****

**机器视觉 实验报告**

**专业： 自动化**

**班级： 2103206 班**

**姓名：**

**学号：**

**实验日期： 2024 年 4 月**

实验与创新实践教育中心

Education Center of Experiments and Innovations

# 实验1

选择工具栏“文件”——“导出”，选择“C++”，即可生成 C++代码，并将其粘贴或导入到创建的VS工程中。导出的C++代码很长，截取主体action()函数如下：

（略）

可以看出，该函数依次进行了以下操作：（1）定义局部变量；（2）打开相机（OpenFramegrabber）；（3）开始采集图像（GrabImageStart）；（4）设置窗口背景色（SetWindowAttr）；（5）打开窗口（OpenWindow）；（6）死循环中采集图像并且将其在窗口中显示（GrabImageAsync、DispObj）；（7）关闭相机（CloseFramegrabber）。

编译、运行 VS 程序，就会不断弹窗显示采集到的图片。

# 实验2

打开Halcon软件，首先获取模板。利用相机采集待用图像10张。代码如下：

\* 由Halcon的Image Acquisition 生成打开相机及图像采集的代码，勿照抄

open\_framegrabber ('GigEVision2', 0, 0, 0, 0, 0, 0, 'progressive', -1, 'default', -1, 'false', 'default', '003053313bc6\_Basler\_acA130060gm', 0, -1, AcqHandle)

for i:=1 to 10 by 1

\* 抓取图像

grab\_image\_async (Image, AcqHandle, -1)

\* 保存图像

（略，请自行补全）

endfor

close\_framegrabber (AcqHandle)

从刚才获取的10张图像中挑选一张质量较高的，保存并重命名，然后，新开页面并键入以下代码：（其中\* 为注释行，通过注释行即可看出模板匹配原理）

总体功能介绍：（1）显示模板图片及用于生成模板的区域；（2）实时拍摄并检测图片（每隔2s拍摄一次，共拍摄11次，检测前10次），如检测到目标对象则框出，并在右侧新开一黑色背景窗口显示匹配到的位置在原图像上的位置（行、列）以及匹配分数；若没有检测到，则不会框出，也不会显示出匹配分数。

\*关闭窗口更新

dev\_update\_off ()

\*读取单张图片

read\_image (…)

\*返回输入图像的大小

get\_image\_size (…)

\*关闭激活的活动图形显示窗口

dev\_close\_window ()

\*打开一个新的图形窗口

dev\_open\_window (…)

\*设置当前窗口的字体属性

set\_display\_font (…)

\*设置一个或多个（窗口）输出颜色

dev\_set\_color (…)

\*定义region填充模式，只显示轮廓

dev\_set\_draw (…)

\*创建直立矩形，得到的Rectangle用于绘制 ROI

……

\*用于缩小图像域，确定ROI

reduce\_domain (Image, Rectangle, ImageReduced)

\*创建NCC模板，做匹配使用，角度选择全部角度（0-360），灰度值完全相反的图像也接受

create\_ncc\_model (…)

\*在当前图形窗口中显示图像及ROI

dev\_display (Image)

dev\_display (Rectangle)

\*调试时使用

stop ()

\*打开并配置所选的图像采集设备

open\_framegrabber (…)

\*采集11张图片并依次作匹配

for i := 1 to 11 by 1

grab\_image (…)

find\_ncc\_model (…)

dev\_display (Image)

\*显示NCC匹配结果图形

dev\_display\_ncc\_matching\_results (…)

\*显示匹配结果，包含xy位置、分数等

……

wait\_seconds(2)

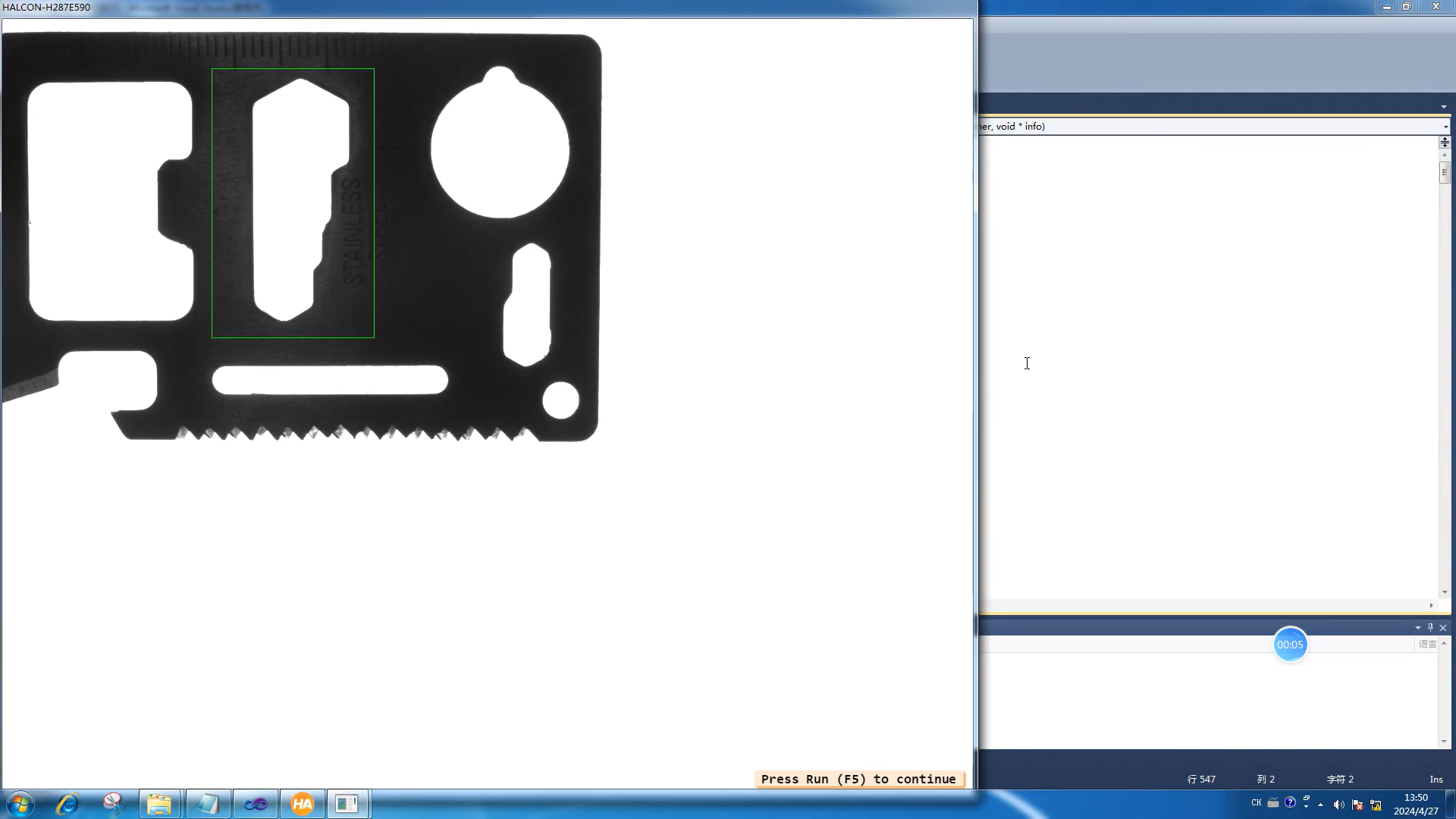
dev\_close\_window ()

