手動搭建一個車牌識別系統| 附源碼

小白學視覺 今天

車牌識別是一種圖像處理技術，用於識別不同車輛。這項技術被廣泛用於各種安全檢測中。現在讓我一起基於OpenCV編寫Python代碼來完成這一任務。

車牌識別的相關步驟

1.車牌檢測：第一步是從汽車上檢測車牌所在位置。我們將使用OpenCV中矩形的輪廓檢測來尋找車牌。如果我們知道車牌的確切尺寸，顏色和大致位置，則可以提高準確性。通常，也會將根據攝像機的位置和該特定國家/地區所使用的車牌類型來訓練檢測算法。但是圖像可能並沒有汽車的存在，在這種情況下我們將先進行汽車的，然後是車牌。

2.字符分割：檢測到車牌後，我們必須將其裁剪並保存為新圖像。同樣，這可以使用OpenCV來完成。

3.字符識別：現在，我們在上一步中獲得的新圖像肯定可以寫上一些字符（數字/字母）。因此，我們可以對其執行OCR（光學字符識別）以檢測數字。

1.車牌檢測

讓我們以汽車的樣本圖像為例，首先檢測該汽車上的車牌。然後，我們還將使用相同的圖像進行字符分割和字符識別。如果您想直接進入代碼而無需解釋，則可以向下滾動至此頁面的底部，提供完整的代碼，或訪問以下鏈接。https://github.com/GeekyPRAVEE/OpenCV-Projects/blob/master/LicensePlateRecoginition.ipynb

在次使用的測試圖像如下所示。

圖片來源鏈接：https : //rb.gy/lxmiuv

第1步：將圖像調整為所需大小，然後將其灰度。相同的代碼如下

```
img = cv2.resize(img, (620,480) )
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #convert to grey scale
```

調整大小後，可以避免使用較大分辨率的圖像而出現的以下問題，但是我們要確保在調整大小後，車號牌仍保留在框架中。在處理圖像時如果不再需要處理顏色細節，那麼灰度變化就必不可少，這加快了其他後續處理的速度。完成此步驟後，圖像將像這樣被轉換

步驟2：每張圖片都會包含有用和無用的信息，在這種情況下，對於我們來說，只有牌照是有用的信息，其餘的對於我們的程序幾乎是無用的。這種無用的信息稱為噪聲。通常，**使用雙邊濾波（模糊）會從圖像中刪除不需要的細節**。

```
gray = cv2.bilateralFilter(gray, 13, 15, 15)
```

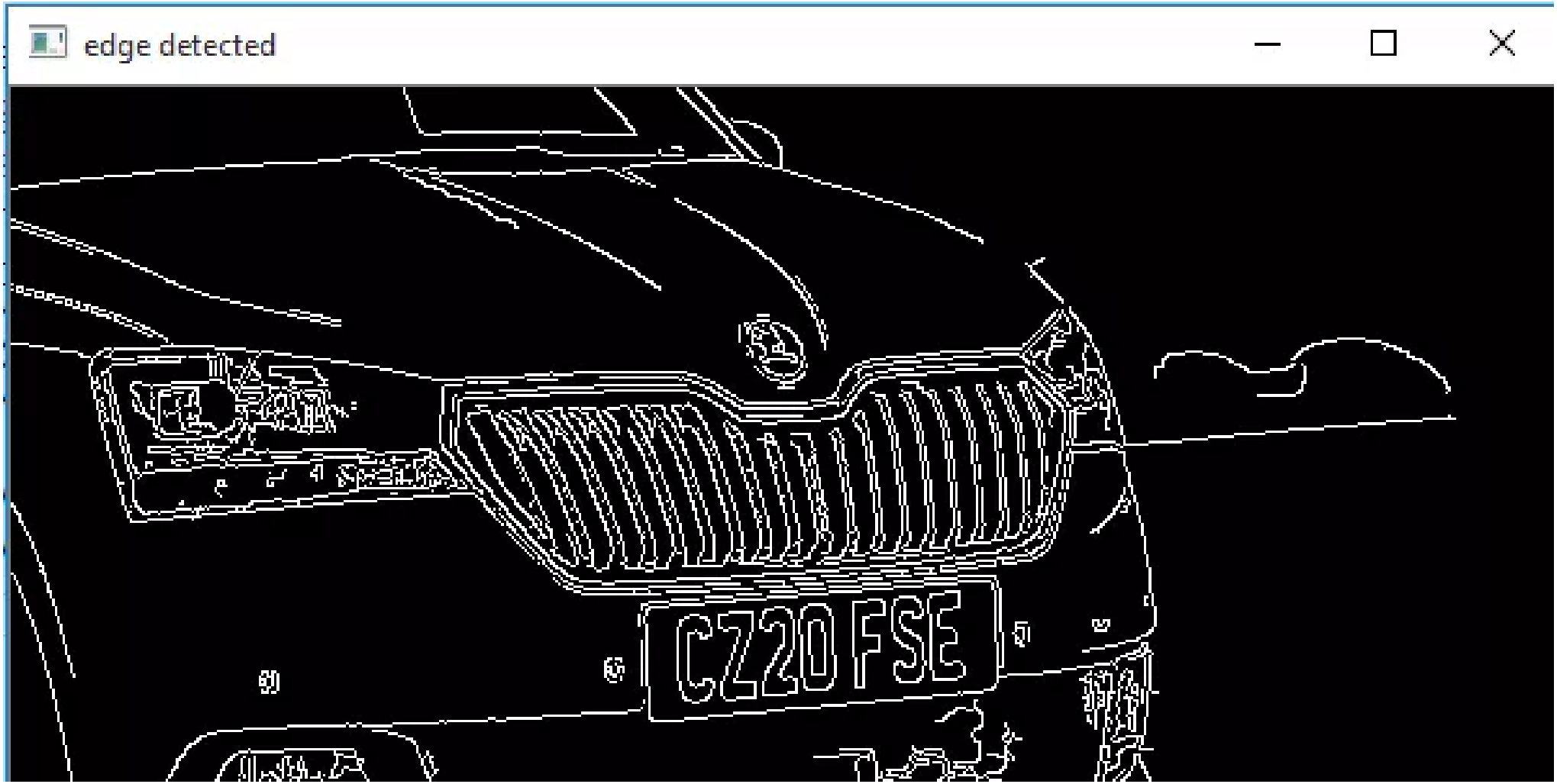
语法为 destination_image = cv2.bilateralFilter(source_image, diameter of pixel, sigmaColor, sigmaSpace)。我们也可以将sigma颜色和sigma空间从15增加到更高的值，以模糊掉更多的背景信息，但请注意不要使有用的部分模糊。输出图像如下所示可以看到该图像中的背景细节（树木和建筑物）模糊了。这样，我们可以避免程序处理这些区域。

