**BÁO CÁO ĐỀ TÀI**

**Trường: Học viện công nghệ bưu chính viễn thông cơ sở TP.HCM**

**Nhóm 8:**

**Trần An Khoa N17DCCN074**

**Hồ Mai Anh Khoa N17DCCN073**

**Bành Cẩm Vỉnh N17DCCN187**

**Phạm Nhật Quan N17DCCN135**

**Môn: Xử lý ảnh**

**Tên đề tài: Nhận diện biển số xe**

1. **Import thư viện**

import cv2

import numpy as np

import imutils

import os

import shutil

import requests

1. **Khởi tạo kích thước của kí tự trên biển số**

digit\_w =30

digit\_h =60

#Ba hàm này đưa về giá trị cho mỗi cột (x, y, ký tự) nhằm để vào hàm sort(key=) để sắp xếp

def takeSecond(elem):

return elem[1]

def takeFirst(elem):

return elem[0]

def takeChar(elem):

return elem[2]

def Pretreatment(imgLP):

1. **Tiền xử lý**

#Chuyển ảnh thành ảnh xám, khử noise, cân bằng histogram và chuyển ảnh thành ảnh nhị phân.

def Pretreatment(imgLP):

grayImg = cv2.cvtColor(imgLP, cv2.COLOR\_BGR2GRAY)

noise\_removal = cv2.bilateralFilter(grayImg,9,75,75)

equal\_histogram = cv2.equalizeHist(noise\_removal)

ret, binImg = cv2.threshold(grayImg, 100, 255, cv2.THRESH\_BINARY\_INV+ cv2.THRESH\_OTSU)

kerel3 = cv2.getStructuringElement(cv2.MORPH\_RECT,(1,4))

binImg = cv2.morphologyEx(binImg,cv2.MORPH\_DILATE,kerel3)

return binImg

1. **Phát hiện contour**

def contours\_detect(binImg):

cnts, \_ = cv2.findContours(binImg, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_NONE)

1. **Vẽ contour**

def draw\_rects\_on\_img(img, cnts):

imgtemp=img.copy()

cv2.drawContours(imgtemp,cnts,-1,(0,120,0),1)

return imgtemp

1. **Tìm và sắp xếp ký tự trên biển số**

#khởi tạo

plate\_number=''

coorarr=[]

firstrow=[]

lastrow=[]

#load file cascade để nhận diện biển số trong ảnh và load file svm để nhận diện ký tự trên biển số

model\_svm =cv2.ml.SVM\_load(**'svm.xml'**)

plate\_cascade = cv2.CascadeClassifier("./**cascade2.xml**")

def find\_number(cnts,binImg,imgtemp):

count=0

global plate\_number

global coorarr

global firstrow

global lastrow

(himg,wimg,chanel)=imgtemp.shape

if(wimg/himg >2):

hf=himg\*0.6

hl=himg\*0.8

else:

hf=0.25\*himg

hl=0.5\*himg

plate\_number = ''

for c in (cnts):

x,y,w,h=cv2.boundingRect(c)

cv2.rectangle(imgtemp, (x, y), (x + w, y + h), (0, 255, 0), 1)

if h/w >1.5 and h/w <4 and h>=hf or cv2.contourArea(c)>4000 and h<= hl:

#cái này áp dụng cho cả biển xe máy xe hơi

#if h/w >1.5 and h/w <4 and h>= hf and h<= hl:

#2500 là kích thước tối thiểu của diện tích contour đảm bảo cho các contour "rác" không nhận diện

#7000 là kích thước tối đa của contour số đảm bảo không nhận diện contour "rác" có cỡ lớn, chủ yếu đến từ viền biển số

#đóngóng khung cho kí tự

cv2.rectangle(imgtemp, (x, y), (x + w, y + h), (0, 0, 255),2)

#crop thành những số riêng lẻ

crop=imgtemp[y:y+h, x:x+w]

#dùng để ghi vào thư mục number

count+=1

cv2.imwrite('./number/number%d.jpg'% count,crop)

#lưu vào mảng tọa độ

#tách số và predict

#sao chép cái ảnh bin để giữ ảnh kia tuy hơi dài nhưng an toàn

binImgtemp=binImg

curr\_num=binImgtemp[y:y+h, x:x+w]

