序号: _____

学号: 15416117

常州大学

课 程 设 计

设计课程名称: 《C 语言课程设计》

题 目: 数字图像处理—几何变换 1

学 生 姓 名: 蒋炜成

学 院: 华罗庚学院 专 业 班 级: 华院 152

指 导 教 师: 程起才 专业技术职务: 讲师

设计时间: 2016 年 6 月 20 日 ~ 2016 年 7 月 1 日

目录

1.任务陈述

1.1 项目意义陈述

进入 21 世纪以来，我国信息产业在生产和科研领域都出现了长足的进步，并成为国民经济的支柱产业之一。数字图像处理作为信息产业的重要一环，从 20 世纪 20 年代第一张数字图像通过海底电缆从伦敦传送至纽约以来，数字图像的处理收到了充分的关注和普遍的运用。图像处理科学与国民计生关系密切的学科，他能够为人类带来巨大的经济与社会效益。

本课程设计要求我们利用 OpenCV 1.0。OpenCV 作为一个轻量级且高效的跨平台图像处理库，为图像处理和计算机视觉提供了很多算法。我们需要通过利用其图像容器，对图像进行各种操作。

我们通过学习简单的数字图像处理技术，对数字图像处理有一个感性的认识，对以后涉足此领域提供了兴趣基础。

1.2 项目设计任务

