 点阵。

```
settextjustify(1,1);
```

设置字体的对齐方式。settextjustify 函数的第一个参数为水平方向的对齐方式，1 代表中间对齐；第二个参数为垂直方向的对齐方式，1 代表中间对齐。

```
setcolor(textcolor1);
```

设置字体颜色。

```
outtextxy(320,240,"Press any key to start the bike.");
```

用当前画笔颜色在屏幕中间输出提示字符串。前两个参数代表输出位置，最后的参数代表输出的字符串内容。该字符串的含义是“按下任意键以驱动自行车。”

```
getch();
```

等待用户按下任意键。

```
setcolor(textcolor2);
```

改变字体颜色。

```
outtextxy(320,240,"Press any key to start the bike.");
```

将颜色为背景色的原字符串内容覆盖至原位置，实现字符串“消失”。

9. 自行车的移动

```
setcolor(bikecolor);
```

改变画笔颜色。以便后续重复绘制车轮滚轴。

```
for(i=bikespeed,j=rollspeed;i<=465;i+=bikespeed,j+=rollspeed)
```

利用 for 循环来实现自行车的整体运动。i<=465 控制自行车行驶至左边界时停止运动。i+=bikespeed, j+=rollspeed 控制自行车行驶速度和车轮旋转速度。

```
{
   putimage(465-i,325,w,0);
```

循环绘制自行车框架实现自行车运动。第一个参数为存取区域左上角的横坐标；第二个参数为纵坐标，由于自行车平行移动，所以纵坐标不变；第三个参

数为指向 void 类型的指针，它指向用 getimage 函数复制的数组；最后一个参数是图形复制到屏幕上的显示模式，0 代表原样拷贝到屏幕。

```
    pieslice(600-i,370,90+j,270+j,30);
    pieslice(600-i,370,135+j,315+j,30);
    pieslice(600-i,370,180+j,360+j,30);
    pieslice(600-i,370,225+j,45+j,30);
    pieslice(500-i,370,90+j,270+j,30);
    pieslice(500-i,370,135+j,315+j,30);
    pieslice(500-i,370,180+j,360+j,30);
    pieslice(500-i,370,225+j,45+j,30);
```

循环绘制不同角度的滚轴实现车轮转动。第一个参数是扇形的圆心横坐标，由于与自行车共同平行移动，所以改变量 i 与自行车框架相同；第二个参数是扇形的圆心纵坐标，由于自行车平行移动，所以纵坐标不变；第三个参数与第四个参数分别是扇形的起始角与终止角的角度，变化量相同；最后一个参数是扇形的半径大小。

```
    delay(time);
```

