Object Detection

Template matching

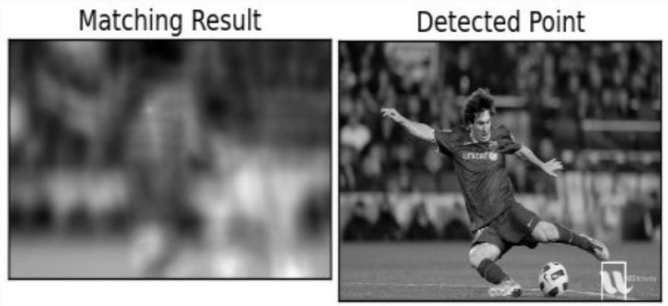
Slides a template image across another image until a match is found.

Template matching is different than shape matching, this uses an image, shape matching uses contours. There are different methods to perform the template match:

- cv2.TM_CCOEFF

- cv2.TM_CCOEFF_NORMED

- cv2.TM_CCORR



- cv2.TM_CCORR_NORMED

- cv2.TM_SQDIFF

- cv2.TM_SQDIFF_NORMED

https://docs.opencv.org/2.4/modules/imgproc/doc/object_detection.html

Template matching

A simple example:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img_rgb = cv2.imread('mario.png')
img_gray = cv2.cvtColor(img_rgb, cv2.COLOR_BGR2GRAY)
template = cv2.imread('mario_coin.png',0)
w, h = template.shape[::-1]

res =
cv2.matchTemplate(img_gray,template,cv2.TM_CCOEFF_NORMED)
threshold = 0.8
loc = np.where( res >= threshold)
for pt in zip(*loc[::-1]):
    cv2.rectangle(img_rgb, pt, (pt[0] + w, pt[1] + h), (0,0,255), 2)

