"""

Example "Batch analysis from input-images to output-images": python barcode-detect.py [default]

#Example "Analyse individual image and show box": python barcode-detect.py --image input-images/barcode1.jpg

"""

import numpy as np

import cv2

import argparse

import glob

import os

# construct the argument parse and parse the arguments

#ap = argparse.ArgumentParser()

#ap.add\_argument("-i", "--image", required = False, help = "path to the image file")

#args = vars(ap.parse\_args())

for filename in glob.iglob('input-images/\*.jpg'):

#print('/foobar/%s' % filename)

#image = cv2.imread(args["image"])

image = cv2.imread(filename)

# convert the image to grayscale

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# compute the Scharr gradient magnitude representation of the images

# in both the x and y direction

gradX = cv2.Sobel(gray, ddepth = cv2.CV\_32F, dx = 1, dy = 0, ksize = -1)

gradY = cv2.Sobel(gray, ddepth = cv2.CV\_32F, dx = 0, dy = 1, ksize = -1)

# subtract the y-gradient from the x-gradient

gradient = cv2.subtract(gradX, gradY)

gradient = cv2.convertScaleAbs(gradient)

# blur and threshold the image

blurred = cv2.blur(gradient, (3, 3))

(\_, thresh) = cv2.threshold(blurred, 210, 250, cv2.THRESH\_BINARY)

# construct a closing kernel and apply it to the thresholded image

kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (21, 7))

closed = cv2.morphologyEx(thresh, cv2.MORPH\_CLOSE, kernel)

# perform a series of erosions and dilations

closed = cv2.erode(closed, None, iterations = 7)

closed = cv2.dilate(closed, None, iterations = 2)

#cv2.imshow("Image", closed)

#cv2.waitKey(0)

# find the contours in the thresholded image

(\_,cnts, \_) = cv2.findContours(closed.copy(), cv2.RETR\_EXTERNAL,

cv2.CHAIN\_APPROX\_SIMPLE)

# otherwise, sort the contours by area and compute the rotated

# bounding box of the largest contour

c = sorted(cnts, key = cv2.contourArea, reverse = True)[0]

rect = cv2.minAreaRect(c)

box = np.int0(cv2.boxPoints(rect))

cv2.drawContours(image, [box], -1, (0, 255, 0), 3)

cv2.imshow("Image", image)

cv2.waitKey(0)

image\_file = 'output-images/' + os.path.basename(filename);

cv2.imwrite(image\_file, image)