1.2.1 题目描述

读入一幅彩色的数字图像，完成一系列的几何运算，并分别输出每个运算的效果图。

1.2.2 题目要求

（1）先将彩色图像变为灰度图像；

（2）然后将灰度图像旋转任意角度；如果能对彩色图像进行相应旋转加分；

（3）最后将灰度图像和彩色图像进行水平镜像和垂直镜像；

（4）自定义函数完成将图像保存为 bmp 格式系统功能结构和调用关系。

1.2.3 输入/输出要求

（1）应用程序运行后，先显示一个菜单，然后用户根据需要选择相应的操作项目。进入每个操作后，根据程序的提示输入相应的信息；

（2）输出每个功能的效果图。

1.2.4 编写源程序的要求：

(1)能够实现任务书中的功能；

(2)尽可能使界面友好、直观、易操作；

(3)源程序要有适当的注释，使程序容易阅读。

2. 功能介绍和流程图

该程序主要实现对彩色图像或灰度图像分别进行旋转、镜像和保存的功能

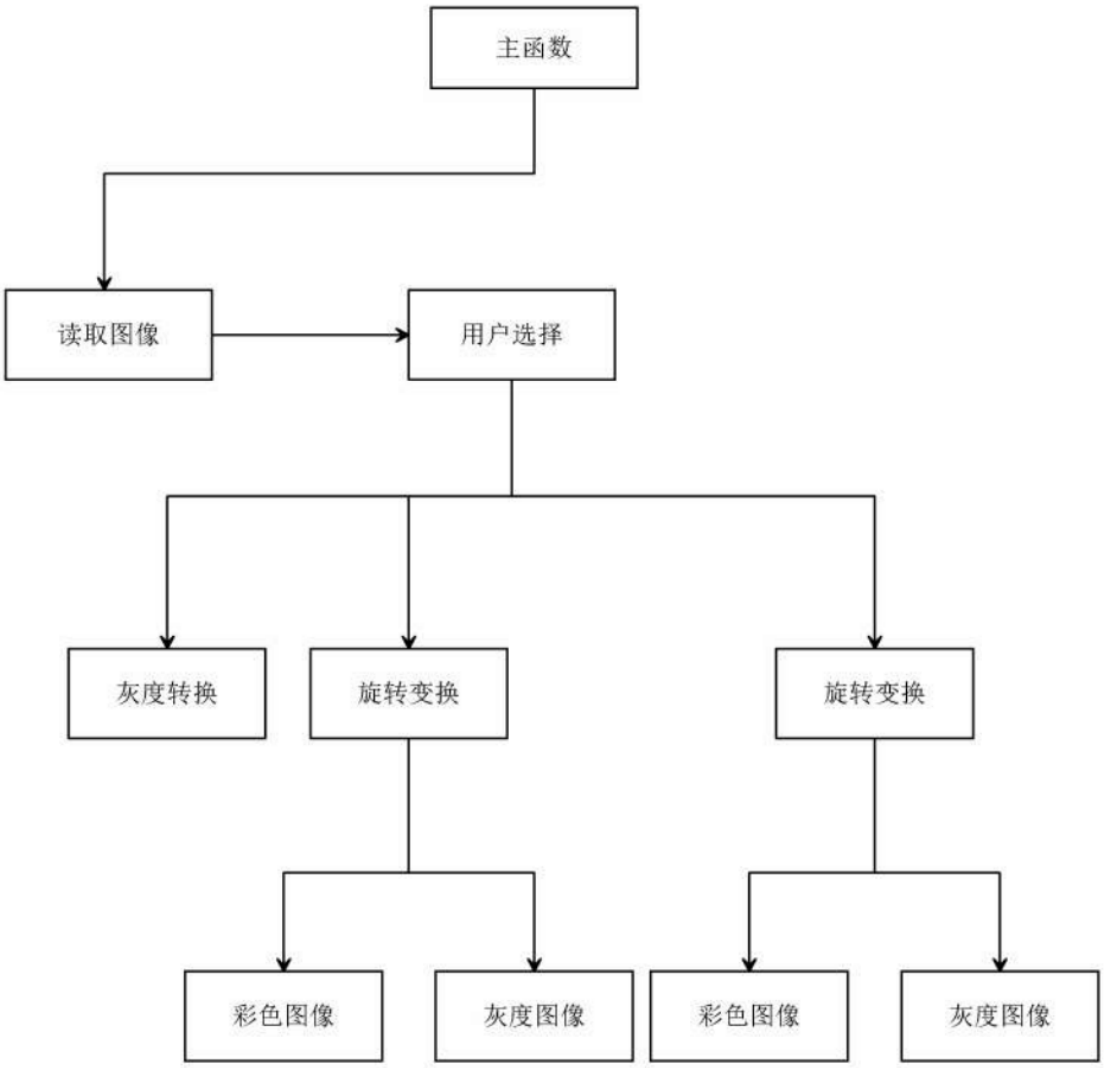


图 3-1-1 程序总体设计图

3. 设计详情

3.1 主函数功能

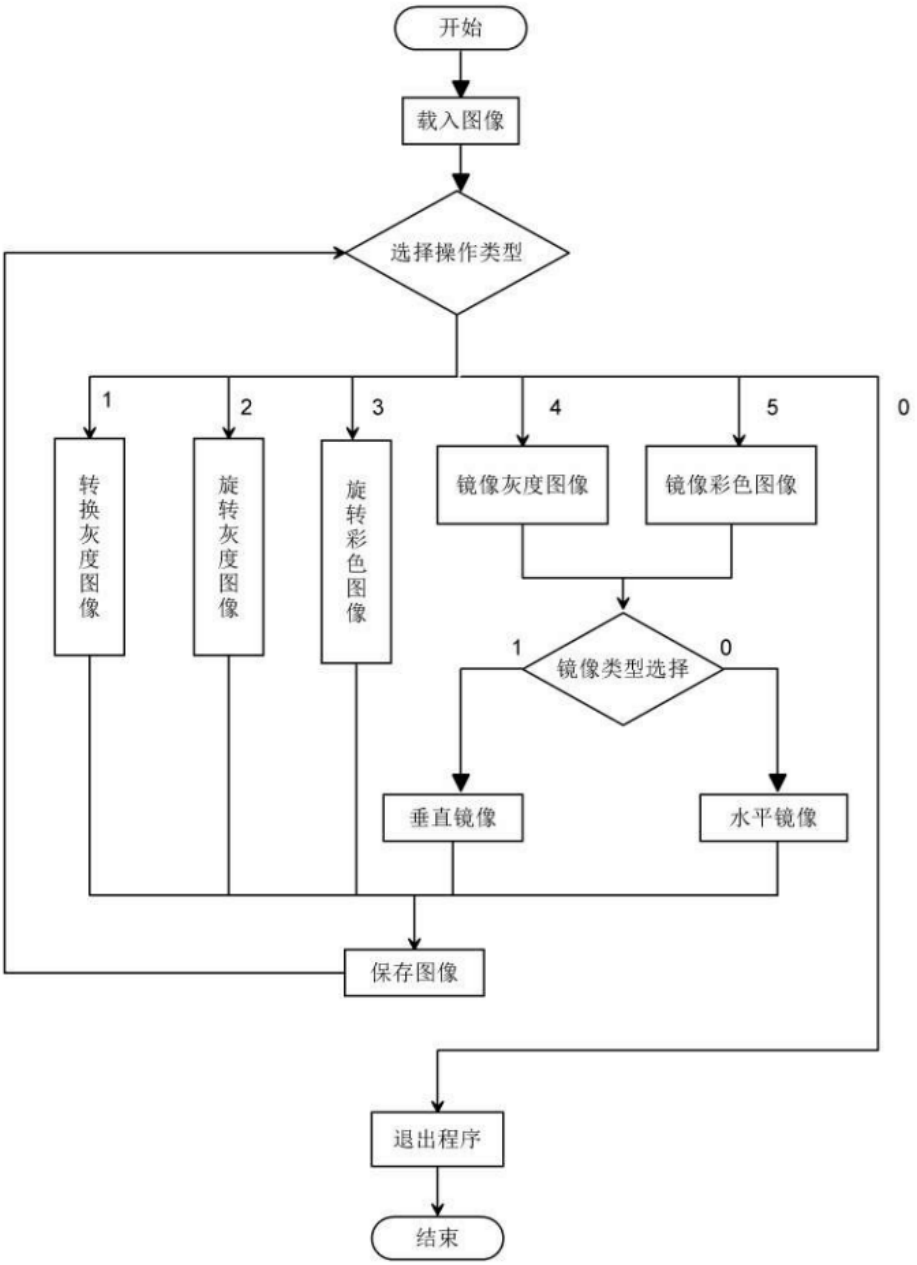
（1） 主函数原型：int main()