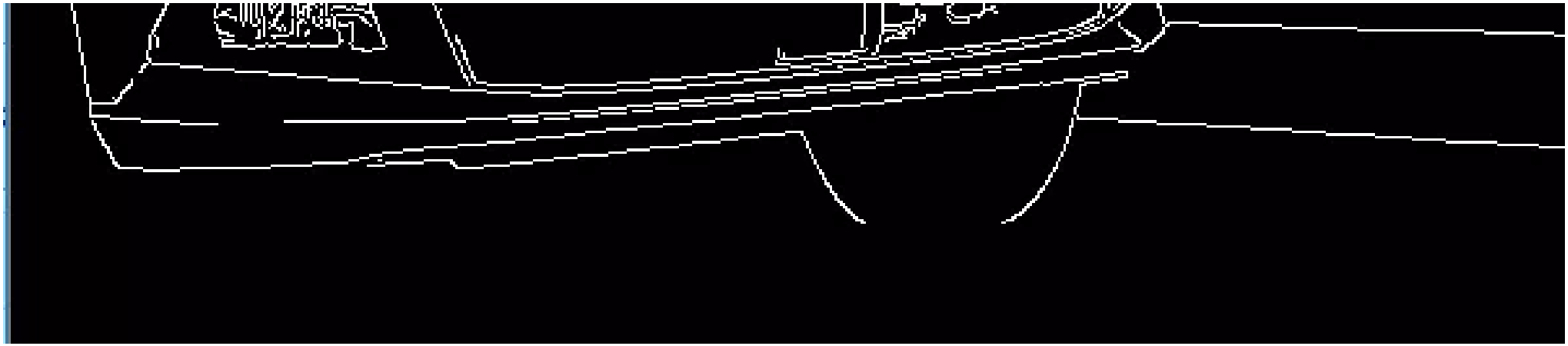

blurred
CZ20 FSE

步驟3：下一步是我们执行**边缘检测的**有趣步骤。有很多方法可以做到，最简单和流行的方法是使用**OpenCV中**的**canny edge方法**。执行相同操作的行如下所示

```
edged = cv2.Canny(gray, 30, 200) #Perform Edge detection
```

语法为*destination_image = cv2.Canny (source_image , thresholdValue 1 , thresholdValue 2)* 。阈值谷1和阈值2是最小和最大阈值。仅显示强度梯度大于最小阈值且小于最大阈值的边缘。结果图像如下所示





步骤4：现在我们可以开始**在图像上寻找轮廓**

```
contours=cv2.findContours(edged.copy(),cv2.RETR_TREE,
                                        cv2.CHAIN_APPROX_SIMPLE)
contours = imutils.grab_contours(contours)
contours = sorted(contours,key=cv2.contourArea, reverse = True)[:10]
screenCnt = None
```