增大 time 的值可使自行车“减速”。

```
}
```

10. 退出

```
getch();
```

等待用户按下任意键。

```
closegraph();
```

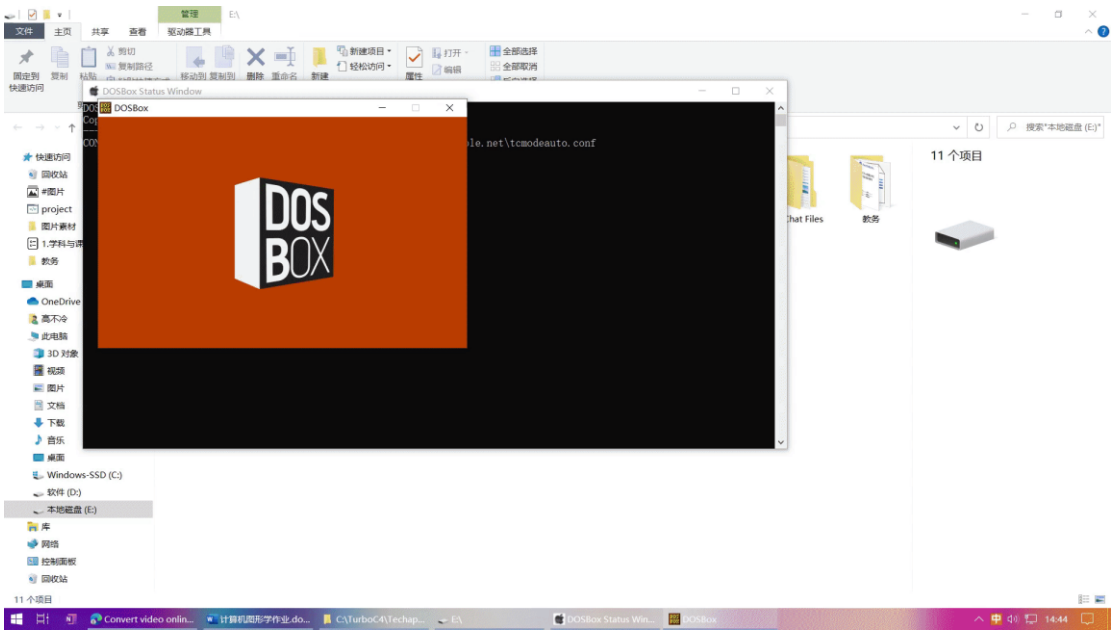
退出图形模式。

```
return 0;
```

程序结束。

（二）最终效果

程序的最终运行效果如下面得到动态图所示。



五、问题与解决

在完成本项目的过程中，我遇到了许多可预料与不可预料的问题。在发现问题，积极思考对策，不断实践的过程中我成功解决了这些问题。

1. 问题一

（1）自行车架构问题：在初步设计自行车框架时，如何确定各个线段的两点位置，从而构建出符合预期的自行车框架。

（2）解决：从后轮到前轮，再到前轮杆等等，不断选取合适的参考，采用预估的思想，再运用基本的数学知识计算出未知点的坐标，最后不断修改与调整。

2. 问题二

（1）图形存取问题：在设计图形存取来实现自行车的运动时，由于缺少 C 语言指针的相关知识，导致频繁出错，无法实现预期的效果。

（2）解决：仔细阅读教材，参考教材相关处理过程，最终成功实现图形存取。

3. 问题三

（1）文字消失问题：在设计好提示文字后，无法成功实现文字的消失。

（2）解决：将颜色为背景色的原字符串内容输出至原位置，实现字符串“消失”。

4. 问题四

（1）for 循环问题：在设计 for 循环时，出现未知错误。

（2）解决：通过不断检查出错位置的代码，发现问题所在：for 循环参数之间用分号给开而不是逗号。改正后错误得以解决。

5. 问题五

（1）最终效果问题：在最终效果的呈现上，发现运动的自行车总是“一闪一闪”的。

（2）解决：仔细思考后，我认为应该是在 for 循环中，首先实现图形区域，即自行车框架的拷贝输出，然后是车轮滚轴的移动，再然后又是自行车框架的拷贝输出，再然后是车轮滚轴的移动。如此循环，造成了交替覆盖问题，从而出现“一闪一闪”的视觉效果。解决方法就是不使用图形存取来实现自行车框架的移动，采用和车轮滚轴相同的方式，针对自行车框架的每个部分逐“帧”移动，缺点是代码较为冗杂。

六、参考资料

为了完成本次项目，我主要参考和使用了以下资料和网站：

[1]王汝传.计算机图形学教程（第 3 版）[M].北京：人民邮电出版社，2014：46-80.

[2] Turbo C4.0 官方下载|Turbo C++ V4.0 官方版 下载_当下软件园_软件下载. Downxia. http://www.downxia.com/downinfo/273931.html

[3] TurboC 7 2.1 Download (Free) - DB.EXE. Turboc-7-by-akki. https://turboc-7-by-akki.software.informer.com/2.1/

[4]Graphics (gdiplusgraphics.h) - Win32 apps | Microsoft Learn. Learn. https://learn.microsoft.com/en-us/windows/win32/api/gdiplusgraphics/nl-gdiplusgraphics-graphics

