

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 机器感知实验 |
| 姓 名： | 金镇雄 |
| 学 院： | 元培 |
| 系： | 智能科学系 |
| 专 业： | 智能科学与技术 |
| 年 级： | 19 |
| 学 号： | 1900094619 |
| 指导教师： | 曲天书 |
| 职 称： | 副教授 |

2022年 4月 25日

**实验四、图像变换**

1. **实验目的**

* 使用MATLAB系统进行图像变换
* 图像变换包括形状变换和颜色变换
  + 图像变换：平移变换、尺度变换、旋转变换等
  + 颜色变换：冰冻效果、融炎效果、漫画效果、羽化效果、缩放模糊效果、LOMO特效、电影效果等

1. **实验要求**

* 界面清晰美观
* 可播放原始图像和变换的图像
* 实验结果分析
* 实验讨论

1. **实验仪器设备**

PC机、图像文件

1. **实验原理**
   1. 图像变换

表示原始图像中像素的位置，表示变换后的像素位置。

1. 平移变换

为x轴和y轴的平移量

平移变换的变换矩阵为：

1. 尺度变换

为x轴和y轴尺度变换的大小

尺度变换的变换矩阵为：

1. 旋转变换

为旋转角

旋转变换的变换矩阵为：

1. 错切变换

为错切变换量

错切变换的变换矩阵为：

* 1. 颜色变换

为变换后的像素值，为变换前的像素值。

1. 漫画特效

漫画特效的像素替换公式为：

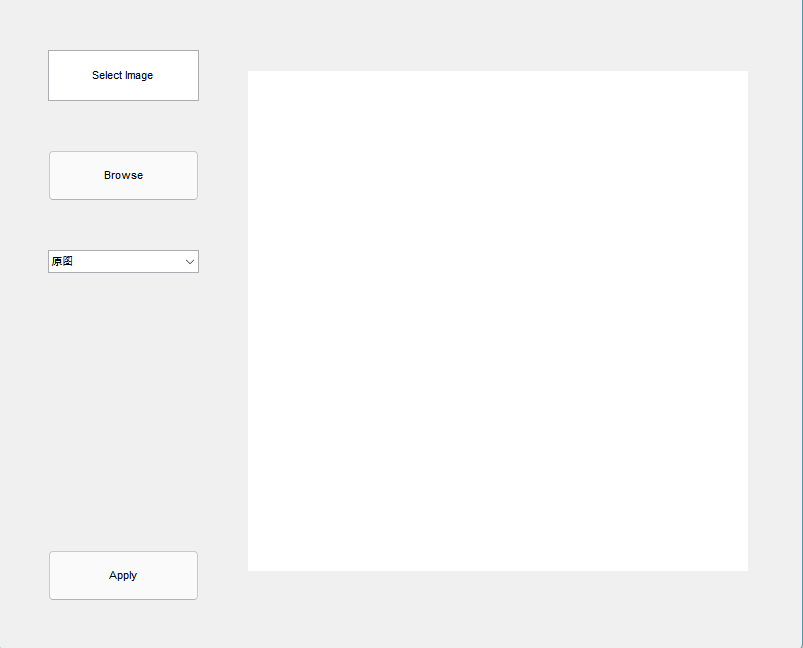
1. 冰冻特效

冰冻特效的像素替换公式为：

1. 熔炎特效

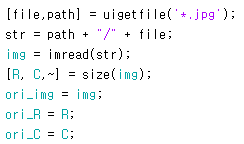
熔炎特效的像素替换公式为：

1. **实验内容和步骤**
   1. 设计GUI界面



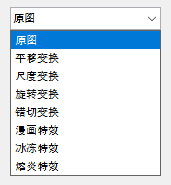
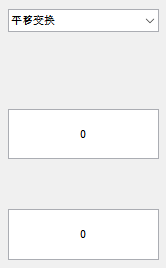
* 1. 读入原始图像

用imread函数读入图像，函数的返回值是大小为height×width×3的矩阵。并用size函数获取图像的宽和长。矩阵的每个元素为uint8类型的RGB值之一，其范围为0~255。为了以后进行图像变换后复原原始图像，另外保存图像信息。