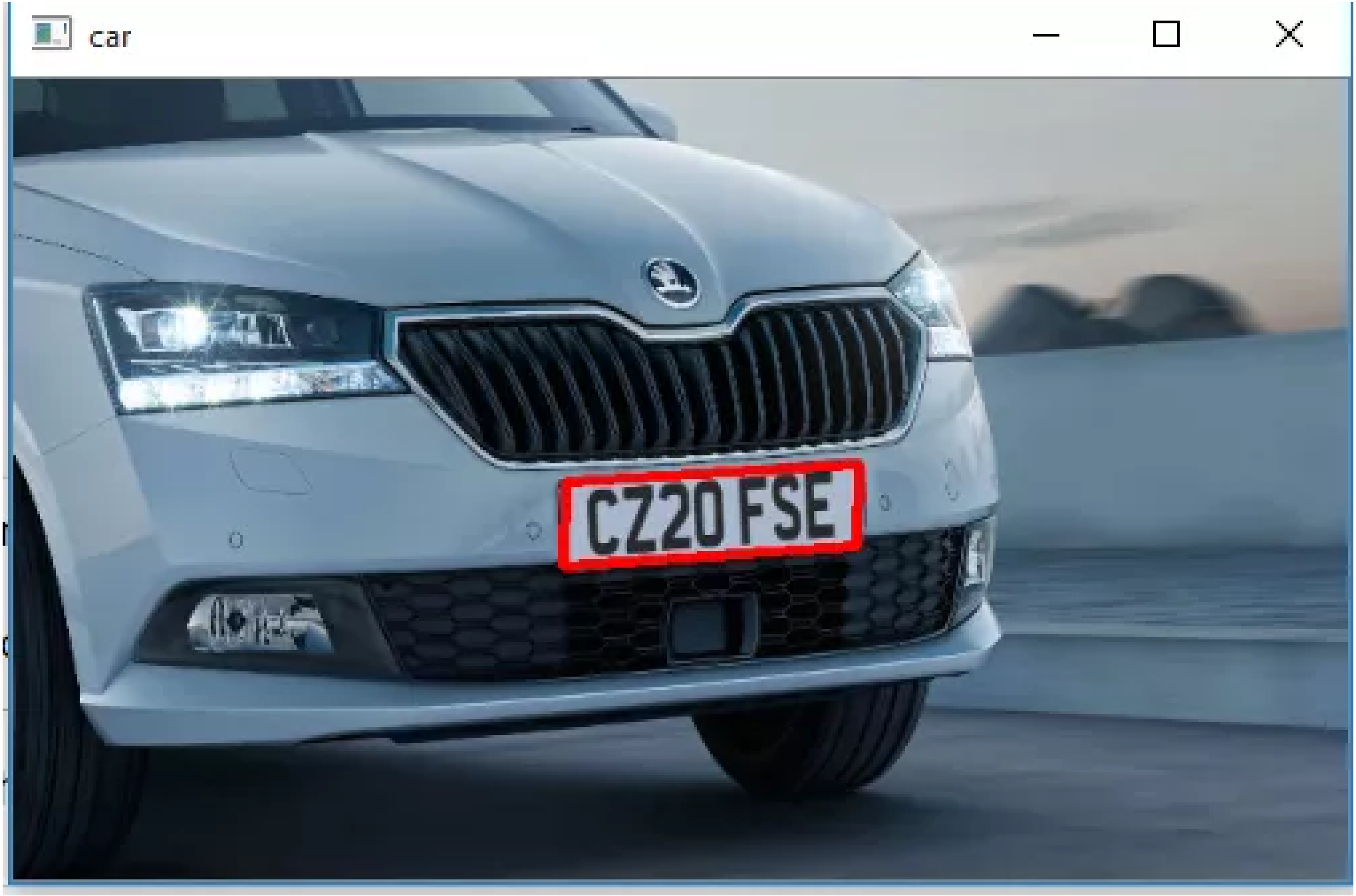
一旦检测到计数器，我们就将它们从大到小进行排序，并只考虑前10个结果而忽略其他结果。在我们的图像中，计数器可以是具有闭合表面的任何事物，但是在所有获得的结果中，牌照号码也将存在，因为它也是闭合表面。

为了过滤获得的结果中的车牌图像，我们将遍历所有结果，并检查其具有四个侧面和闭合图形的矩形轮廓。由于车牌肯定是四边形的矩形。

```
for c in cnts:
    # approximate the contour
    peri = cv2.arcLength(c, True)
```

```
    approx = cv2.approxPolyDP(c, 0.018 * peri, True)
    # if our approximated contour has four points, then
    # we can assume that we have found our screen
    if len(approx) == 4:
        screenCnt = approx
        break
```