[5] How to add "graphics.h" C/C++ library to gcc compiler in Linux - GeeksforGeeks. Geeksforgeeks. https://www.geeksforgeeks.org/add-graphics-h-c-library-gcc-compiler-linux/

附录 源代码

```
/*BIKE.c--Write a program for a bicycle traveling
 fast from right to left on a road.*/
#include"graphics.h"
#include"conio.h"//代码中使用 getch()函数需包含该库
#include"stdlib.h"//使用 malloc()函数需包含该库

int main(void)
{
    int gdriver=VGA,gmode=VGAHI;

    int bgcolor=9,roadcolor=2,bikecolor=4;
/*bgcolor 为背景颜色变量，roadcolor 为公路颜色变量，bikecolor 为自行
车颜色变量*/

    int textcolor1=62,textcolor2=57;//文字颜色
    int bikespeed=3,rollspeed=1,time=0;
/*bikespeed 为控制自行车行驶速度变量，rollspeed 为控制自行车车轮转动
速度变量，time 为延迟函数 delay()的参数，控制延迟时间*/

    int i,j;/*i 为控制自行车移动的循环变量，j 为控制车轮滚轴旋转角度
循环变量*/
    void *w;//指针，指向图形存储的数组

    initgraph(&gdriver,&gmode,"");//初始化

    setbkcolor(bgcolor);//设置背景颜色
    setfillstyle(1,roadcolor);//设置公路填充颜色
    bar(0,400,639,479);//画公路

    setcolor(bikecolor);//设置画自行车的画笔颜色
    circle(600,370,30);//画后轮
    circle(500,370,30);//画前轮
    line(500,370,530,328);
    line(519,328,540,328);
    line(600,370,561,370);
    line(561,370,580,328);
    line(586,328,575,328);
    line(600,370,578,332);
    line(578,332,520,343);
    line(561,370,520,343);//用多个线条绘制自行车车身

    w=malloc(imagesize(465,325,635,400));//检测图形区所占内存区域
```

```
    getimage(465,325,635,400,w);//将图像，即自行车框架存入内存

    setfillstyle(0,bgcolor);//设置画车轮滚轴填充颜色
    pieslice(600,370,90,270,30);
    pieslice(600,370,135,315,30);
    pieslice(600,370,180,360,30);
    pieslice(600,370,225,45,30);
    pieslice(500,370,90,270,30);
    pieslice(500,370,135,315,30);
    pieslice(500,370,180,360,30);
    pieslice(500,370,225,45,30);//用多个扇形绘制车轮滚轴

    settextstyle(0,0,2);//设置字体类型与大小
    settextjustify(1,1);//设置字体的对齐方式
    setcolor(textcolor1);//设置字体颜色
    outtextxy(320,240,"Press any key to start the bike.");
//在屏幕中间输出提示字符串
    getch();//等待按下任意键
    setcolor(textcolor2);//改变字体颜色
    outtextxy(320,240,"Press any key to start the bike.");
//将颜色为背景色的原字符串内容覆盖至原位置，实现字符串“消失”

    setcolor(bikecolor);//改变画笔颜色
    for(i=bikespeed,j=rollspeed;i<=465;i+=bikespeed,j+=rollspee
d)//实现自行车整体运动
    {
        putimage(465-i,325,w,0);//循环绘制自行车框架实现自行车运动
        pieslice(600-i,370,90+j,270+j,30);
        pieslice(600-i,370,135+j,315+j,30);
        pieslice(600-i,370,180+j,360+j,30);
        pieslice(600-i,370,225+j,45+j,30);
        pieslice(500-i,370,90+j,270+j,30);
        pieslice(500-i,370,135+j,315+j,30);
        pieslice(500-i,370,180+j,360+j,30);
        pieslice(500-i,370,225+j,45+j,30);
       //循环绘制不同角度的滚轴实现车轮转动
        delay(time);//增大 time 的值可使自行车“减速”
    }

    getch();//按任意键退出
    closegraph();
    return 0;
}
```