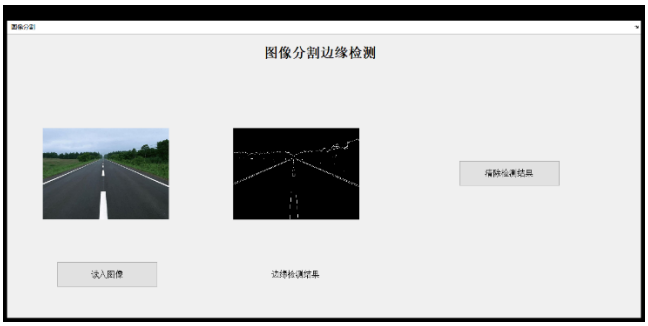
1、编程实现基于边缘检测和 Hough 变换的车道线检测算法，边缘检测利用梯度算子、拉普拉斯算子、LOG 算子、DOG 算子、Canny 算子等进行检，在此基础上利用 Hough 变换进行车道线的检测，并比较分析各种边缘检测算子的优缺点。要求创建用户交互界面，能够实现图像读入、边缘检测、车道检测、结果显示等功能。（车道图像可在微助教下载，也可直接采集拍照车道图像）

（1） 各种边缘检测算子的优缺点结果分析

1) Roberts 算子利用局部差分算子寻找边缘，边缘定位精度较高，但容易丢失一部分边缘，不具备抑制噪声的能力；
2) Prewitt 算子对灰度渐变的图像边缘提取效果较好，而没有考虑相邻点的距离远近对当前像素点的影响；
3) Sobel 算子考虑了综合因素，对噪声较多的图像处理效果更好。
4) Laplacian 算子对噪声非常敏感，它使噪声成分得到加强，这两个特性使得该算子容易丢失一部分边缘的方向信息，造成一些不连续的检测边缘，同时抗噪声能力比较差，其算法可能会出现双像素边界，常用来判断边缘像素位于图像的明区或暗区；
5) LOG 算子是对 laplacian 算子的改进，即可平滑图像，又可降低噪声，提升边缘检测准确度；
6) DOG 算子作为 LOG 算子的进一步简化，3 结果图可以看出，可以提取更多的细节与特点；
7) Canny 算子作为目前理论上相对最完善的一种边缘检测算法，对边缘检测的细节提取的更精确，为了得到较好的边缘检测结果，它通常使用较大的滤波尺度，这样容易丢失一些细节。
8) 进行边缘检测后，在边缘检测的基础上进行 Hough 变换，由结果图可知，Prewitt 算子在提取车道信息时，提取的直线更多，而拉普拉斯算子提取的直线只有两条，效果不佳。