（2） 功能：声明版权，在主函数里调用功能函数，实现功能。

（3） 代码：

```
int main()
{
    printf("  数字图像处理\n"
           "几何变换 1\n
           "By 华院152 蒋炜成\n\n");
    loadimg();
    return 0;
}
```

3.2 函数调用关系



3.2 读取图像

（1） 函数原型：void loadimg()；

（2） 功能：使用 cvLoadimg 函数读取图像存在 IplImage 类指针中，判断图像是否正确读取，若正确读取则显示图像并显示图像信息，否则退出程序；

（3） 代码

```
void loadimg()//加载图像并且显示图像信息,提供操作选项
{
    img=cvLoadImage("input.png");
    if(!img)
        {
            printf("无法加载本目录以下图片: %s\n, 请把图片改名为\"input.png\"放于本目录下","input.png");
            getchar();
        }
    else
    {
        height    = img->height;//图片参数
        width     = img->width;
        step      = img->widthStep;
        channels  = img->nChannels;
        depth     = img->depth;
        data      = (uchar *)img->imageData;
        printf("程序正在处理一个尺寸为%dx%d的%d通道%d位图像\n",height,width,depth,channels);
        cvNamedWindow("原图",CV_WINDOW_AUTOSIZE);//创建窗口
        cvShowImage("原图",img);
        cvWaitKey(0);
        cvDestroyWindow("原图");
        slco();
    }
}
```

3.3 选择操作类型函数

（1） 函数原型：void slco()；

（2） 功能：打印界面，提示选择程序操作；

说明：通过循环调用，程序将反复调用选择操作函数。

（3） 代码：

3.4 灰度图像转换函数

（1） 函数原型：void toGrayimg()；

（2） 功能：把图像转换成灰度图像，并调用保存图像函数保存图像；

（3） 代码：

```
void toGrayimg()//转换灰度图像
{
    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
    //使用库转换图像空间
    cvCvtColor(img,result,CV_RGB2GRAY);
    saveimg(result);
}
```

3.6 灰度图像旋转函数

（1） 函数原型：void roGimg()；

（2） 功能：先构建单通道图像，把图像转换为灰度图像后旋转，先将图像转换为 cvMat 类型，构建变换函数，利用 cvWarpAffine 的图像仿射函数实现旋转后存在单通道图像内传递给保存函数。

（3） 代码：

```
void roGimg()
{
    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
    IplImage *res=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
    cvCvtColor(img,result,CV_RGB2GRAY);
    double angle;
    printf("请输入要旋转的角度: ");
    scanf("%lf", &angle);
    float m[6];
    CvMat rotMat = cvMat(2, 3, CV_32F, m);
    CvPoint2D32f center;
    center.x = (float) (result->width / 2.0 + 0.5);
    center.y = (float) (result->height / 2.0 + 0.5);
    cv2DRotationMatrix(center, -angle, 1, &rotMat);
    cvWarpAffine(result,res, &rotMat, CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
    //图像仿射变换函数
    saveimg(res);
}
```

