Image Recognition using Python

1. Introduction

1.1 Overview

1.2 Purpose

2. Literature Survey

2.1 Existing Problem

2.2 Proposed Solution

3. Theoretical Analysis

3.1 Block Diagram

3.2 Hardware/Software Requirement

4. Result

5. Advantages and Disadvantages

7. Application

8. Conclusion

1 Introduction:

1.1 Overview

Image recognition, in the context of machine vision, is the ability of software to identify objects, places, people, writing and actions in images. Computers can use machine vision technologies in combination with a camera and artificial intelligence software to achieve image recognition.

Image recognition is used to perform a large number of machine-based visual tasks, such as labeling the content of images with meta-tags, performing image content search and guiding autonomous robots, self-driving cars and accident avoidance systems.

1.2 Purpose

While human and animal brains recognize objects with ease, computers have difficulty with the task. Software for image recognition requires deep learning. Performance is best on convolutional neural net processor as the specific task otherwise requires massive amounts of power for its compute-intensive nature. Image recognition algorithm can function by use of comparative 3D Models, appearances from different angles using edge detection or by components. Image recognition algorithms are often trained on millions of pre-labeled pictures with guided computer learning.

2 Literature Survey

2.1 Existing Problem

Disadvantages of Existing System

This was the main disadvantage of the existing system as it is not supported for all kinds of mobiles so later on, we developed a new version.

Becoming too dependent on technology. By uploading personal photos to the Google Assistant app, it could violate, ones privacy.

2.2 Proposed Problem

We implemented the project in the python package of numpy and OpenCV, we detect the object in the single and multiple object detection, we should overcome the disadvantage of Google Lens, to detect the single and multiple object detection using python numpy and OpenCv

Source Code:

Index.py: For Multi Object Detection

import cv2 as cv

import numpy as np

import os

os.chdir(os.path.dirname(os.path.abspath(\_\_file\_\_)))

coc\_img= cv.imread('coc.PNG',cv.IMREAD\_UNCHANGED)

coc\_cannon\_img= cv.imread('coc cannon.PNG',cv.IMREAD\_UNCHANGED)

found= cv.matchTemplate(coc\_img, coc\_cannon\_img,cv.TM\_CCOEFF\_NORMED)

print(found)

threshold=0.60

locations = np.where(found>=threshold)

locations = list(zip(\*locations[::-1]))

print(locations)

if locations:

print('Found match')

needle\_w = coc\_cannon\_img.shape[1]

needle\_h = coc\_cannon\_img.shape[0]

for loc in locations:

top\_left = loc

bottom\_right = (top\_left[0]+needle\_w,top\_left[1]+needle\_h)

cv.rectangle(coc\_img, top\_left, bottom\_right, color=(0,0,255),thickness=2,lineType=cv.LINE\_4)

cv.imshow('match', coc\_img)

cv.waitKey()

else:

print('Not Found')

main.py: For Single Object detection

#import lib

import cv2 as cv

import numpy as np

coc\_img= cv.imread('coc.PNG',cv.IMREAD\_UNCHANGED) #Cropped image

coc\_cannon\_img= cv.imread('coc cannon.PNG',cv.IMREAD\_UNCHANGED) #original image

found= cv.matchTemplate(coc\_img, coc\_cannon\_img,cv.TM\_CCOEFF\_NORMED) #match template function used to match 2 pic

min\_val, max\_val, min\_loc, max\_loc = cv.minMaxLoc(found)#for perfect match we use minmax function

print('best top left match:%s' %str(max\_loc))

print('best match confidence: %s' % max\_val)

threshold=0.8#threshold to take perfect match

if max\_val >= threshold:

print('found')

cannon\_w = coc\_cannon\_img.shape[1] #with for bottom right

cannon\_h = coc\_cannon\_img.shape[0] #height for bottom right

top\_left=max\_loc

bottom\_right= (top\_left[0]+ cannon\_w,top\_left[1]+cannon\_h)

#this line to make box search part of whole img

cv.rectangle(coc\_img,top\_left,bottom\_right,color=(255,0,0),thickness=2,lineType=cv.LINE\_4)

#imshow function used to display the result image

cv.imshow('RESULT',coc\_img)

#waitkey is used to hold the pic ,until we press some key

cv.waitKey()

else:

print('NOT FOUND')