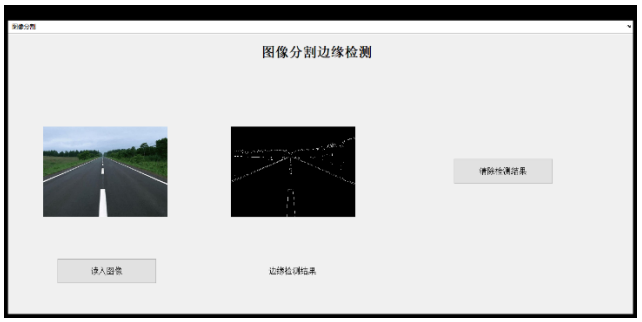
（2） 结果图

1) Roberts



2） Prewitt

3） Sobel



4） 拉普拉斯

5） Log

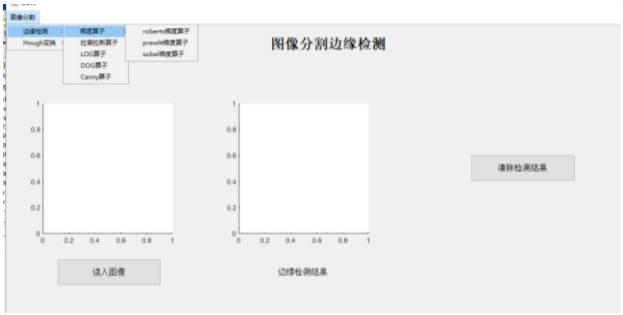
6） Dog

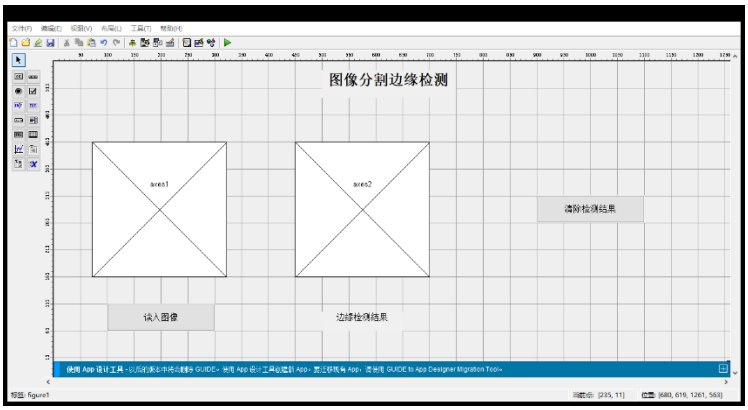
7） Canny

8） Prewitt-Hough

9） laplician-Hough

（3） 交互界面





（4） 编程代码

```
function varargout = GUI3(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUI3_OpeningFcn, ...
                   'gui_OutputFcn',  @GUI3_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 GUI3_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
 guidata(hObject, handles);
function varargout = GUI3_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
```

```
%导入图片
function pushbutton1_Callback(hObject, eventdata, handles)
[filename,pathname,filterindex]=...
uigetfile({'*.*';'*.bmp';'*.tif';'*.png';'*.jpg';'*.jpeg'},'select picture');
str=[pathname filename];
s=str;
handles.filebig=filterindex;
if filterindex==0
return
else
im1=imread(str);
end
axes(handles.axes1);
imshow(im1);
handles.img=im1;
guidata(hObject,handles);

function Untitled_1_Callback(hObject, eventdata, handles)
function Untitled_3_Callback(hObject, eventdata, handles)

function Untitled_4_Callback(hObject, eventdata, handles)

%laplacian
function Untitled_5_Callback(hObject, eventdata, handles)
function Untitled_6_Callback(hObject, eventdata, handles)
global BW4
I = getimage(gca);
I=im2double(I);
[M,N]=size(I);
BW1=zeros(size(I));
for x=2:M-1
    for y=2:N-1
        BW1(x,y)=I(x+1,y)+I(x-1,y)+I(x,y+1)+I(x,y-1)-4*I(x,y);
    end
end
BW1=im2uint8(BW1);
axes(handles.axes2);
imshow(BW1);
handles.img=BW1;
guidata(hObject,handles);

%log
function Untitled_7_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
BW2 = edge(I,'log');
axes(handles.axes2);
imshow(BW2);
handles.img=BW2;
guidata(hObject,handles);


%dog
function Untitled_9_Callback(hObject, eventdata, handles)
I = double(getimage(gca))/255;
gray=rgb2gray(I);
sigma1=0.1;
sigma2=0.8;
window=7;
H1=fspecial('gaussian', window, sigma1);
H2=fspecial('gaussian', window, sigma2);
DiffGauss=H1-H2;
BW3=imfilter(gray,DiffGauss,'replicate');
BW3=mat2gray(BW3);
axes(handles.axes2);
imshow(BW3);
handles.img=BW3;
```

```
guidata(hObject,handles);

%canny
function Untitled_8_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
BW4 = edge(I,'canny');
axes(handles.axes2);
imshow(BW4);
handles.img=BW4;
guidata(hObject,handles);

%清除界面
function pushbutton2_Callback(hObject, eventdata, handles)
axes(handles.axes1);
cla reset;
axes(handles.axes2);
cla reset;

%roberts
function Untitled_10_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
BW5 = edge(I,'roberts');
axes(handles.axes2);
imshow(BW5);
handles.img=BW5;
guidata(hObject,handles);

%Sobel
function Untitled_11_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
BW6 = edge(I,'%Sobel
');
axes(handles.axes2);
imshow(BW6);
handles.img=BW6;
guidata(hObject,handles);