endfor

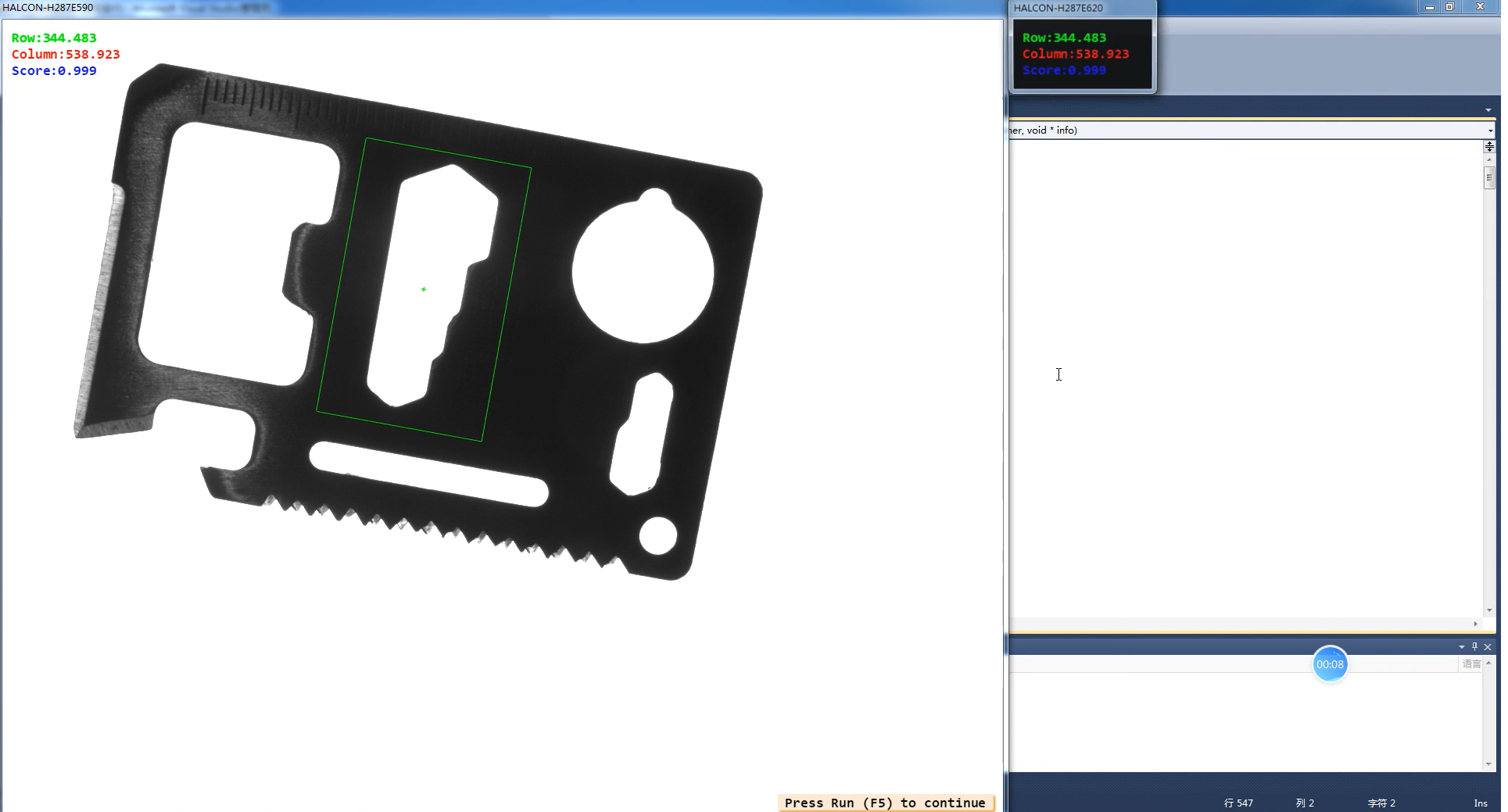
close\_framegrabber(AcqHandle)