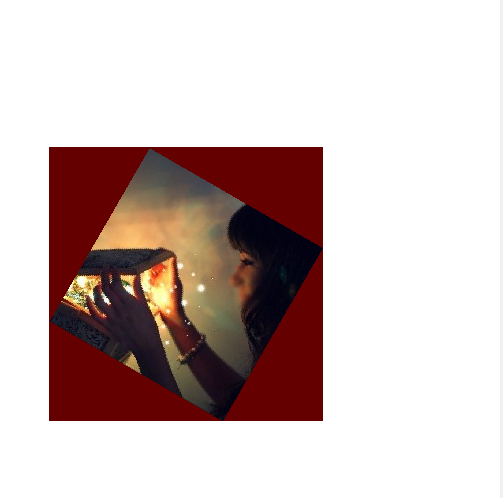
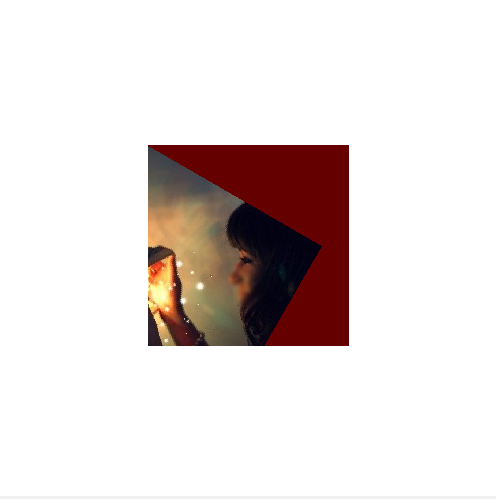
* 1. 选定变换方法进行变换

通过GUI的popupmenu选定变换方法，且选定后显示对应个数的输入框，例如，平移变换需要两个输入框（x轴、y轴上的平移量）；旋转变换需要一个输入框（旋转角度）；颜色变换不需要输入框。

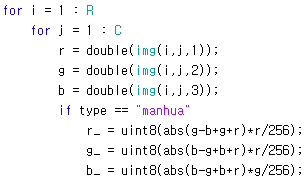
 

根据变换方法对应的矩阵运算，定义变换矩阵，对原始图像的每个像素乘以变换矩阵，实现各种变换。需要注意，若平移量、旋转角等变换参数为double类型，则做矩阵乘法后其元素类型会变成double类型。而矩阵乘法的结果中u和v表示变换后图像中像素的位置，应为整数，因此需要进行强制的类型转换，转换成uint8类型或uint16类型（若变换后图像的宽和长比255大，即超出uint8类型的范围，则需要用uint16类型）。

需要特别注意，旋转变换和错切变换需要注意其变换后的大小，否则变换后的图像会不完整。以旋转变换为例，以下两个图分别为对原始图像进行30°的旋转变换时不考虑大小变化和考虑大小变化的变换结果。可见，不考虑图像大小变化时，部分像素不能得到旋转变换而结果不完整。假设原始图像的行数和列数分别是和，则旋转后，其行数变为，列数变为。



颜色变换中也需要注意类型。从原始图像中获取像素值时应先将其值转换成double类型，这样才能保证以后做像素变换时不丢失信息，否则变换结果会成为黑白图。另外，变换后需要将double类型的像素值转换成uint8类型。