3.7 彩色图像旋转

（1） 函数原型：void roRGBimg()；

（2） 功能：先将图像转换为 cvMat 类型，构建变换函数，利用 cvWarpAffine 的图像仿射函数实现旋转

（3） 代码 switch-case 选择结构撰写。

```
void roRGBimg()
{
    double angle;
    printf("请输入要旋转的角度: ");
    scanf("%lf", &angle);
    float m[6];
    CvMat rotMat = cvMat(2, 3, CV_32F, m);
    CvPoint2D32f center;
    center.x = (float) (img->width / 2.0 + 0.5);
    center.y = (float) (img->height / 2.0 + 0.5);//计算二维旋转的仿射变换矩阵
    cv2DRotationMatrix(center, -angle, 1, &rotMat);
    IplImage *result = cvCreateImage(cvGetSize(img),8,3);
    cvWarpAffine(img, result, &rotMat, CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
    //图像仿射变换函数
    saveimg(result);
}
```

3.8 选择镜像图像操作种类

（1） 函数原型：灰度图像 void seltype()； 彩色图像 void seltype1()；

（2） 功能：seltype()中现将图像转换至灰度图像，再根据传递给变换函数。seltype1()中是根据用户指令直接传递至变换函数

（3） 代码：此处为 seltype()函数代码作为样例

```
void seltype()
{
    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
    cvCvtColor(img,result,CV_RGB2GRAY);
    int q;
    printf("请选择镜像类型"
        "0------水平镜像\n"
        "1------垂直镜像\n");
    scanf("%d",&q);
    switch(q)
    {
        case 0:
            rollkd1(result);

        case 1:
            rollkd2(result);
    }
}
```

3.9 水平镜像函数

（1） 函数原型：void rollkd1(IplImage *src);

（2） 功能：同样利用 cvWarpAffine 函数，通过构建变换矩阵 m

$$\begin{pmatrix} -1.0f & 0 & width \\ 0 & 1.0f & 0 \end{pmatrix}$$

其中 width 表示图像宽度，利用函数进行水平镜像变换。

（3） 代码：

```
void rollkd1(IplImage *src)//水平镜像
{
    float m[6] = {-1.0f, 0, (float) src->width,
                  0, 1.0f, 0};//构建变换矩阵
    CvMat fliper = cvMat(2, 3, CV_32F, m);//构建Mat
    //现场判断位深和通道以免出错
    IplImage *result = cvCreateImage(cvGetSize(src), src->depth, src->nChannels);
    cvWarpAffine(src, result, &fliper, CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
    saveimg(result);
}
```

（4） 说明：src 表示原图像，result 表示目标图像，&flipper 表示变换矩阵，CV_INTER_LINEAR 表示填充像素，用来 cvScalarALL(0)表示图像通道。

3.10 垂直镜像函数

（1） 函数原型：void rollkd2(IplImage *src);

（2） 功能：和水平镜像类似，对图像进行垂直变换，构建矩阵如下：

$$\begin{pmatrix} 1.0f & 0 & 0 \\ 0 & -1.0f & width \end{pmatrix}$$

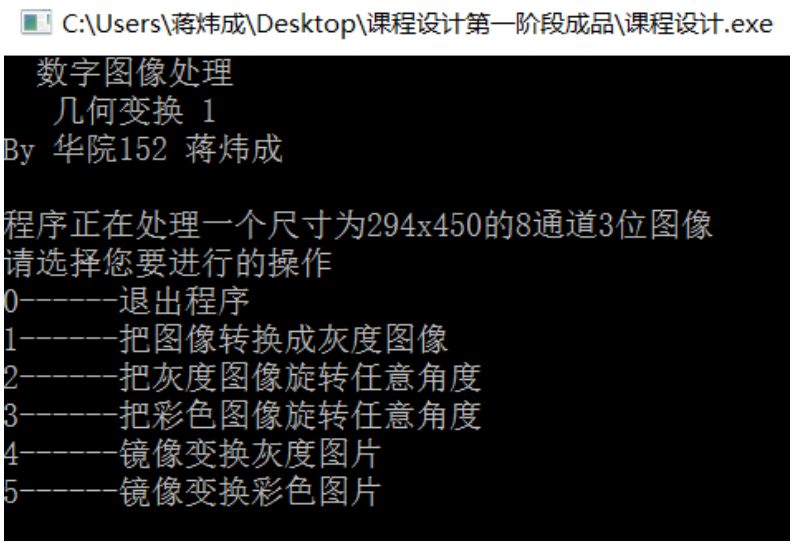
（4） 代码：

```
void rollkd2(IplImage *src)//垂直镜像
{
    float m[6] = {1.0f, 0, 0,
                  0, -1.0f, (float) src->height};
    CvMat fliper = cvMat(2, 3, CV_32F, m);
    IplImage *result = cvCreateImage(cvGetSize(src), src->depth, src->nChannels);
    cvWarpAffine(src, result, &fliper, CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
    saveimg(result);
}
```