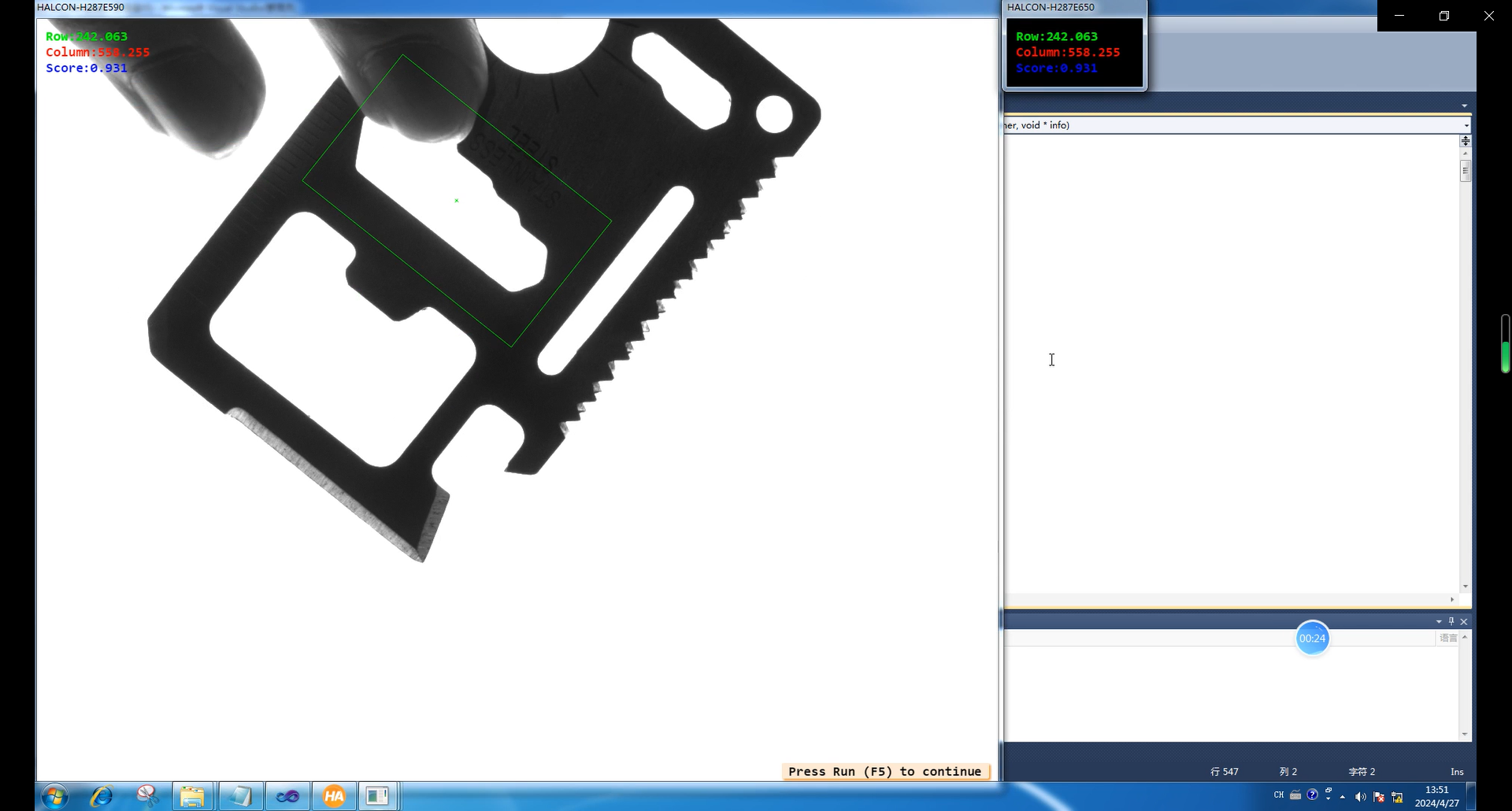
接着，再用Halcon将其导出为C++代码，导入至 Visual Studio 2010 生成的工程文件，并运行，效果如下（截图）：

1. 显示模板图片及用于生成模板的区域：



1. 实时拍摄并检测图片：





可见，如果遮住小部分边缘，检测仍可正常进行。

上面整个检测流程有录制视频，可通过此链接查看：

<https://pan.baidu.com/s/1RQNigv-ZHJxzN_9f_YGlRg?pwd=0430>

# 实验3

## 任务1 Halcon代码

\* 以下代码大多由 HALCON 生成，不用抄

\* Measure 01: Code generated by Measure 01

\* Measure 01: Initialize acquisition

dev\_close\_window ()

\*打开并配置所选的图像采集设备

open\_framegrabber (…)

\* Measure 01: Prepare measurement

AmplitudeThreshold := 9

RoiWidthLen2 := 19.5

set\_system ('int\_zooming', 'true')

\* Measure 01: Coordinates for line Measure 01 [0]

LineRowStart\_Measure\_01\_0 := 169.063

LineColumnStart\_Measure\_01\_0 := 785.501

LineRowEnd\_Measure\_01\_0 := 263.063

LineColumnEnd\_Measure\_01\_0 := 783.501

\* Measure 01: Convert coordinates to rectangle2 type

TmpCtrl\_Row := 0.5\*(LineRowStart\_Measure\_01\_0+LineRowEnd\_Measure\_01\_0)

TmpCtrl\_Column := 0.5\*(LineColumnStart\_Measure\_01\_0+LineColumnEnd\_Measure\_01\_0)