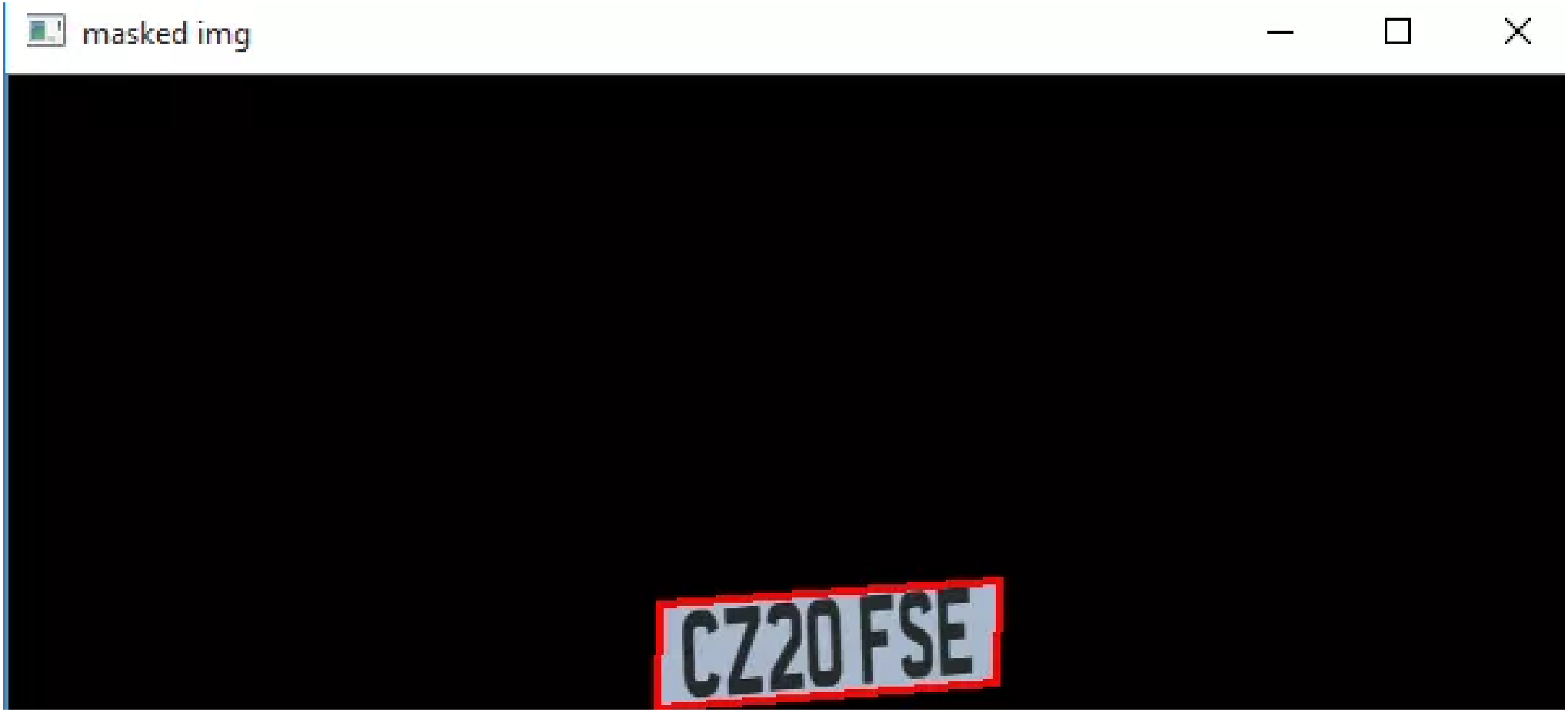
找到正确的计数器后，我们将其保存在名为*screenCnt*的变量中，然后在其周围绘制一个矩形框，以确保我们已正确检测到车牌。


car
CZ20 FSE

步骤5：现在我们知道车牌在哪里，剩下的信息对我们来说几乎没有用。因此，我们可以**对整个图片进行遮罩，除了车牌所在的地方**。相同的代码如下所示

```
# Masking the part other than the number plate
mask = np.zeros(gray.shape,np.uint8)
new_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)
new_image = cv2.bitwise_and(img,img,mask=mask)
```

被遮罩的新图像将如下所示



2.字符分割

车牌识别的下一步是通过**裁剪**车牌**并将其保存为新图像**，将车牌从图像中分割出来。然后，我们可以使用此图像来检测其中的字符。下面显示了从主图像裁剪出ROI（感兴趣区域）图像的代码