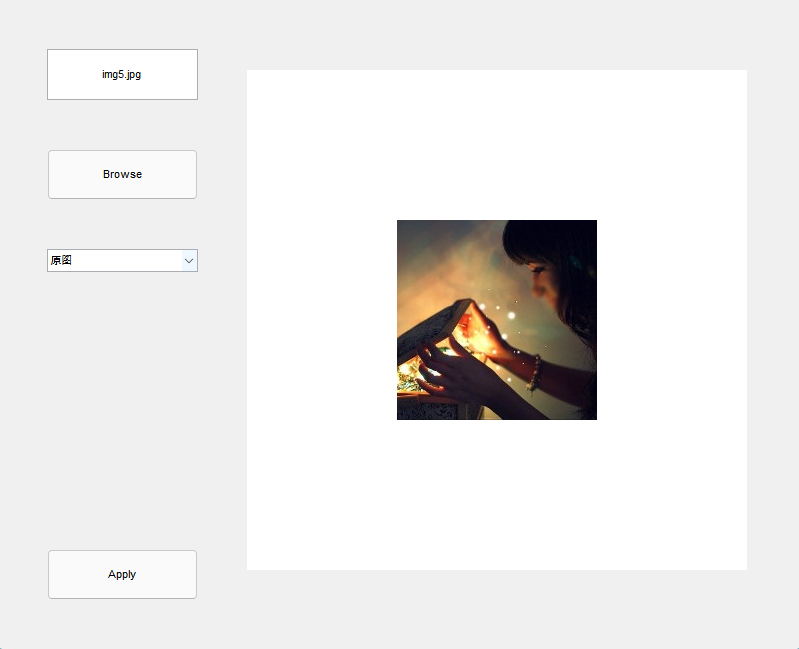


* 1. 输出变换后的图像

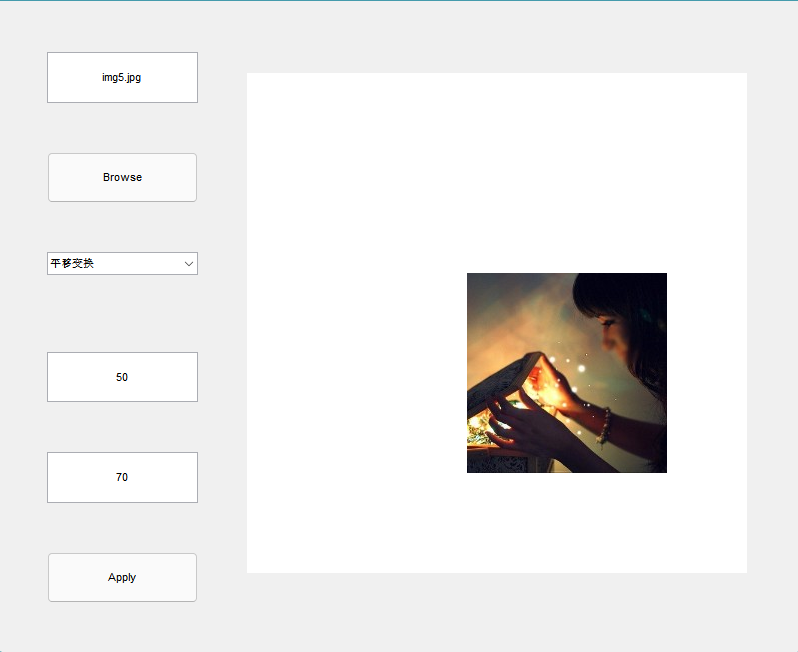
通过MATLAB中imshow函数输出原始图像以及变换后的图像。变换后图像的大小会发生变化，为了避免变换后的图像发生部分截断，应在图像周围扩充白色。其实现方法可采用图像平移变换，将图像平移到白图的中心处。输出图像时也需要保证图像的类型为uint8类型，若图像为double类型的三维矩阵，则像素值大于1的部分都显示成白色。