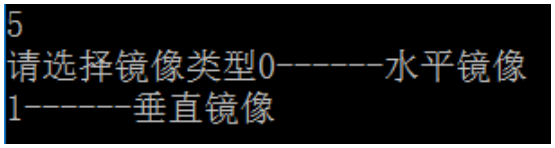
4. 调试运行与输出结果

4.1 程序界面

4.1.1 主界面操作选择



4.1.2 选择镜像类型



4.1.3 结果输出

4.2 程序输出结果展示

图4-2-1 原图

图4-2-2 转换灰度图像结果

图4-2-3 对灰度图像旋转90度结果

图4-2-4 对灰度图像旋转90度结果

图4-2-5 对灰度图像水平翻镜像结果

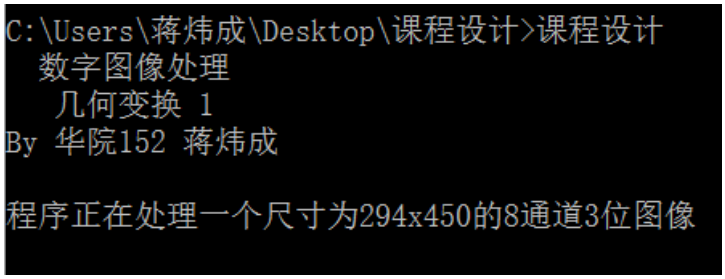
图4-2-5 对彩色图像水平翻镜像结果

图4-2-5 对灰度图像垂直翻镜像结果

图4-2-5 对彩色图像处置翻镜像结果

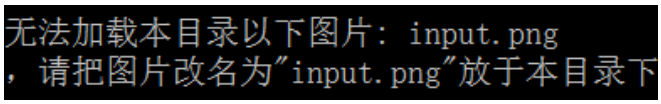
5. 程序使用说明

（1）使用 cmd 命令进入程序

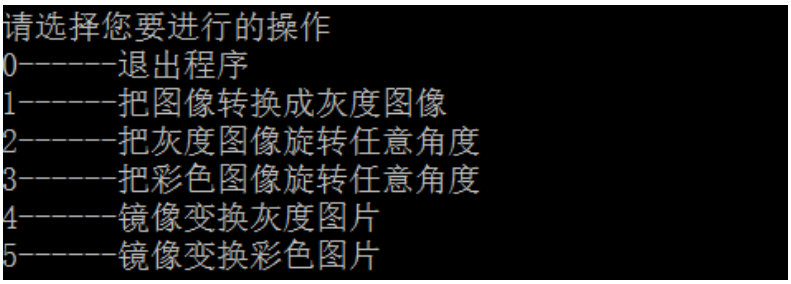


（2）图片显示跳出后按任意键进入主界面操作。

（3）若图片无法成功读取，请确保文件格式为 png 且文件名为 input，错误提示将如下图所示，



（4）成功读取图片后，进入菜单选择要进行的操作



（5） 每次操作，图片都会跳出显示转换后图片，并按任意键退出。

6. 心得体会

本次课程设计，如果使用 OpenCV 库，难度还是可以的。我也在尝试写纯 C 代码，看了网上不少别人的代码和资料，尝试着写了纯 C 代码，始终在构建点操作的矩阵时程序会报错，花了一个通宵写了两百行纯 C 却发现 bug 多的难以修改。我对数字图像处理本身就有着浓烈的兴趣，也一直有阅读冈萨雷斯的《数字图像处理》这本大部头的书，也和侯老师参加图像处理的讨论班。对数字图像处理的原理有一定了解，对其中也略知一二。这是第一次自己真正参与编写这样的代码，发现不依赖 OpenCV 库就难以编写图像变换程序，让我深知编写底层代码的不易，对现代编程有完整的库可以调用深感幸运。