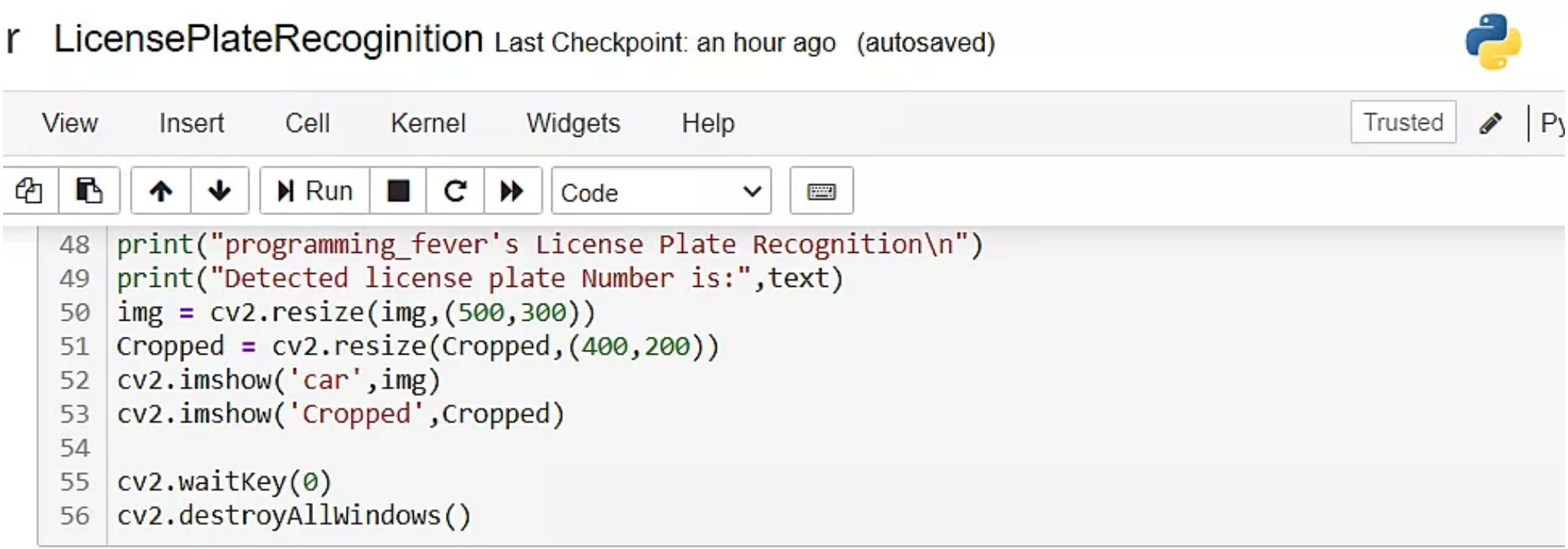
```
# Now crop
(x, y) = np.where(mask == 255)
(topx, topy) = (np.min(x), np.min(y))
(bottomx, bottomy) = (np.max(x), np.max(y))
Cropped = gray[topx:bottomx+1, topy:bottomy+1]
```

结果图像如下所示。通常添加到裁剪图像中，如果需要，我们还可以对其进行灰色处理和边缘化。这样做是为了改善下一步的字符识别。但是我发现即使使用原始图像也可以正常工作。

3.字符识别

该**车牌识别**的最后一步是**从分割的图像中**实际**读取车牌信息**。就像前面的教程一样，我们将使用***pytesseract***包从图像读取字符。相同的代码如下