#xử lí để tí nữa đưa cái này vào hàm nó ràng buộc input phải kiểu dữ liệu

#đầu tiên là resize lại cho nó cùng kích thước nhau cũng như= kích thước khi train

curr\_num=cv2.resize(curr\_num,dsize=(digit\_w,digit\_h))

\_, curr\_num=cv2.threshold(curr\_num,30,255,cv2.THRESH\_BINARY)

#chuyển thành np để tí xài tạo thành mảng numpy

curr\_num= np.array(curr\_num,dtype=np.float32)

#reshape lại nha ae tự sắp xếp thành hàng ngang thì phải

#ví dụ ảnh cao 2 ngang 10 thì giờ thành 1 hàng n20 pixel

curr\_num=curr\_num.reshape(-1,digit\_w\*digit\_h)

result=model\_svm.predict(curr\_num)[-1]

result= int(result[0,0])

if result<=9:

result= str(result)

else:

result=chr(result)

#ráp kết quả các ký tự và in kết quả lên ảnh

coorarr.append((x,y,result))

cv2.putText(imgtemp,result,(x-50,y+50),cv2.FONT\_HERSHEY\_COMPLEX,3,(0, 255, 0), 2, cv2.LINE\_AA)

#sắp xếp theo y, nhằm lấy hàng đầu tiên với y thấp nhất

coorarr.sort(key=takeSecond)

#Lấy ra 4 giá trị đầu tiên có y thấp nhất, nhằm đưa về hàng đầu

firstrow = coorarr[:4]

#Sắp xếp lại hàng đầu từ trái qua phải

firstrow.sort(key=takeFirst)

#tương tự với hàng sau

lastrow = coorarr[4:]

lastrow.sort(key=takeFirst)

#Đưa từng hàng vào

for x, y, c in firstrow:

plate\_number+=c

for x, y, c in lastrow:

plate\_number+=c

return imgtemp, plate\_number

1. **Phát hiện**

def detect(img):

(himg,wimg,chanel)=img.shape

if(wimg/himg >2):

img=cv2.resize(img,dsize=(1000,200))

else:

img=cv2.resize(img,dsize=(800,500))

binImg=Pretreatment(img) #tiền xử lý

cnts=contours\_detect(binImg) #vẽ contour

imgtemp=draw\_rects\_on\_img(img,cnts)

#tìm và sắp xếp ký tự trên biển

imgtemp2, sort\_number = find\_number(cnts,binImg,imgtemp)

#in biển số ra màn hình

print('bien so xe: ',sort\_number)

plate\_number=''

coorarr.clear()

return sort\_number

1. **Nhận diện biển số xe trong ảnh**

def findLP\_img(OriImg):

#nhận diện biển trong img

plates = plate\_cascade.detectMultiScale(OriImg, 1.1, 3)

#Tạo ảnh trước khi cắt

img = OriImg

#in vùng chứa biển số và cắt

for (x,y,w,h) in plates:

cv2.rectangle(OriImg,(x,y),(x+w,y+h),(255,0,0),1)

img = OriImg[y:y+h, x:x+w]

plate\_num = detect(img)

cv2.putText(OriImg, plate\_num, (x, y-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (255,0,0), 2)

cv2.imshow("Original image", OriImg)

return img

def pic(OriImg):

#Tìm biển số gọi lại hàm trên

img=findLP\_img(OriImg);

#resize lại hình

(himg,wimg,chanel)=img.shape

if(wimg/himg >2):

img=cv2.resize(img,dsize=(1000,200))

else:

img=cv2.resize(img,dsize=(800,500))

cv2.imshow('Image',img)

binImg=Pretreatment(img)

cnts=contours\_detect(binImg)

imgtemp=draw\_rects\_on\_img(img,cnts)

imgtemp, sort\_number=find\_number(cnts,binImg,imgtemp)

cv2.imshow('binary',binImg)

cv2.imshow('result',imgtemp)

print('bien so xe: ',sort\_number)

cv2.waitKey()

cv2.destroyAllWindows()

