CS3150 Fall 2022 Final Project: Self-determined Topic

Professor: Feng Jiang

Student: Matthew Mazzaccaro

-all code used was provided in class by Professor

Statement of project:

Bicycle designs have unique features that allow for the potential identification. For this project my goal was to create a template generating tool that would process an photo to a binary image with enough detail for quick comparison ``. I used the tools learned in class to process the image.

Process to create template:

Read image from file

Convert to gray scale

Morphological transformation

Gaussian Blur

Apply Thresholding

A red and white bicycle

Description automatically generated with low confidence

https://www.santacruzbicycles.com/en-US/bikes/nomad

Image copied from

A picture containing text, bicycle

Description automatically generated

G\_ker=np.array([

       [ 1,  4,  7,  4, 1],

       [4, 16, 26, 16, 4],

       [7, 26, 41, 26, 7],

       [4, 16, 26, 16, 4],

       [1, 4, 7, 4, 1]])

G\_ker =G\_ker/np.sum(G\_ker)

Gaussian = cv.filter2D(img, -1, G\_ker)

Using cv2 library to apply a gaussian kernel that was made using a numpy array to blur the image

Trying morophological tools to identify features better

A picture containing text, bicycle

Description automatically generated

kernel = np.ones((3,3),np.uint8)

opening = cv.morphologyEx(Gaussian, cv.MORPH\_OPEN, kernel, iterations=1)

dilation = cv.dilate(Gaussian,kernel,iterations = 1)

Diagram

Description automatically generatedclosing = cv.morphologyEx(opening, cv.MORPH\_CLOSE,kernel, iterations=1)

A picture containing diagram

Description automatically generated

Applying threshold to create a black and white image with enough detail for template matching…?

ret,thresh1 = cv.threshold(opening,215,255,cv.THRESH\_BINARY)

ret,thresh2 = cv.threshold(closing,215,255,cv.THRESH\_BINARY)

Images produced come from low noise but still produce undesired effects.

The next step was to use the same program to produce images similar to the results of the above images from user generated images. I used an image similar to one a user might create with a noisy background.

A picture containing bicycle, grass, outdoor, parked

Description automatically generated

https://mtbdatabase.com/bikes/2022/santa-cruz/nomad

A picture containing shape

Description automatically generatedA picture containing shape

Description automatically generated

A noisy background produced less than desirable results. Removing the background is necessary. Unfortunately, this is as far as I got with this project. I spent more time trying to remove the background but was hopelessly searching for examples to use in the last few minutes of this assignment unsuccessfully. I partly have concluded that this may not be the best way to solve this problem. While searching for images to use I used google image search which produced amazing results. This suggests to me that a ML program may be a better choice. Ultimately, this is a feature to an application idea that I have and not the main feature. A convenience to the app, but certainly convenience is what makes technology desirable.

Here is the code I used:

import cv2 as cv

import numpy as np

import matplotlib.pyplot as plt

template = cv.imread('Final Project/bike1.jpg')

img = cv.cvtColor(template, cv.COLOR\_BGR2GRAY)

G\_ker=np.array([

       [ 1,  4,  7,  4, 1],

       [4, 16, 26, 16, 4],

       [7, 26, 41, 26, 7],

       [4, 16, 26, 16, 4],

       [1, 4, 7, 4, 1]])

G\_ker =G\_ker/np.sum(G\_ker)

Gaussian = cv.filter2D(img, -1, G\_ker)

plt.figure()

plt.title('Gaussian')

plt.imshow(cv.cvtColor(Gaussian,cv.COLOR\_BGR2RGB))

kernel = np.ones((3,3),np.uint8)

opening = cv.morphologyEx(Gaussian, cv.MORPH\_OPEN, kernel, iterations=1)

plt.figure()

plt.title('opening')

plt.imshow(cv.cvtColor(opening,cv.COLOR\_BGR2RGB))

closing = cv.morphologyEx(opening, cv.MORPH\_CLOSE,kernel, iterations=1)

plt.figure()

plt.title('closing')

plt.imshow(cv.cvtColor(closing,cv.COLOR\_BGR2RGB))

dilation = cv.dilate(Gaussian,kernel,iterations = 1)

plt.figure()

plt.title('dilation')

plt.imshow(cv.cvtColor(dilation,cv.COLOR\_BGR2RGB))

ret,thresh1 = cv.threshold(opening,215,255,cv.THRESH\_BINARY)

plt.figure()

plt.title('Threshold Open')

plt.imshow(cv.cvtColor(thresh1,cv.COLOR\_BGR2RGB))

ret,thresh2 = cv.threshold(closing,215,255,cv.THRESH\_BINARY)

plt.figure()

plt.title('Threshold Closed')

plt.imshow(cv.cvtColor(thresh2,cv.COLOR\_BGR2RGB))

plt.show()