****

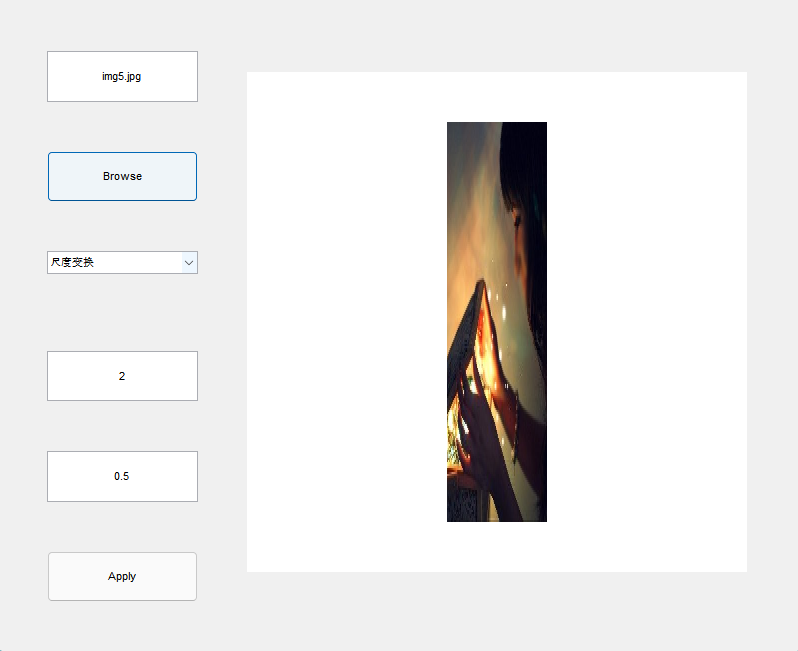
1. **实验结果和分析**
   1. 原始图像

****

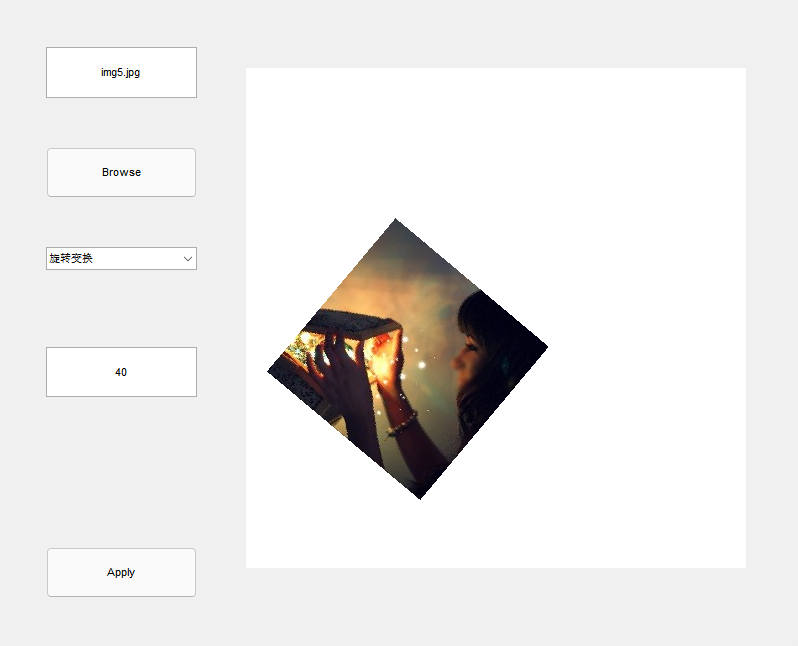
* 1. 平移变换：(50,70)

****

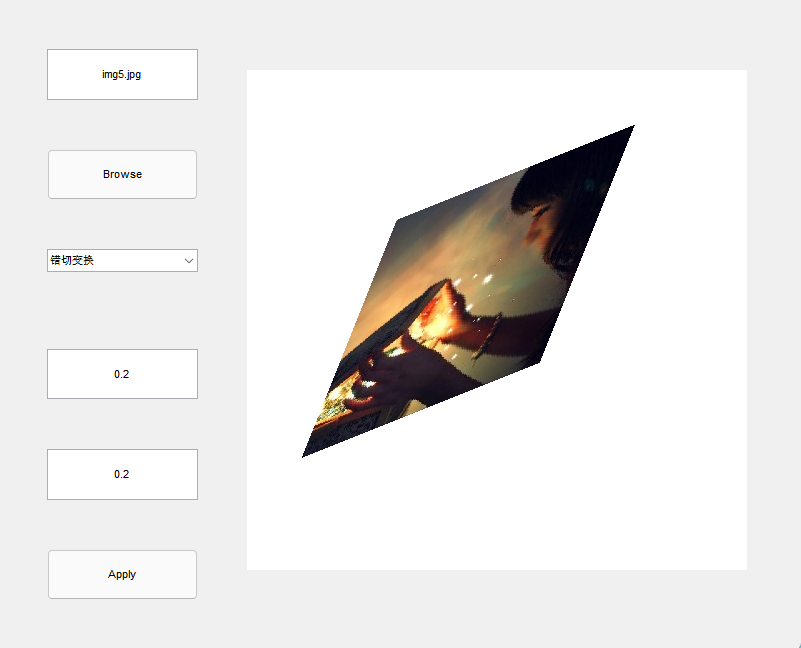
* 1. 尺度变换：(2,0.5)