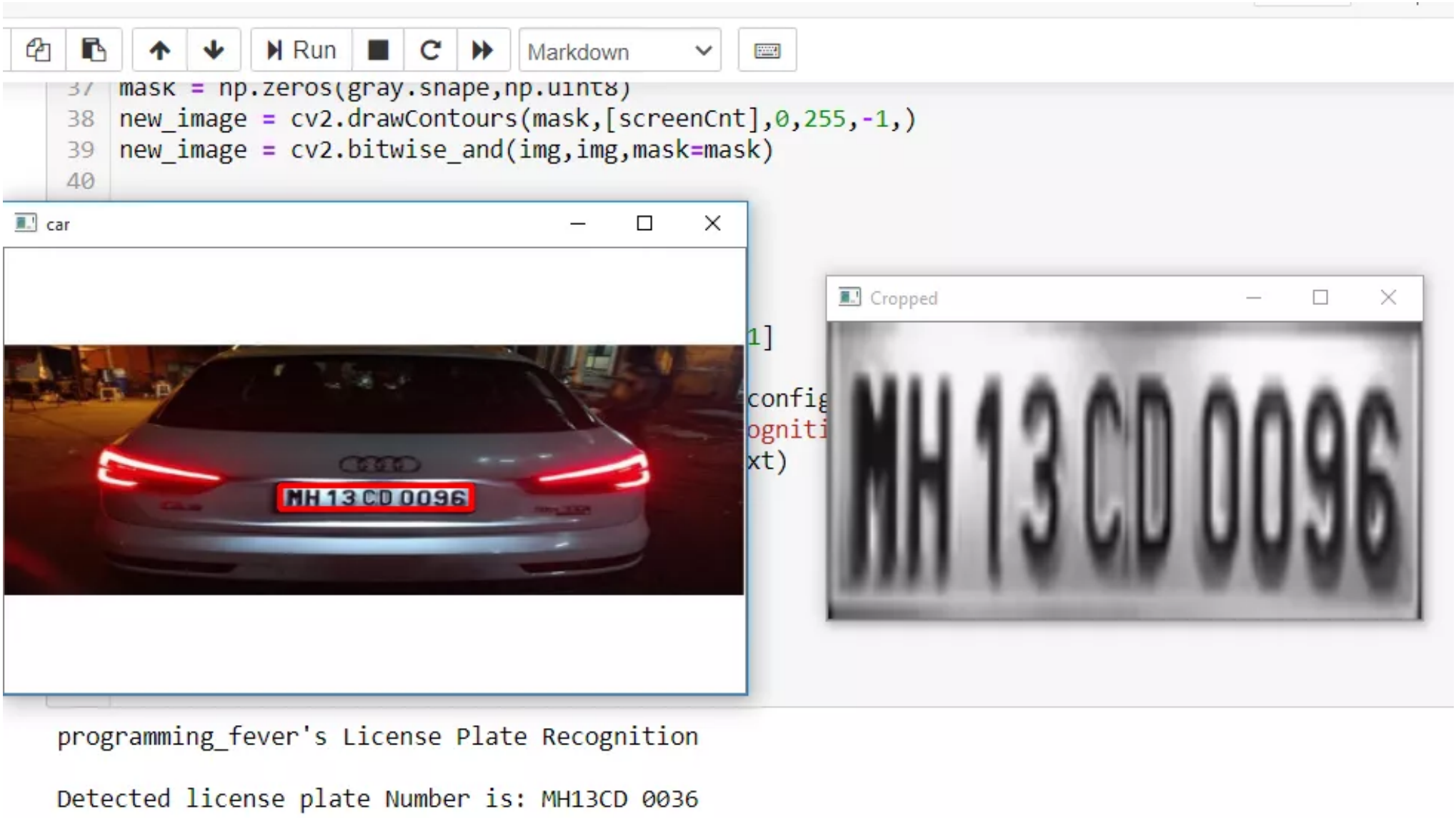
```
#Read the number plate
text = pytesseract.image_to_string(Cropped, config='--psm 11')
print("Detected license plate Number is:",text)
```