TmpCtrl\_Dr := LineRowStart\_Measure\_01\_0-LineRowEnd\_Measure\_01\_0

TmpCtrl\_Dc := LineColumnEnd\_Measure\_01\_0-LineColumnStart\_Measure\_01\_0

TmpCtrl\_Phi := atan2(TmpCtrl\_Dr, TmpCtrl\_Dc)

TmpCtrl\_Len1 := 0.5\*sqrt(TmpCtrl\_Dr\*TmpCtrl\_Dr + TmpCtrl\_Dc\*TmpCtrl\_Dc)

TmpCtrl\_Len2 := RoiWidthLen2

\* Measure 01: Create measure for line Measure 01 [0]

\* Measure 01: Attention: This assumes all images have the same size!

gen\_measure\_rectangle2 (TmpCtrl\_Row, TmpCtrl\_Column, TmpCtrl\_Phi, TmpCtrl\_Len1, TmpCtrl\_Len2, 1280, 1024, 'bilinear', MsrHandle\_Measure\_01\_0)

grab\_image (Image, AcqHandle)

get\_image\_size (Image, Width, Height)

dev\_open\_window (0, 0, Width/2 , Height/2 , 'light gray', WindowID)

dev\_display(Image)

close\_framegrabber (…)

measure\_pairs (Image, MsrHandle\_Measure\_01\_0, 5.1, AmplitudeThreshold, 'all', 'all', Row1\_Measure\_01\_0, Column1\_Measure\_01\_0, Amplitude1\_Measure\_01\_0, Row2\_Measure\_01\_0, Column2\_Measure\_01\_0, Amplitude2\_Measure\_01\_0, Width\_Measure\_01\_0, Distance\_Measure\_01\_0)

close\_measure (MsrHandle\_Measure\_01\_0)

dev\_set\_draw('margin')

dev\_set\_color('green')

\*生成矩形 ROI

gen\_rectangle2(ROI,TmpCtrl\_Row,TmpCtrl\_Column,TmpCtrl\_Phi,TmpCtrl\_Len1,TmpCtrl\_Len2)

\*显示矩形

dev\_display(ROI)

\*根据提取的边缘点生成 XLD 轮廓显示

gen\_contour\_polygon\_xld(EdgeFirst,[-sin(TmpCtrl\_Phi+rad(90))\*TmpCtrl\_Len2+Row1\_Measure\_01\_0[0],\

-sin(TmpCtrl\_Phi-rad(90))\*TmpCtrl\_Len2+Row1\_Measure\_01\_0[0]],\

[cos(TmpCtrl\_Phi+rad(90))\*TmpCtrl\_Len2+Column1\_Measure\_01\_0[0],\

cos(TmpCtrl\_Phi-rad(90))\*TmpCtrl\_Len2+Column1\_Measure\_01\_0[0]])

gen\_contour\_polygon\_xld(Edgesecond,[-sin(TmpCtrl\_Phi+rad(90))\*TmpCtrl\_Len2+Row2\_Measure\_01\_0[0],\

-sin(TmpCtrl\_Phi-rad(90))\*TmpCtrl\_Len2+Row2\_Measure\_01\_0[0]],\

[cos(TmpCtrl\_Phi+rad(90))\*TmpCtrl\_Len2+Column2\_Measure\_01\_0[0],\

cos(TmpCtrl\_Phi-rad(90))\*TmpCtrl\_Len2+Column2\_Measure\_01\_0[0]])

\*用不同颜色显示边缘

dev\_set\_line\_width (3)

dev\_set\_color('blue')

dev\_display(EdgeFirst)

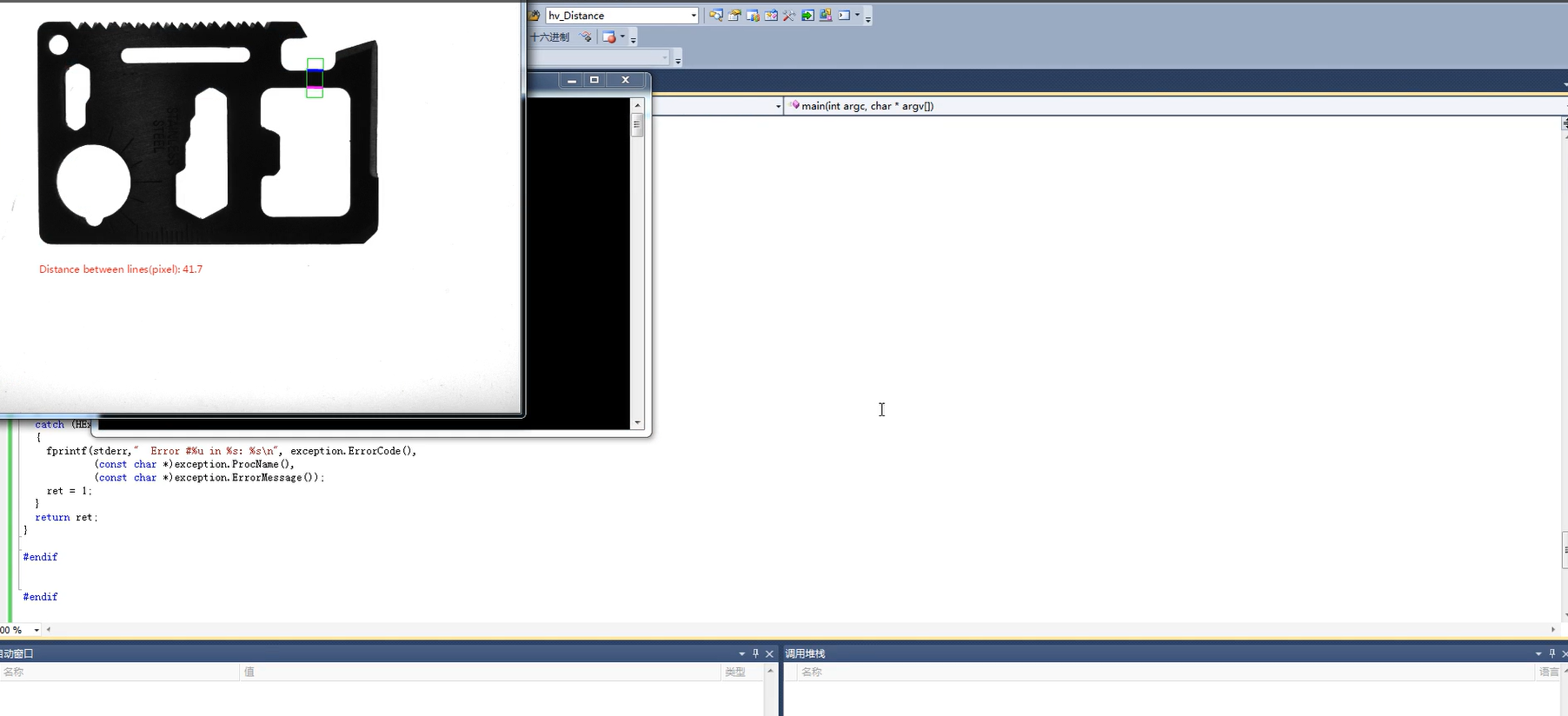
dev\_set\_color('magenta')