cv2.imwrite('res.png',img_rgb)
```

Template matching

Not effective with:

- Rotations
- Distortions
- Size (Scaling)
- Photometric changes (blightness, contrast,hue)

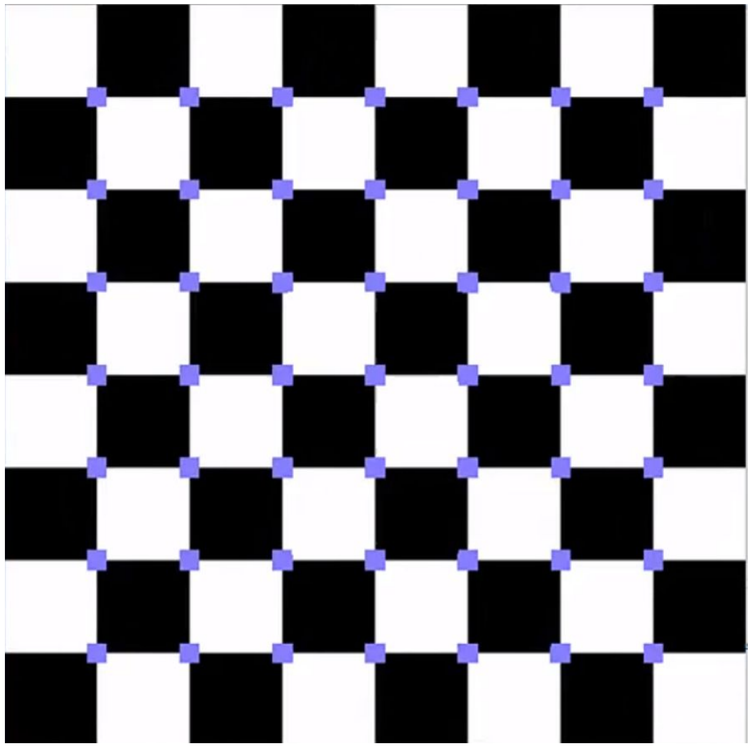
Corners

Detecting corners with ***cv2.cornerHarris***

cv2.cornerHarris(input image, block size, ksize, k)

- Input image - should be grayscale and float32 type.
- blockSize - the size of neighborhood considered for corner detection
- ksize - aperture parameter of Sobel derivative used.
- k - harris detector free parameter in the equation
- Output – array of corner locations (x,y)

Harris Corner Detection is an algorithm developed in 1998 for corner detection (http://www.bmva.org/bmvc/1988/avc-88-023.pdf) and works fairly well.



Corners

Detecting corners with ***cv2.goodFeaturesToTrack***

cv2.goodFeaturesToTrack(input image, maxCorners, qualityLevel, minDistance)

- Input Image - 8-bit or floating-point 32-bit, single-channel image.
- maxCorners – Maximum number of corners to return. If there are more corners than are found, the strongest of them is returned.
- qualityLevel – Parameter characterizing the minimal accepted quality of image corners. The parameter value is multiplied by the best corner quality measure (smallest eigenvalue). The corners with the quality measure less than the product are rejected. For example, if the best corner has the quality measure = 1500, and the qualityLevel=0.01 , then all the corners with the quality - - measure less than 15 are rejected.
- minDistance – Minimum possible Euclidean distance between the returned corners.

Corners

Tolerant to:

- Rotations
- Translations
- Slight photometric changes (brightness, intensity, etc)

Intolerant to:

- Large changes in intensity or photometric changes
- Scaling (enlarging, shrinking)

Scale Invariant Feature Transformation (SIFT)

SIFT is widely used feature detector in computer vision, however is no longer freely available from OpenCV 3.0 onwards (because it was patented)

https://www.youtube.com/watch?v=oKAnOzIu66c

Other feature detection algorithms are:

- SURF (Also patented)
- FAST
- ORB
- BRIEF
- HoG (Histogram of Gradients)

ORB (Oriented FAST and Rotated BRIEF)

An efficient alternative to SIFT or SURF created by “OpenCV Labs”.

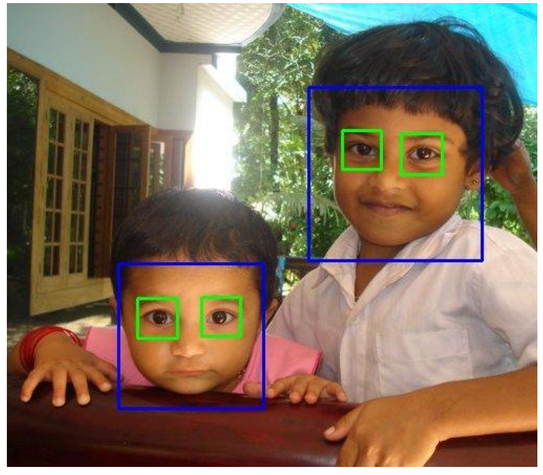
It is a good alternative to SIFT and SURF in computation cost, matching performance and mainly the patents.

Yes, SIFT and SURF are patented and you are supposed to pay them for its use. But ORB is not.

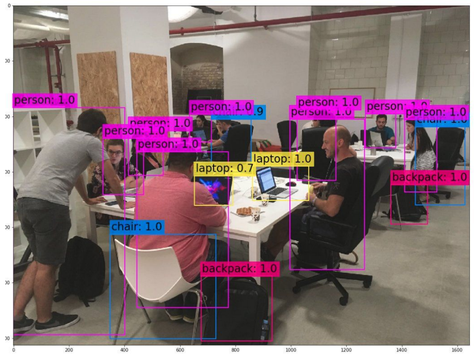
Feature Matching with ORB

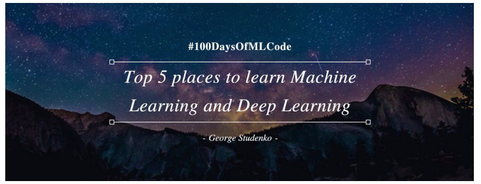
HAAR Cascade

- Object Detection using Haar feature-based cascade classifier.
- It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.
- OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one.
- OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc. Those XML files are stored in the opencv/data/haarcascades/ folder.
- You can find more trained models here: https://github.com/opencv/opencv/tree/master/data/haarcascades
- But you should be to find more trained models if you google them as well



About me...





George Studenko

Software Engineer and AI Enthusiast

georgestudenko

georgestudenko

george-studenko