原始图像上印有数字“ CZ20FSE”，并且我们的程序检测到它在jupyter笔记本上打印了相同的值。

车牌识别失败案例

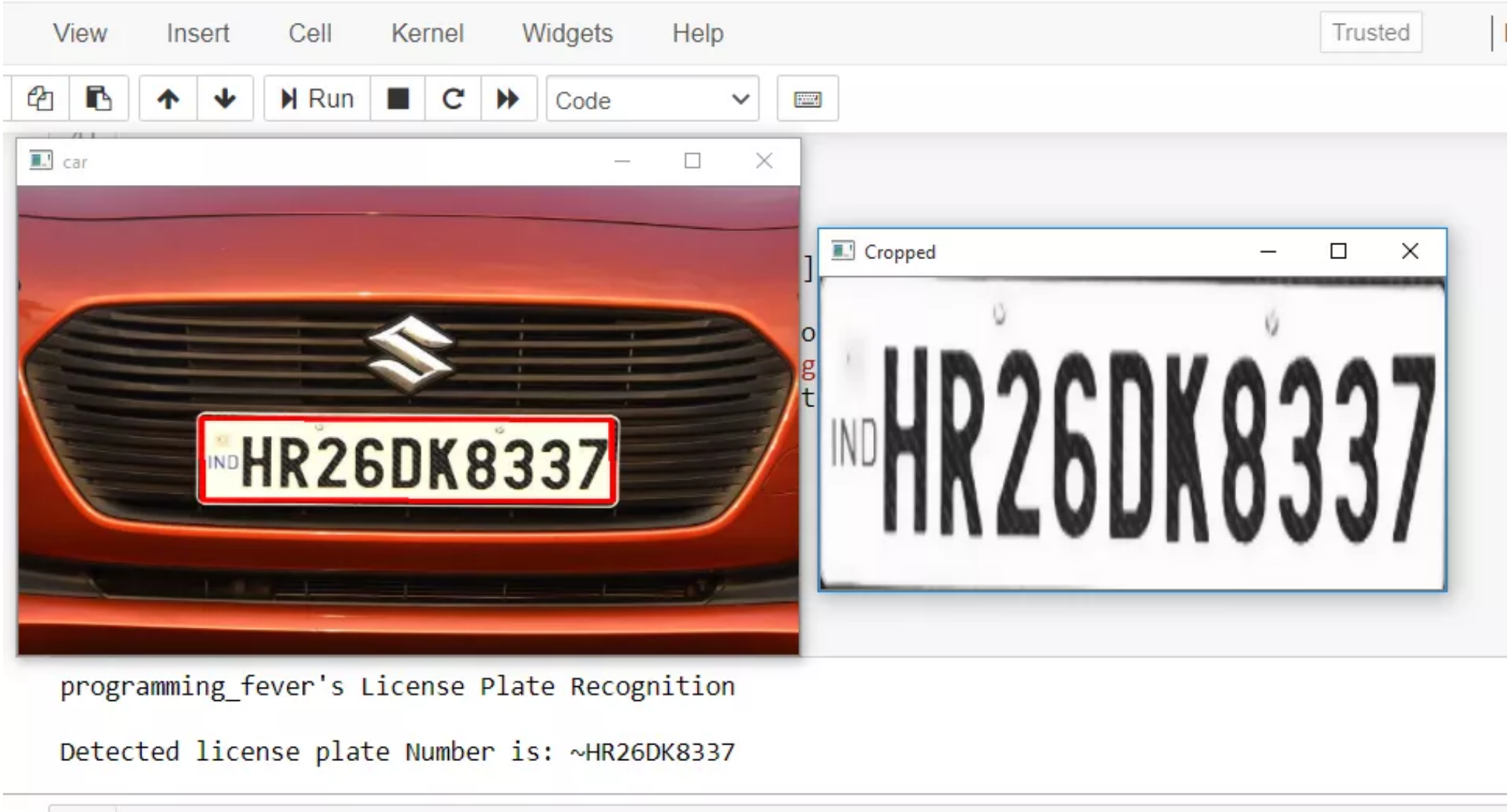
车牌识别的完整代码，其中包含程序和我们用来检查程序的测试图像。要记住，此方法的结果将不准确。**准确度取决于图像的清晰度，方向，曝光等**。为了获得更好的结果，您可以尝试同时实现机器学习算法。