%Sobel
function Untitled_12_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
BW7 = edge(I,'sobel');
axes(handles.axes2);
imshow(BW7);
handles.img=BW7;
guidata(hObject,handles);

% P-Hough--------------------------------------------------------------------------
function Untitled_13_Callback(hObject, eventdata, handles)
I = getimage(gca);
I=rgb2gray(I);
imshow(I);
axis on;
BW=edge(I,'prewitt');
axis on;
[H,T,R]=hough(BW);
xlabel('\theta'),ylabel('\rho');
axis on , axis normal, hold on;
P=houghpeaks(H,5,'threshold',ceil(0.3*max(H(:))));
x=T(P(:,2));y=R(P(:,1));
plot(x,y,'s','color','white');
lines=houghlines(BW,T,R,P,'FillGap',5,'MinLength',7);
axes(handles.axes2);
imshow(I);
handles.img=I;
```

```
guidata(hObject,handles);
title('»ô·ò±ä»»Í¼ÏñÌ¼ì²â');
axis on;
hold on;
max_len=0;
for k=1:length(lines)
xy=[lines(k).point1;lines(k).point2];
plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','yellow');
plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','red');
len=norm(lines(k).point1-lines(k).point2);
if(len>max_len)
max_len=len;
xy_long=xy;
end
end
plot(xy_long(:,1),xy_long(:,2),'LineWidth',2,'Color','cyan');

%L-Hough ---------------------------------------------------------------------
function Untitled_14_Callback(hObject, eventdata, handles)
a = getimage(gca);
a=rgb2gray(a);
bw1=LapuLas(a);
[H,theta,rho]=naiveHough(bw1);
P = houghpeaks(H,6);
lines = houghlines(bw1,theta,rho,P);
axes(handles.axes2);
imshow(a);
handles.img=a;
guidata(hObject,handles);
hold on;
for k = 1:length(lines)
xy = [lines(k).point1; lines(k).point2];
plot(xy(:,1),xy(:,2),'LineWidth',1,'Color','green');%»-³öÏß¶Î
plot(xy(1,1),xy(1,2),'o','LineWidth',1,'Color','yellow');%Æðµã
plot(xy(2,1),xy(2,2),'o','LineWidth',1,'Color','red');%ÖÕµã
end
function [ Hough, theta_range, rho_range ] = naiveHough(I)
[rows, cols] = size(I);
theta_maximum = 90;
rho_maximum = floor(sqrt(rows^2 + cols^2)) - 1;
theta_range = -theta_maximum:theta_maximum - 1;
rho_range = -rho_maximum:rho_maximum;
Hough = zeros(length(rho_range), length(theta_range));
for row = 1:rows
    for col = 1:cols
        if I(row, col) > 0
            x = col - 1;
            y = row - 1;
            for theta = theta_range
                rho = round((x * cosd(theta)) + (y * sind(theta)));
                rho_index = rho + rho_maximum + 1;
                theta_index = theta + theta_maximum + 1;
                Hough(rho_index, theta_index) = Hough(rho_index, theta_index) + 1;
            end
        end
    end
end
function [p] = LapuLas(e)
r=e;
[m,n]=size(e);
dd=sum(sum(e)/(m*n));
for x=1:m
    for y=1:n
        if(r(x,y)>=dd)
            r(x,y)=255;
        else
            r(x,y)=0;
```

```
        end
    end
end
p=zeros(m-1,n-1);
for ii=2:m-1
    for jj=2:n-1
        p(ii,jj)=r(ii,jj+1)+r(ii,jj-1)+r(ii+1,jj)+r(ii-1,jj)-4*(r(ii,jj));
    end
end
p=uint8(p);
```