****

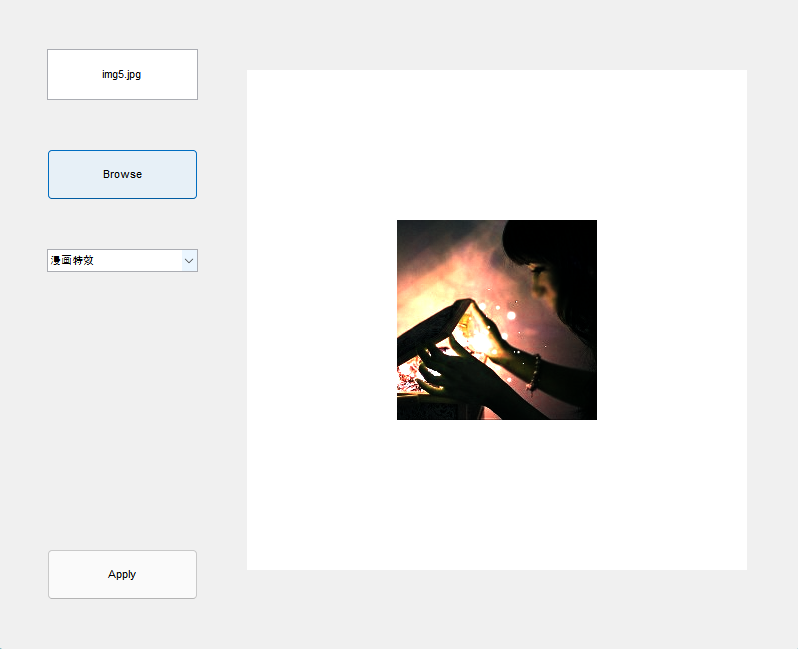
* 1. 旋转变换：30°

****

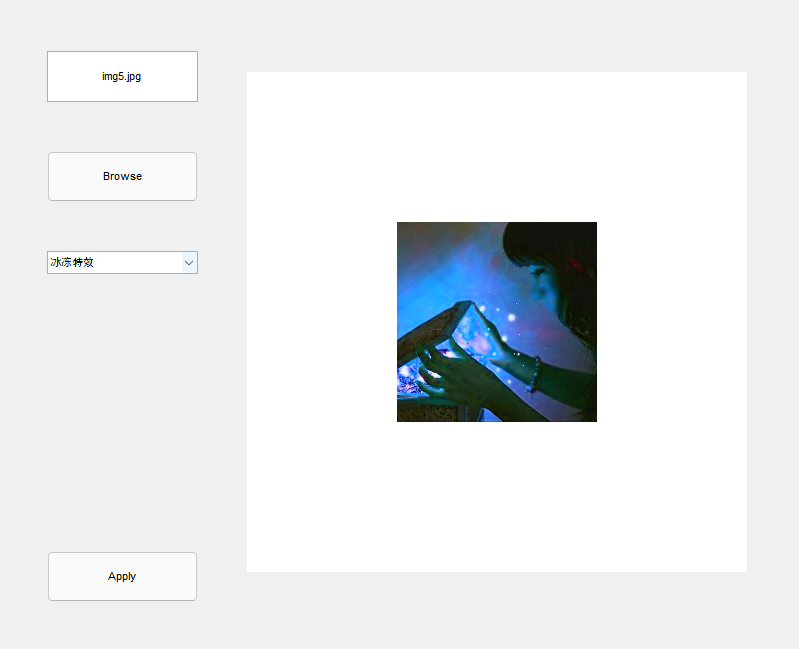
* 1. 错切变换：(0.2,0.2)