1. **Nhận diện biển số xe bằng WebCam**

def video\_webcam():

cap = cv2.VideoCapture(0)

while True:

ret, frame = cap.read()

print(ret)

img = findLP\_img(frame);

key = cv2.waitKey(1)

if key == ord("q"):

break

cv2.destroyAllWindows()

1. **Nhận diện biển số xe bằng IP Camera**

def ipCam():

while True:

# Lấy từ ip nội bộ của camera, ở đây sử dụng phần mềm ip camera cho android nên việc kết nối khá đơn giản

# nên chuyển độ phân giải về thấp để mượt hơn

#Gọi ip, đưa kết quả từ web (là hình ảnh) về

img\_res = requests.get("http://192.168.1.101:8080/shot.jpg")

img\_arr = np.array(bytearray(img\_res.content), dtype = np.uint8)

img = cv2.imdecode(img\_arr,-1)

img = findLP\_img(img)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

1. **Train biển số**

Để phát hiện biển số (định vị biển số) dùng Haar Cascade với tool của opencv là opencv\_createsamples.exe để tạo vecto chứa ảnh và opencv\_traincascade.exe để train theo hướng dẫn bên dưới:

<https://thigiacmaytinh.com/phat-hien-doi-tuong-p1-ly-thuyet/>

Chuẩn bị 2 bộ ảnh, một bộ ảnh có chứa biển số là ảnh dương. Bộ còn lại là những ảnh bất kỳ không chứa biển số là ảnh âm.

Khi training, chương trình sẽ rút ra đặc trưng rồi so sánh với các ảnh âm, nếu đặc trưng đó được tìm thấy trong ảnh âm thì đặc trưng đó không phù hợp.

Hoàn thành bước này sẽ tạo ra file **cascade2.xml**

1. **Train ký tự trên biển số**

Dùng SVM để nhận dạng ký tự. Chuẩn bị bộ ảnh (data) gồm các ký tự 0-9 và A-Z (không có IJ) để train các ký tự. Hoàn thành bước này sẽ tạo ra file **svm.xml**

import cv2

import os

import numpy as np

import glob

digit\_w = 30

digit\_h = 60

write\_path="data/"

def get\_digit\_data(path):#:, digit\_list, label\_list):

digit\_list = []

label\_list = []

for number in range(10):

i=0

for img\_org\_path in glob.iglob(path + str(number) + '/\*.jpg'):

print(img\_org\_path)

img = cv2.imread(img\_org\_path, 0)

img = np.array(img)

img = img.reshape(-1, digit\_h \* digit\_w)

# print(img.shape)

digit\_list.append(img)

label\_list.append([int(number)])

for number in range(65, 91):

#number = chr(number)

print(number)

i=0

for img\_org\_path in glob.iglob(path + str(number) + '/\*.jpg'):

print(img\_org\_path)

img = cv2.imread(img\_org\_path, 0)

img = np.array(img)

img = img.reshape(-1, digit\_h \* digit\_w)

# print(img.shape)

digit\_list.append(img)

label\_list.append([int(number)])

return digit\_list, label\_list

#lấy dữ liệu

digit\_path = "data/"

digit\_list, label\_list = get\_digit\_data(digit\_path)

digit\_list = np.array(digit\_list, dtype=np.float32)

digit\_list = digit\_list.reshape(-1, digit\_h \* digit\_w)

label\_list = np.array(label\_list)

label\_list = label\_list.reshape(-1, 1)

svm\_model = cv2.ml.SVM\_create()

# sử dụng SVM\_C\_SVC để có thể phần loại nhiều lớp n (n>2)

svm\_model.setType(cv2.ml.SVM\_C\_SVC)

# inter phân lớp dựa vào điểm tương đồng trong biểu đồ histogram của chúng

svm\_model.setKernel(cv2.ml.SVM\_INTER)

#svm\_model.setKernel(cv2.ml.SVM\_RBF)

#Max\_Iter là kiểu điều kiện dừng

#100 số lần lặp tối đa cho phép

#1e-6 độ chính xác của model khi đạt sẽ không lặp nữa

svm\_model.setTermCriteria((cv2.TERM\_CRITERIA\_MAX\_ITER, 100, 1e-6))

svm\_model.train(digit\_list, cv2.ml.ROW\_SAMPLE, label\_list)

svm\_model.save("**svm.xml**")