dev\_display(Edgesecond)

disp\_message (WindowID, 'Distance between lines(pixel): ' +Width\_Measure\_01\_0$'.4', 'window', 330, 60, 'red', 'false')

wait\_seconds (6)

## 任务1（1D测量）效果



效果视频链接：https://pan.baidu.com/s/1Mw2zRnMKI1HJDMihoAxmkQ?pwd=0430

## 任务2 Halcon代码

dev\_update\_off ()

\*打开并配置所选的图像采集设备

open\_framegrabber ('GigEVision2', 0, 0, 0, 0, 0, 0, 'progressive', -1, 'default', -1, 'false', 'default', '003053313bc6\_Basler\_acA130060gm', 0, -1, AcqHandle)

\*采集单张图像

grab\_image\_async (Image, AcqHandle, -1)

close\_framegrabber (AcqHandle)

get\_image\_size (…)

dev\_close\_window ()

\* 显示部分略…

\*阈值分割，提示：利用fast\_threshold和connection，略去

\*边缘检测、提取边缘，提示如下

boundary (…)

\* 将边缘做膨胀

dilation\_circle (…)

\* 将膨胀后的边缘连在一起

union1 (…)

\* 以上述联合边缘裁切图像的ROI（感兴趣区域只有边缘）

reduce\_domain (…)

dev\_clear\_window ()

dev\_display (ImageReduced)

CenterRow := []

CenterColumn := []

wait\_seconds(1)

\*亚像素精度边缘提取

edges\_sub\_pix (…)

\*按照圆形度选出形状轮廓，详见指导书

select\_shape\_xld(…)

\*计算符合要求的轮廓数量

count\_obj (selectedXLD1, NumberCircles)

for i := 1 to NumberCircles by 1

select\_obj (selectedXLD1, ObjectSelected, i)

\*匹配圆形轮廓

fit\_circle\_contour\_xld (…)

\*生成圆形轮廓并绘制轮廓及圆心到轮廓的箭头（后者代码提供给大家参考）

gen\_circle\_contour\_xld (…)

dev\_set\_color ('red')

dev\_display (ContCircle)

CenterRow := [CenterRow,Row]

CenterColumn := [CenterColumn,Column]

Row2 := Row + Radius \* sin(rad(-45))

Column2 := Column + Radius \* cos(rad(-45))

set\_tposition (WindowID, Row2 - 35, Column2 + 5)

disp\_arrow (WindowID, Row, Column, Row2, Column2, 2)

write\_string (WindowID, i)

disp\_message (…)

wait\_seconds(0.5)

endfor

wait\_seconds(0.5)

if (NumberCircles ==2)

\*计算两圆中心距

distance\_pp(…)

disp\_message (…)

endif

wait\_seconds(0.5)

\*将提取出的边缘分割成线段或圆

segment\_contours\_xld (…)

\*按照宽度选出底边

select\_shape\_xld(…)

dev\_display (selectedXLD2)

\*拟合底边并绘制

fit\_line\_contour\_xld (…)

\* 绘图略去…

wait\_seconds(0.5)