****

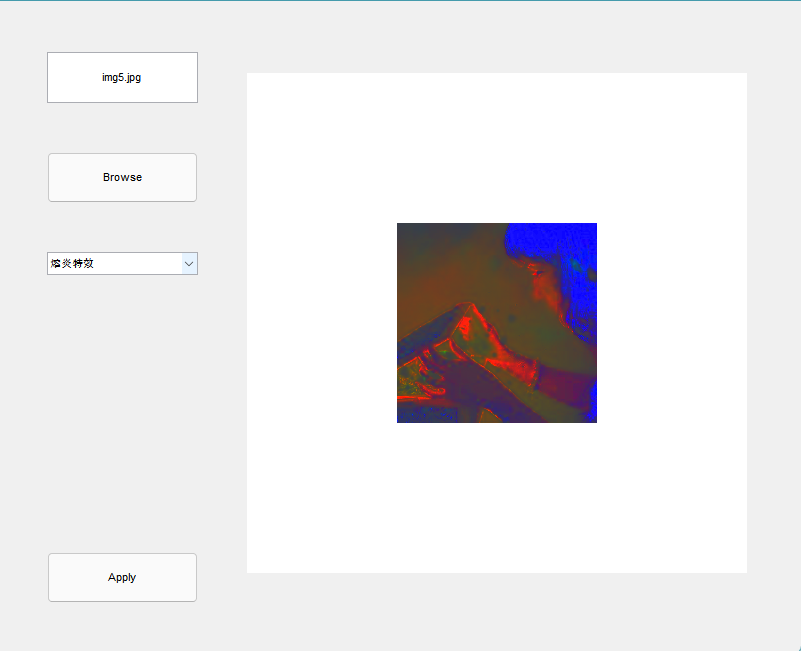
* 1. 漫画特效

****

* 1. 冰冻特效

****

* 1. 熔炎特效

****

1. **讨论**

在图像处理中，常常出翔图像显示成黑白图的情况，这是数据类型转换时出错导致的。因此，图像变换的过程中需注意像素的数据类型。

* 1. double和uint8类型

MATLAB中double是默认的数据类型，允许双精度浮点数据和负数；uint8是无符号整形数据，取值范围为0到，即0到255。

* 1. im2uint8函数

对于double类型数据，im2uint8将其0到1部分重新映射到 0到255 之间，然后对于溢出的部分，就近处理为0或255。

* 1. im2double函数

对于uint8 类型数据，im2double将0到255部分重新映射到0到1之间，这里uint8数据本身有溢出限制，因此不存在小于0或大于1的部分。

* 1. imread函数

图像文件经过imread函数读取后，获取到的图像数据类型为uint8。

* 1. imshow函数

对于uint8数据，imshow 直接按照0到255将数值映射到灰度级上，0对应纯黑，255 对应纯白，中间为渐变灰。对于double数据，imshow仅保留其0到1之间的数值，并映射到灰度级上，0对应纯黑，1对应纯白，中间为渐变灰；对于溢出的部分，就近处理到0或1，显示为纯黑或纯白。

假设用imread函数读取图像后得到的矩阵A 为uint8类型，而用zeros函数得到的矩阵B的数据类型为double，则直接将A的像素值付给B后用imshow函数显示的图像会出现纯黑或纯白的情况。

1. **代码**

function varargout = lab4(varargin)

% Begin initialization code - DO NOT EDIT

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @lab4\_OpeningFcn, ...

'gui\_OutputFcn', @lab4\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

function lab4\_OpeningFcn(hObject, eventdata, handles, varargin)

% Choose default command line output for lab4

handles.output = hObject;