这次编写提高了我写代码的水平和理解能力，如果有更加充足的时间，我想我的程序还有改进空间：第一，是尽量不使用库函数而是尽量使用纯 C 代码编写，对于图像点运算只是简单地数组变换并不难，重点就在于怎么将图像填充在数组之内。第二，扩充程序功能，是程序有更多的功能例如拉伸等。第三，让程序有界面，是用户能通过简单地点击鼠标实现功能，更具有美观性。

最后感谢老师这几天以来的教导，能让我们获得知识，也复习了 C 语言，为接下啦爱的考试做准备。

7. 问题解决办法

在程序编写过程之中，我遇到了一些困难，通过查找资料顺利的解决了这些问题。

首先是处理永远在循环之中无法正常退出，后来检查时发现忘记加 break 控制语句，实属粗心不应该。

其次是镜像变换时 OpenCV 总是会提示图像通道错误，后来发现是灰度图像和彩色图像通道不一致所致，于是在 190 行（代码见下）处的创建目标图像时判断传入图像的通道得以解决。

还有的问题未能及时记录，但是也通过查阅书籍解决。其中《OpenCV 教程:基础篇》为我提供了极大地帮助，促进了代码的编写，让我能够顺利完成课程设计。

8. 附录

8.1 源程序代码

```
1. #include <stdlib.h>
2. #include <stdio.h>
3. #include <cv.h>
4. #include <highgui.h>
5. IplImage *img = 0;  //定义图像
6. int height,width,step,channels,depth;
7. int i,j,k;
8. uchar *data;
9. //定义函数
10.void loadimg();//加载图像并且显示图像信息
11.void slco();//选择图像操作
12.void saveimg();//保存图像
13.void toGrayimg();//转换灰度图像
14.void roGimg();//旋转灰度图像
15.void roRGBimg();//旋转彩色图像
16.void rollGimg();//翻转灰度图像
17.void rollRGBimg();//翻转彩色图像
18.void seltype();//选择灰度图片镜像种类
19.void seltype1();//选择彩色图片镜像种类
20.void rollkd1(IplImage *src);//水平镜像
21.void rollkd2(IplImage *src);//垂直镜像
22.int main()
23.{
24.printf("  数字图像处理\n    几何变换 1\nBy 华院152 蒋炜成\n\n");
25.loadimg();
26.return 0;
27.}
28.void loadimg()//加载图像并且显示图像信息,提供操作选项
29.{
30.img=cvLoadImage("input.png");
31.if(!img)
32.{
33.printf("无法加载本目录以下图片: %s\n, "
34."请把图片改名为\"input.png\"放于本目录下","input.png");
35.getchar();
```

```
36.}
37.else
38.{
39.height    = img->height;//图片参数
40.width     = img->width;
41.step      = img->widthStep;
42.channels  = img->nChannels;
43.depth     = img->depth;
44.data      = (uchar *)img->imageData;
45.printf("程序正在处理一个尺寸为%dx%d的%d通道%d位图像
   \n",height,width,depth,channels);
46.cvNamedWindow("原图",CV_WINDOW_AUTOSIZE);//创建窗口
47.cvShowImage("原图",img);
48.cvWaitKey(0);
49.cvDestroyWindow("原图");
50.slco();
51.}
52.}
53.
54.void slco()
55.{
56.int n;
57.printf("请选择您要进行的操作\n"
58."0------退出程序\n"
59."1------把图像转换成灰度图像\n"
60."2------把灰度图像旋转任意角度\n"
61."3------把彩色图像旋转任意角度\n"
62."4------镜像变换灰度图片\n"
63."5------镜像变换彩色图片\n");
64.scanf("%d",&n);
65.switch(n)
66.{
67.
68.case 0:
69.break;
70.
71.case 1:
```

```
72. toGrayimg();
73. break;

74. case 2:
75. roGimg();
76. break;
77.
78. case 3:
79. roRGBimg();
80. break;
81.
82. case 4:
83. seltype();
84. break;
85.
86. case 5:
87. seltype1();
88. break;
89.
90. }
91. }
92.
93. void saveimg(IplImage *src)//保存图像
94. {
95. cvSaveImage("out.bmp",src);
96. if(!cvSaveImage("out.bmp",src))
97. printf("Could not save: %s\n","output.bmp");
98. else
99. printf("图像已经保存为%s。\n","output.bmp");
100.    cvNamedWindow("转换后",CV_WINDOW_AUTOSIZE);//创建窗口
101.    cvShowImage("转换后",src);
102.    cvWaitKey(0);
103.    cvDestroyWindow("转换后");
104.    slco();
105.    }

106.    void toGrayimg()//转换灰度图像
```

```
107.    {
108.    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);//使用
    库转换图像空间
109.    cvCvtColor(img,result,CV_RGB2GRAY);
110.    saveimg(result);
111.    }
112.
113.    void roGimg()
114.    {
115.    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