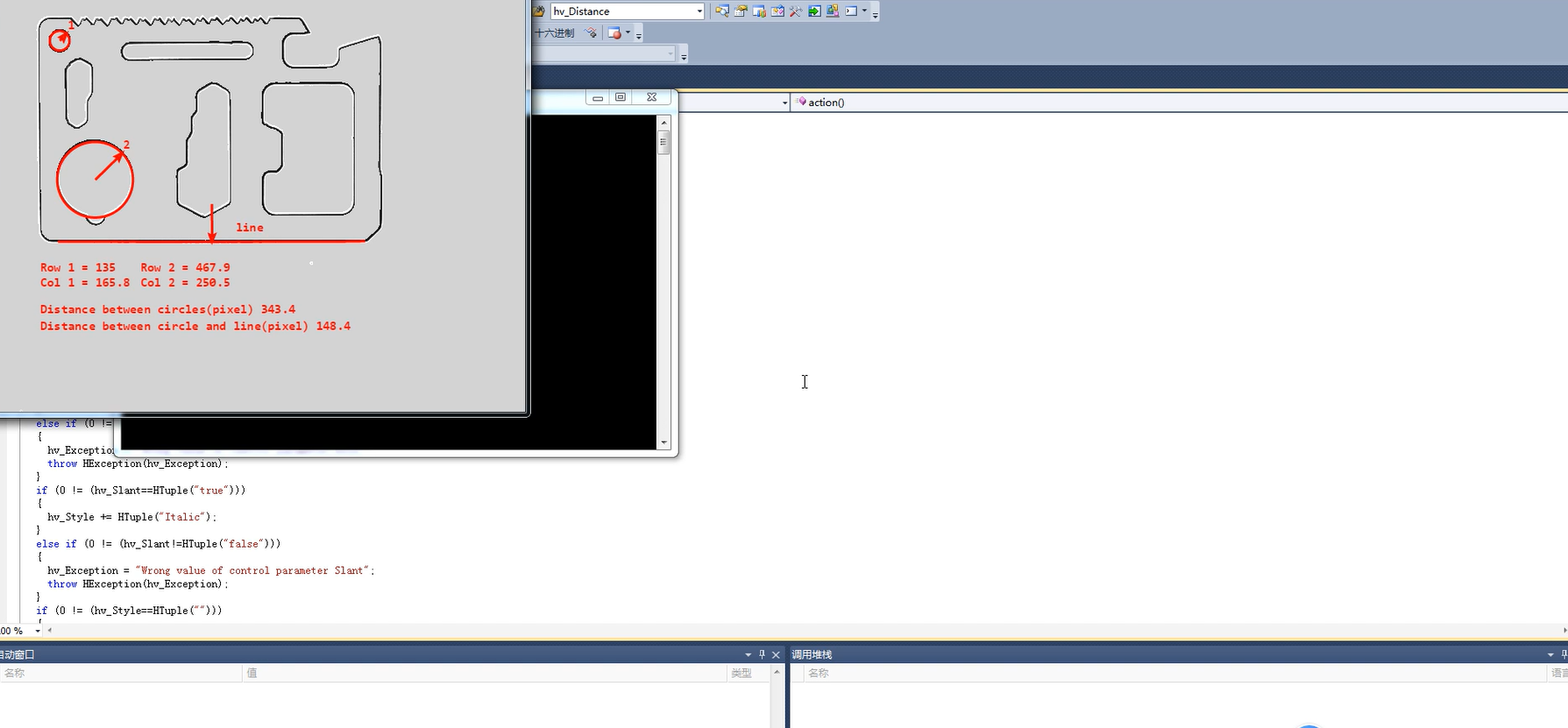
\*计算圆形与底边距离

distance\_pl(…)

disp\_message (…)

wait\_seconds(3)

## 任务2（2D测量）效果图



上面整个检测流程有录制视频，可通过此链接查看：https://pan.baidu.com/s/1PgDw1ZT9130Vwo3nVlwETw?pwd=0430

# 实验4

## Halcon代码

\* 任务1

dev\_update\_off ()

\* 读取参考图像

read\_image (Image, 'E:/210320621/OCR/4.jpg')

\* 获取图片大小以正确显示

get\_image\_size (Image, Width, Height)

dev\_close\_window ()

\* 显示图片，大小等于原图大小

dev\_open\_window (0, 0, Width, Height, 'black', WindowHandle)

\* 生成矩形

gen\_rectangle1 (Rectangle, …)

\* 裁剪出ROI

dev\_display (…)

\* 调试时使用

\*stop()

wait\_seconds(1)

\* 阈值分割

threshold (ImageReduced, Region, 200, 255)

\* 连接连通域

connection (Region, ConnectedRegions)

dev\_display (…)

\* 调试时使用

\* stop()

wait\_seconds(1)

select\_shape(ConnectedRegions, SelectedRegions, 'area', 'and', 800, 99999)

\* 连通域排序

sort\_region(SelectedRegions, SortedRegions, 'first\_point', 'true', 'column')

\* 读分类器

read\_ocr\_class\_mlp(…)

\* 取反图像（只有黑底图像需要）

invert\_image(ImageReduced, ImageInvert)

\* 字符识别

do\_ocr\_multi\_class\_mlp(SortedRegions, ImageInvert, OCRHandle, Class, Confidence)

set\_display\_font (WindowHandle, 24, 'mono', 'true', 'false')

\* 计数

count\_obj (SortedRegions, NumberCharacters)

\* 显示信息

for index:=0 to NumberCharacters-1 by 1

disp\_message(WindowHandle, Class[index], 'window', 12, 12 + index \* 40, 'red', 'false')

endfor

\* 调试时使用

\* stop()

wait\_seconds(1)

\* 任务2

\* 划取汉字区域ROI

gen\_rectangle1 (ModelRegion, 205, 256, 360, 316)

\* 剪裁

reduce\_domain (Image, ModelRegion, TemplateImage)

create\_shape\_model (TemplateImage, 4, 0, rad(360), 0.0175, ['none', 'no\_pregeneration'], 'use\_polarity', [11,17.4], 4, ModelID)

get\_shape\_model\_contours(ModelContours, ModelID, 1)

area\_center (ModelRegion, ModelRegionArea, RowRef, ColumnRef)

vector\_angle\_to\_rigid (0, 0, 0, RowRef, ColumnRef, 0, HomMat2D)

\* 将轮廓移到轮廓所属的位置

affine\_trans\_contour\_xld (ModelContours, TransContours, HomMat2D)

dev\_display (ModelRegion)

dev\_display (TransContours)

\* 调试时使用

\* stop()

wait\_seconds(1)

TestImages:=['E:/210320621/OCR/1.jpg','E:/210320621/OCR/3.jpg', 'E:/210320621/OCR/6.jpg']

\* 弹出不同窗口，需要不同的窗口ID号

WindowHandleI:=3200

for T:=0 to 2 by 1

WindowHandleI:=3200+T

read\_image(Image, TestImages[T])