% Update handles structure

guidata(hObject, handles);

set(handles.param1\_text,'Visible','off')

set(handles.param2\_text,'Visible','off')

global res;

init\_bg()

imshow(res);

function varargout = lab4\_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function menu\_Callback(hObject, eventdata, handles)

m = get(hObject,'Value');

if m == 2

set(handles.param1\_text,'Visible','on')

set(handles.param2\_text,'Visible','on')

set(handles.param1\_text,'String','0');

set(handles.param2\_text,'String','0');

elseif m == 3

set(handles.param1\_text,'Visible','on')

set(handles.param2\_text,'Visible','on')

set(handles.param1\_text,'String','1');

set(handles.param2\_text,'String','1');

elseif m == 5

set(handles.param1\_text,'Visible','on')

set(handles.param2\_text,'Visible','on')

set(handles.param1\_text,'String','1');

set(handles.param2\_text,'String','1');

elseif m == 4

set(handles.param1\_text,'Visible','on')

set(handles.param2\_text,'Visible','off')

set(handles.param1\_text,'String','0');

else

set(handles.param1\_text,'Visible','off')

set(handles.param2\_text,'Visible','off')

end

global img;

if ~isempty(img)

init\_res()

end

function menu\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function param1\_text\_Callback(hObject, eventdata, handles)

function param1\_text\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function param2\_text\_Callback(hObject, eventdata, handles)

function param2\_text\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function applyButton\_Callback(hObject, eventdata, handles)

global res;

global angle;

global shx;

global shy;

m = get(handles.menu,'Value');

x = str2double(get(handles.param1\_text,'String'));

y = str2double(get(handles.param2\_text,'String'));

init\_bg()

if m == 2

x = round(x);

y = round(y);

move\_img(x,y)

elseif m == 3

times\_img(x,y)

elseif m == 4

angle = angle + x;

rotate\_img(angle);

elseif m == 5

shx = x \* (1+shx);

shy = y \* (1+shy);

sh\_img(shx,shy);

elseif m == 6

effect("manhua");

elseif m == 7

effect("bingdong");

elseif m == 8

effect("rongyan");

end

imshow((res))

function file\_text\_Callback(hObject, eventdata, handles)

function file\_text\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function browseButton\_Callback(hObject, eventdata, handles)

global img;

global R;

global C;

global ori\_img;

global ori\_R;

global ori\_C;

[file,path] = uigetfile('\*.jpg');

str = path + "/" + file;

img = imread(str);

[R, C,~] = size(img);

ori\_img = img;

ori\_R = R;

ori\_C = C;

init\_res();

set(handles.file\_text,'String',file);

function init\_res()

global img;

global R;

global C;

global res;

global p;

global q;

global ori\_R;

global ori\_C;

global ori\_img;

global angle;

global shx;

global shy;

R = ori\_R;

C = ori\_C;

p = 250-round(R/2);

q = 250-round(C/2);

img = ori\_img;

angle = 0;

shx = 1;

shy = 1;

init\_bg();

move\_img(0,0);

imshow((res));

function init\_bg()

global res;

res = uint8(zeros(500, 500, 3));

for i = 1:500

for j = 1:500

res(i,j,1:3)=[255,255,255];

end

end

function move\_img(x,y)

global res;

global img;

global R;

global C;

global p;

global q;

p = p+x;

q = q+y;

tras = [1 0 p; 0 1 q; 0 0 1];

for i = 1 : R

for j = 1 : C

temp = [i; j; 1];

temp = tras \* temp;

a = temp(1, 1);

b = temp(2, 1);

if (a <= 500) && (b <= 500) && (a >= 1) && (b >= 1)

res(a,b,:) = img(i,j,:);

end

end

end

function times\_img(sx,sy)

global p;

global q;

global img;

global R;

global C;

global res;

tras = [1/sx 0 0; 0 1/sy 0; 0 0 1];

r = round(sx \* R);

c = round(sy \* C);

tmp = zeros(r,c,3);

p = round(p-(R\*(sx-1))/2);

q = round(q-(C\*(sy-1))/2);

for i = 1 : r

for j = 1 : c

temp = [i; j; 1];

temp = tras \* temp;

x = uint16(temp(1, 1));

y = uint16(temp(2, 1));

a = p+i;

b = q+j;

if (x <= R) && (y <= C) && (x >= 1) && (y >= 1)

tmp(i,j,:) = img(x,y,:);

if (a <= 500) && (b <= 500) && (a >= 1) && (b >= 1)

res(a,b,:) = img(x,y,:);

end

else

tmp(i,j,:) = [255, 255, 255];

if (a <= 500) && (b <= 500) && (a >= 1) && (b >= 1)

res(a,b,:) = [255, 255, 255];

end

end

end

end

img = tmp;