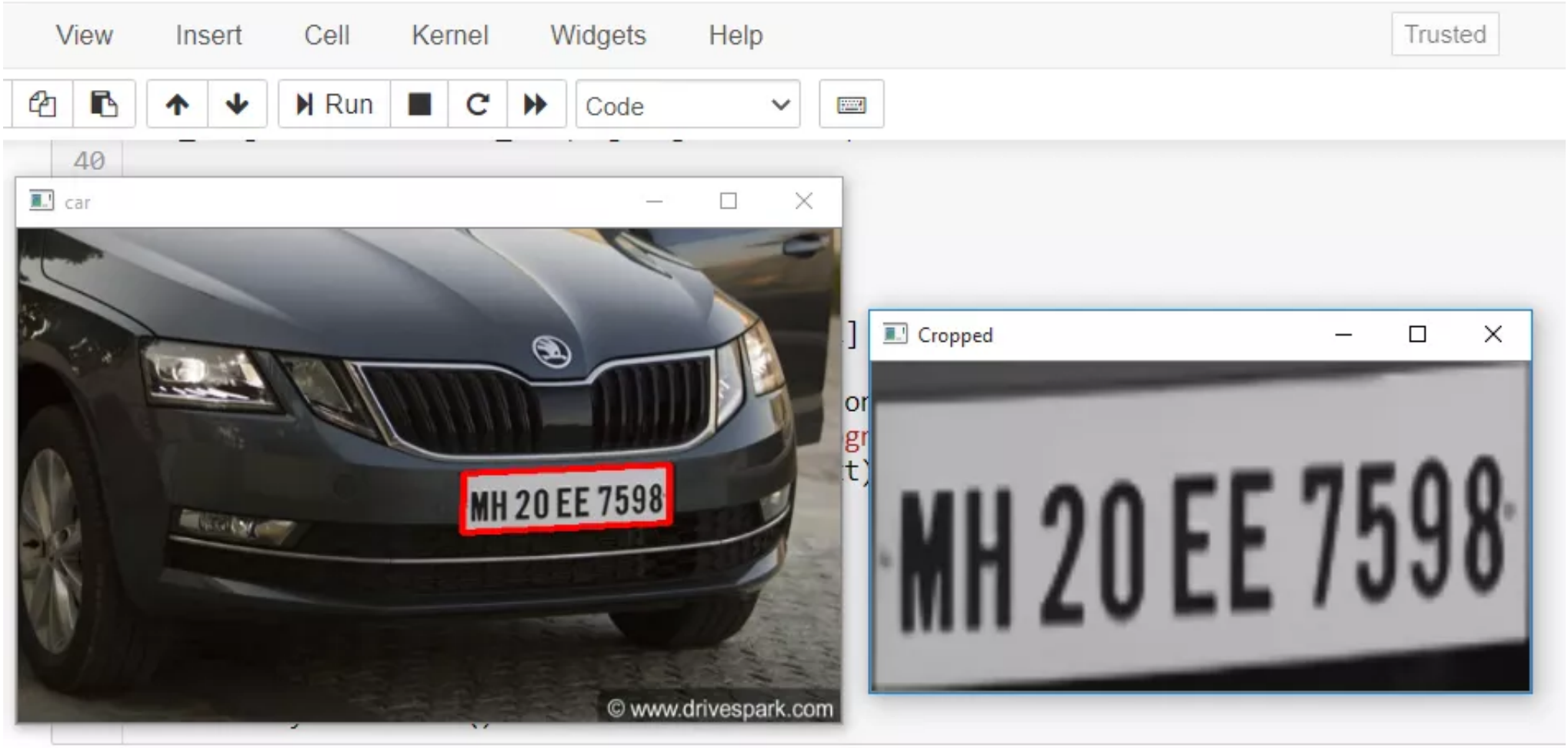


这个案例中我们的程序能够正确检测车牌并进行裁剪。但是，*Tesseract*库无法正确识别字符。OCR已将其识别为“ MH13CD 0036”，而不是实际的“ MH 13 CD 0096”。**通过使用更好的方向图像或配置*Tesseract*引擎，可以纠正此类问题。**

其他成功的例子

大多数时候，图像质量和方向都是正确的，程序能够识别车牌并从中读取编号。下面的快照显示了获得的成功结果。





完整代码

```
1  #@programming_fever
```

```
import cv2
import imutils
import numpy as np
import pytesseract
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files (x86)\Tesseract-OCR\tesseract.exe'

img = cv2.imread('D://skoda1.jpg',cv2.IMREAD_COLOR)
img = cv2.resize(img, (600,400) )

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
gray = cv2.bilateralFilter(gray, 13, 15, 15)

edged = cv2.Canny(gray, 30, 200)
contours = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contours = imutils.grab_contours(contours)
contours = sorted(contours, key = cv2.contourArea, reverse = True)[:10]
screenCnt = None

for c in contours:

    peri = cv2.arcLength(c, True)
    approx = cv2.approxPolyDP(c, 0.018 * peri, True)

    if len(approx) == 4:
        screenCnt = approx
        break

```

```
if screenCnt is None:
    detected = 0
    print ("No contour detected")
else:
     detected = 1

if detected == 1:
    cv2.drawContours(img, [screenCnt], -1, (0, 0, 255), 3)

mask = np.zeros(gray.shape,np.uint8)
new_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)
new_image = cv2.bitwise_and(img,img,mask=mask)

(x, y) = np.where(mask == 255)
(topx, topy) = (np.min(x), np.min(y))
(bottomx, bottomy) = (np.max(x), np.max(y))
Cropped = gray[topx:bottomx+1, topy:bottomy+1]

text = pytesseract.image_to_string(Cropped, config='--psm 11')
print("programming_fever's License Plate Recognition\n")
print("Detected license plate Number is:",text)
img = cv2.resize(img,(500,300))
Cropped = cv2.resize(Cropped,(400,200))
cv2.imshow('car',img)
cv2.imshow('Cropped',Cropped)

cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```

Github链接-https: //github.com/GeekyPRAVEE/OpenCV-Projects/blob/master/LicensePlateRecoginition.ipynb

下载1：OpenCV-Contrib扩展模块中文版教程

在「**小白学视觉**」公众号后台回复：**扩展模块中文教程，**即可下载全网第一份OpenCV扩展模块教程中文版，涵盖**扩展模块安装、SFM算法、立体视觉、目标跟踪、生物视觉、超分辨率处理**等二十多章内容。

下载2：Python视觉实战项目52讲

在「**小白学视觉**」公众号后台回复：**Python视觉实战项目，**即可下载包括**图像分割、口罩检测、车道线检测、车辆计数、添加眼线、车牌识别、字符识别、情绪检测、文本内容提取、面部识别**等31个视觉实战项目，助力快速学校计算机视觉。

下载3：OpenCV实战项目20讲

在「**小白学视觉**」公众号后台回复：**OpenCV实战项目20讲，**即可下载含有**20**个基于**OpenCV**实现20个**实战项目**，实现OpenCV学习进阶。

交流群

欢迎加入公众号读者群一起和同行交流，目前有**SLAM、三维视觉、传感器、自动驾驶、计算摄影**、检测、分割、识别、**医学影像、GAN、算法竞赛**等微信群（以后会逐渐细分），请扫描下面微信号加群，备注：”昵称+学校/公司+研究方向“，例如：”张三 + 上海交大 + 视觉SLAM“。**请按照格式备注，否则不予通过。**添加成功后会根据研究方向邀请进入相关微信群。**请勿**在群内发送**广告**，否则会请出群，谢谢理解~

喜欢此内容的人还喜欢

TransGAN更新！用纯Transformer构建高分辨率GAN

机器学习算法与Python学习

wǒ de shū yào chū bǎn lā！

hahaCoder

論文| COTR 一種基於Transformer的圖像匹配網絡

計算機視覺SLAM