\* 获取图片大小以正确显示

get\_image\_size (Image, Width, Height)

\* 显示图片，大小等于原图大小

dev\_open\_window (0, 0, Width, Height, 'black', WindowHandleI)

find\_shape\_model(Image, ModelID, 0, rad(360), 0.5, 0, 0.5, 'least\_squares', 4, 0.9, Row, Column, Angle, Score)

\* 显示匹配效果也可尝试

\* dev\_display\_shape\_matching\_results (ModelID, 'yellow', Row[K], Column[K], Angle[K], Scale[K], Scale[K], 0)

\* 显示轮廓

for I:=0 to |Score| -1 by 1

\* 从恒等映射开始

hom\_mat2d\_identity(HomMat2D)

\* 添加旋转量

hom\_mat2d\_rotate(HomMat2D, Angle[I], 0, 0, HomMat2D)

\* 添加平移量

hom\_mat2d\_translate(HomMat2D, Row[I], Column[I], HomMat2D)

\* 将轮廓移到轮廓所属的位置

affine\_trans\_contour\_xld (ModelContours, TransContours, HomMat2D)

\* 调试时使用

\* stop()

\* wait\_seconds(1)

endfor

\* 将车牌位置通过仿射变换放正

for I:=0 to |Score| -1 by 1

\* 从恒等映射开始

hom\_mat2d\_identity(RectificationHomMat2D)

\* 添加平移量，现在模板中心从(RowRef, ColRef) 变到了(Row[I], Column[I])，则将其平移回来就需要施加两坐标之差

hom\_mat2d\_translate(RectificationHomMat2D, RowRef-Row[I], ColumnRef-Column[I], RectificationHomMat2D)

\* 再添加旋转量，模板中心已经正确为(RowRef, ColRef)，原模板没有旋转，

\* 现在有了Angle[I]的旋转，则需要绕(RowRef, ColRef)转 -Angle[I] 才能转回来

hom\_mat2d\_rotate(RectificationHomMat2D, -Angle[I], RowRef, ColumnRef, RectificationHomMat2D)

\* 向图片施加仿射变换

affine\_trans\_image(Image, RectifiedImage, RectificationHomMat2D, 'constant', 'false')

\* 接下来进行OCR

\* 生成矩形，此处因经过仿射变换，取值和之前是相同的

gen\_rectangle1 (Rectangle, 205, 316, 360, 810)

\* 裁剪出ROI

reduce\_domain (RectifiedImage, Rectangle, ImageReduced)

\* 阈值分割

threshold (ImageReduced, Region, 200, 255)

\* 连接连通域

connection (Region, ConnectedRegions)

select\_shape(ConnectedRegions, SelectedRegions, 'area', 'and', 800, 99999)

\* 连通域排序

sort\_region(SelectedRegions, SortedRegions, 'first\_point', 'true', 'column')

\* 读分类器

read\_ocr\_class\_mlp(…)

\* 取反图像（只有黑底图像需要）

invert\_image(ImageReduced, ImageInvert)

\* 字符识别

do\_ocr\_multi\_class\_mlp(SortedRegions, ImageInvert, OCRHandle, Class, Confidence)

dev\_display(Image)

set\_display\_font (WindowHandleI, 24, 'mono', 'true', 'false')

\* 计算字符总数

count\_obj (SortedRegions, NumberCharacters)

\* 显示信息

disp\_message(WindowHandleI, 'Image' + T, 'window', 12, 12, 'red', 'true')

for index:=0 to NumberCharacters-1 by 1

disp\_message(WindowHandleI, Class[index], 'window', 40, 12 + index \* 40, 'red', 'false')