R = r;

C = c;

function rotate\_img(d)

global res;

global img;

global ori\_img;

global ori\_R;

global ori\_C;

global R;

global C;

global p;

global q;

alpha = d \* 3.1415926 / 180.0;

c1 = round(-R\*sin(alpha));

c2 = round(C\*cos(alpha));

r1 = round(C\*sin(alpha));

r2 = round(R\*cos(alpha));

if cos(alpha)\*sin(alpha) >= 0

if c1 > c2

j1 = c2;

j2 = c1;

else

j1 = c1;

j2 = c2;

end

if r1+r2 > 0

i1 = 0;

i2 = r1+r2;

else

i1 = r1+r2;

i2 = 0;

end

else

if c1+c2 < 0

j1 = c1+c2;

j2 = 0;

else

j1 = 0;

j2 = c1+c2;

end

if r1 > r2

i1 = r2;

i2 = r1;

else

i1 = r1;

i2 = r2;

end

end

a = abs(i1)+abs(i2);

b = abs(j1)+abs(j2);

tmp = zeros(a,b,3);

tras = [cos(alpha) -sin(alpha) 0; sin(alpha) cos(alpha) 0; 0 0 1];

for i = i1 : i2

for j = j1 : j2

temp = [i; j; 1];

temp = tras \* temp;

x = uint16(temp(1, 1));

y = uint16(temp(2, 1));

if (x <= ori\_R) && (y <= ori\_C) && (x >= 1) && (y >= 1)

if (p+i <= 500) && (q+j <= 500) && (p+i >= 1) && (q+j >= 1)

res(p+i,q+j,:) = ori\_img(x,y,:);

end

else

if (p+i <= 500) && (q+j <= 500) && (p+i >= 1) && (q+j >= 1)

res(p+i,q+j,:) = [256,256,256];

end

end

end

end

img = tmp;

R = a;

C = b;

function sh\_img(shx,shy)

global R;

global C;

global res;

global img;

global p;

global q;

sh = [1 shy 0; shx 1 0; 0 0 1]';

for i = -round(R + shx\*C) : 1 : 2\*R + shx \* C

for j = -round(C + shy \* R) : 1 : 2\*C + shy \* R - 20

temp = [i; j; 1];

temp = sh \* temp;

x = uint16(temp(1, 1));

y = uint16(temp(2, 1));

if (x <= R) && (y <= C) && (x >= 1) && (y >= 1)

if (p+i <= 500) && (q+j <= 500) && (p+i >= 1) && (q+j >= 1)

res(p+i, q+j,:) = img(x, y,:);

end

end

end

end

function effect(type)

global img;

global res;

global R;

global C;

global p;

global q;

for i = 1 : R

for j = 1 : C

r = double(img(i,j,1));

g = double(img(i,j,2));

b = double(img(i,j,3));

if type == "manhua"

r\_ = uint8(abs(g-b+g+r)\*r/256);

g\_ = uint8(abs(b-g+b+r)\*r/256);

b\_ = uint8(abs(b-g+b+r)\*g/256);

elseif type == "bingdong"

r\_ = uint8(abs(r-g-b));

g\_ = uint8(abs(g-b-r));

b\_ = uint8(abs(b-r-g));

elseif type == "rongyan"

r\_ = uint8(r\*128/(g+b+1));

g\_ = uint8(g\*128/(b+r+1));

b\_ = uint8(b\*128/(r+g+1));

end

if r\_ < 0

r\_ = 0;

elseif r\_ > 255

r\_ = 255;

end

if g\_ < 0

g\_ = 0;

elseif g\_ > 255

g\_ = 255;

end

if b\_ < 0

b\_ = 0;

elseif b\_ > 255

b\_ = 255;

end

img(i,j,:) = [r\_ g\_ b\_];

res(p+i,q+j,:) = [r\_ g\_ b\_];

end

end