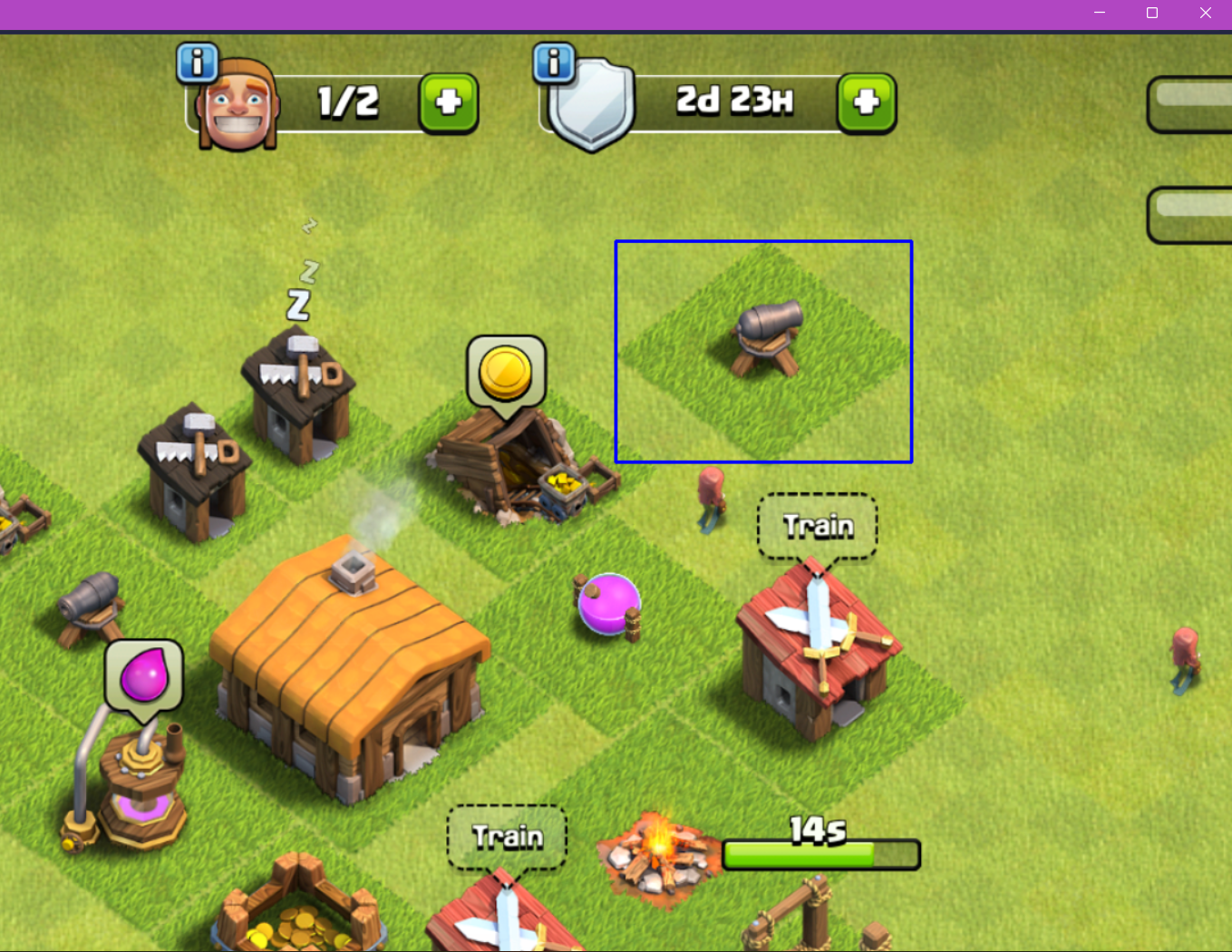
3. Theoretical Analysis:

3.1 Block Diagram

3.2 Hardware/Software Requirement

Anaconda, Visual studio code, python, openCV, numpy, pandas

4. Result: Screenshot for Image recognition





5. Advantages and Disadvantages

Advantages:

1. remove noises.
2. Correct image density and contrast.
3. Helps to easily store and retrieve in computers.
4. Image can be made available in any desired formats like black and white, negative image,

Disadvantage:

1) initial cost is high depending upon the system used.

2) once the system is damaged the image will be lost.

1. Applications

Improving Augmented Reality Gaming and Applications.

Assisting in the Educational System.

Optimizing Medical Imagery.

Boosting Driverless Car Technology.

Predicting Consumerism Behavior.

Giving Machines a Vision.

Iris Recognition Improvement.

1. Conclusion

We implemented the project only object only in the static image that ca only recognize, we can’t implement with the live images that is GIF imaged, We should implemented only for theft identification in the CCTV System and IOT for Agriculture to find the defective crops in cultivation, We will soon create and update my project to the global standard and dynamic images and that is benefit for all domains.