endfor

endfor

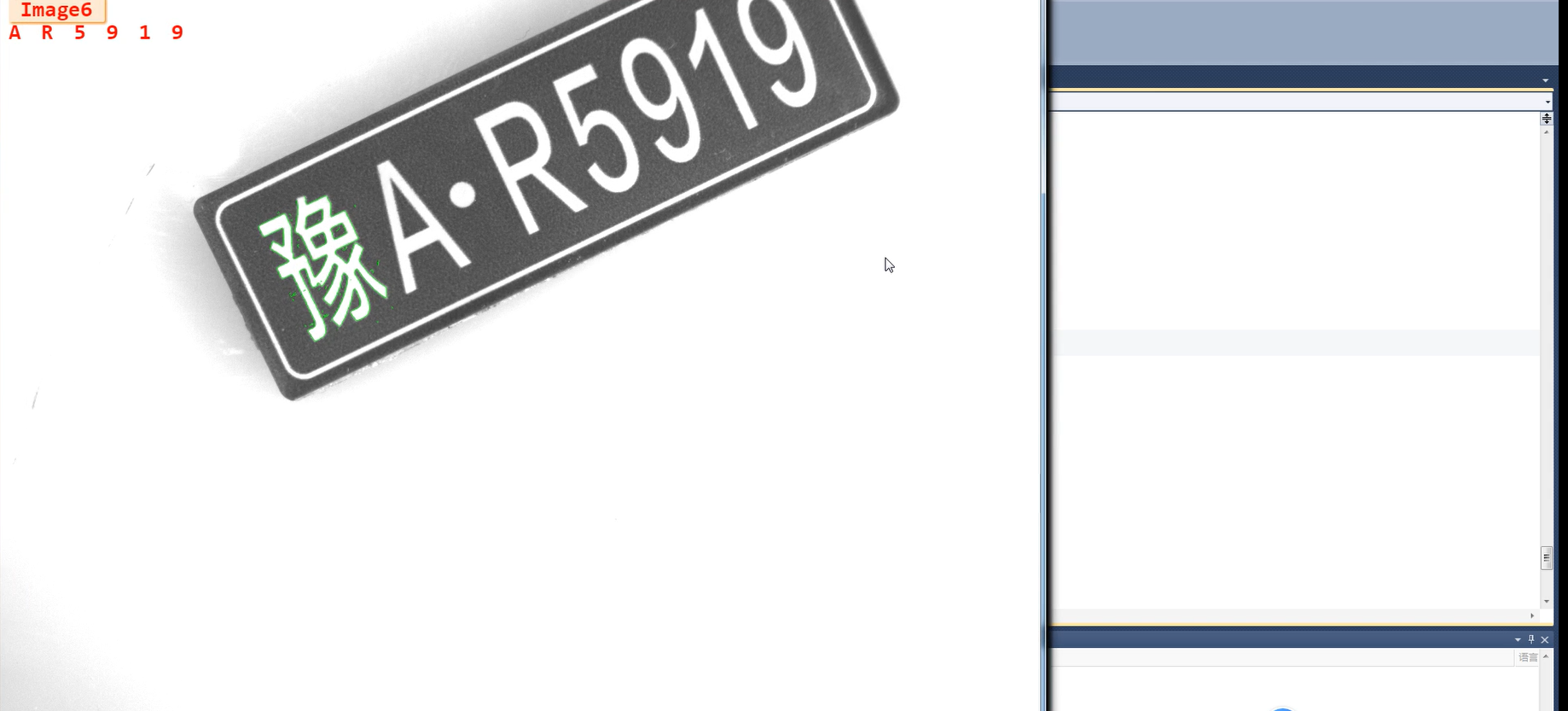
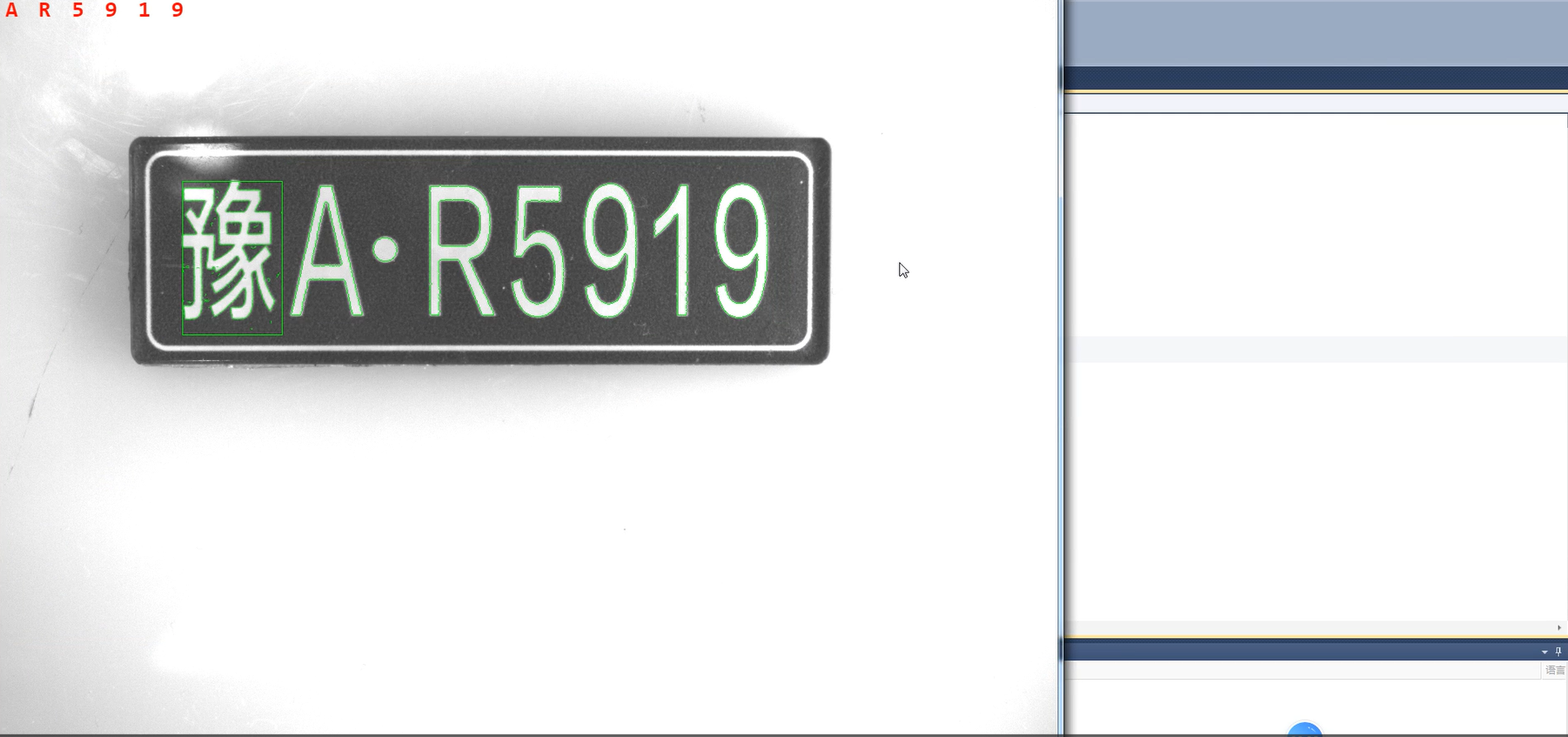
dev\_display (TransContours)

\* stop()

wait\_seconds(3)

endfor

## 效果图



上面整个检测流程有录制视频，可通过此链接查看：https://pan.baidu.com/s/1k2u8iwfTMYms29ZfpVgPeg?pwd=0430