116.    IplImage *res=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
117.    cvCvtColor(img,result,CV_RGB2GRAY);
118.    double angle;
119.    printf("请输入要旋转的角度: ");
120.    scanf("%lf", &angle);
121.    float m[6];
122.    CvMat rotMat = cvMat(2, 3, CV_32F, m);
123.    CvPoint2D32f center;
124.    center.x = (float) (result->width / 2.0 + 0.5);
125.    center.y = (float) (result->height / 2.0 + 0.5);
126.    cv2DRotationMatrix(center, -angle, 1, &rotMat);
127.    cvWarpAffine(result,res, &rotMat, CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS,
    cvScalarAll(0));
128.    //图像仿射变换函数
129.    saveimg(res);
130.    }
131.
132.    void roRGBimg()
133.    {
134.    double angle;
135.    printf("请输入要旋转的角度: ");
136.    scanf("%lf", &angle);
137.    float m[6];
138.    CvMat rotMat = cvMat(2, 3, CV_32F, m);
139.    CvPoint2D32f center;
140.    center.x = (float) (img->width / 2.0 + 0.5);
141.    center.y = (float) (img->height / 2.0 + 0.5);//计算二维旋转的仿射变换矩阵
```

```
142.    cv2DRotationMatrix(center, -angle, 1, &rotMat);
143.    IplImage *result = cvCreateImage(cvGetSize(img),8,3);
144.    cvWarpAffine(img, result, &rotMat, CV_INTER_LINEAR |
   CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
145.    //图像仿射变换函数
146.    saveimg(result);
147.    }
148.
149.    void seltype()
150.    {
151.    IplImage *result=cvCreateImage(cvSize(width,height),IPL_DEPTH_8U,1);
152.    cvCvtColor(img,result,CV_RGB2GRAY);
153.    int q;
154.    printf("请选择镜像类型"
155.    "0------水平镜像\n"
156.    "1------垂直镜像\n");
157.    scanf("%d",&q);
158.    switch(q)
159.    {
160.    case 0:
161.    rollkd1(result);
162.
163.    case 1:
164.    rollkd2(result);
165.    }
166.    }
167.
168.    void seltype1()
169.    {
170.    IplImage *result=img;
171.    int q;
172.    printf("请选择镜像类型"
173.    "0------水平镜像\n"
174.    "1------垂直镜像\n");
175.    scanf("%d",&q);
176.    switch(q)
177.    {
```

```
178.    case 0:
179.    rollkd1(result);

180.    case 1:
181.    rollkd2(result);
182.    }
183.    }
184.
185.    void rollkd1(IplImage *src)//水平镜像
186.    {
187.    float m[6] = {-1.0f, 0, (float) src->width,
188.    0, 10f, 0};//构建变换矩阵
189.    CvMat fliper = cvMat(2, 3, CV_32F, m);//构建Mat
190.    IplImage *result = cvCreateImage(cvGetSize(src), src->depth,
   src->nChannels);//现场判断位深和通道以免出错
191.    cvWarpAffine(src, result, &fliper, CV_INTER_LINEAR |
   CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
192.    saveimg(result);
193.    }
194.
195.    void rollkd2(IplImage *src)//垂直镜像
196.    {
197.    float m[6] = {1.0f, 0, 0,
198.    0, -1.0f, (float) src->height};
199.    CvMat fliper = cvMat(2, 3, CV_32F, m);
200.    IplImage *result = cvCreateImage(cvGetSize(src), src->depth,
   src->nChannels);
201.    cvWarpAffine(src, result, &fliper, CV_INTER_LINEAR |
   CV_WARP_FILL_OUTLIERS, cvScalarAll(0));
202.    saveimg(result);
203.    }
204.    //程序结束
```

9. 参考文献

[1] 《数字图像处理》 第三版 冈萨雷斯编著

[2] 《OpenCV 教程:基础篇》 刘瑞祯，于仕琪编著