International Journal of Future Generation Communication and Networking
Vol. 13, No. 2s, (2020), pp. 1235–1239

Implementing Object Detection System in Image

Mona Mulchandani[1], Priyanka Dudhe[2], Rameshwari Patil[3]

Assistance Professor Computer Science & Engineering Department, JIT Nagpur, India[1]
Assistance Professor, Computer Science & Engineering Department, JIT Nagpur, India[2]
M.Tech Student, Computer Science & Engineering Department, JIT Nagpur, India[3]

Abstract

Matrix matching involves comparing an image to a stored glyph on a pixel-by-pixel basis; it is also known as "pattern matching", "pattern recognition", or "image correlation". This relies on the input glyph being correctly isolated from the rest of the image, and on the stored glyph being in a similar font and at the same scale. This technique works best with typewritten text and does not work well when new fonts are encountered. This is the technique the early physical photocell-based OCR implemented, rather directly.s. The comments are drawn based on the studied literature and key issues are also identified relevant to the object detection. Information about the source codes and online datasets is provided to facilitate the new researcher in object detection area. An idea about the possible solution for the multi class object detection is also presented. This paper is suitable for the researchers who are the beginners in this domain.

Keywords— *Object detection, Localization, Categorization, Object recognition.*

INTRODUCTION

A photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image for example from a television broadcast[1]. It is widely used as a form of information entry from printed paper data records, whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation[2]. It is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine[3] processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining[5]. OCR is a field of research in pattern recognition, artificial intelligence and computer vision[6].

LITERATURE REVIEW

Recognition of Latin-script, typewritten text is still not 100% accurate even where clear imaging is available. One study based on recognition of 19th- and early 20th-century newspaper pages concluded that character-by-character OCR accuracy for commercial OCR software varied from 81% to 99%; total accuracy can be achieved by human review or Data Dictionary Authentication[1]. Other areas—including recognition of hand printing, cursive handwriting, and printed text in other scripts (especially those East Asian language characters which have many strokes for a single character)—are still the subject of active research. The MNIST database is commonly used for testing systems' ability to recognise handwritten digits. .[3] The object template block then makes use of the learning's that were done previously to represent the objects with various representations like histogram representation, random forest representation, etc. [4]Whereas on the other hand, learning through validation block does not require any sort of training as they are validated beforehand. Hence after preprocessing the image, directly template matching is done which produces the features of an object in the image[5]. The main purpose of the testing phase is to decide whether an object is present in the image that is given to the system as input and if yes then to which object class does it belongs to. [6] Here the image is searched for an object by various searching techniques like the sliding window

ISSN: 2233-7857 IJFGCN

technique, and according to the output of the searching mechanism, a decision is made on the object class.

MOTIVATION

Object Detection has been a well studied subject for decades since it arises in many practical scenarios of modern marketing and advertising. Object Detection aimed to enable computers to detect the object in an image without human intervention. Objects are key elements for companies and play essential role in the industry and commerce. Different objects may have similar layout but slight difference in spatial disposition of the graphics elements, difference in orientation, size and shape. Objects of different firms exist in the database. The object exit in text form, graphic form or in both i.e. hybrid form. Therefore it is necessary to extract the feature object of the object image as well as the test image so that it can identify the text portion and graphic portion of the object properly.

IMPLEMENTATION

The operations of the network implementation in this project can be summarized by the following steps:

- **Training phase**
  - Analyze image for characters
  - Convert symbols to pixel matrices
  - Retrieve corresponding desired output character and convert to Unicode
  - Lineraize matrix and feed to network
  - Compute output
  - Compare output with desired output Unicode value and compute error
  - Adjust weights accordingly and repeat process until preset number of iterations
- **Testing phase**
  - Analyze image for characters
  - Convert symbols to pixel matrices
  - Compute output
  - Display character representation of the Unicode output

Essential components of the implementation are:

- Formation of the network and weight initialization routine
- Pixel analysis of images for symbol detection
- Loading routines for training input images and corresponding desired output characters in special files named character trainer sets (*.cts)
- Loading and saving routines for trained network (weight values)
- Character to binary Unicode and vice versa conversion routines
- Error, output and weight calculation routines

Algorithm:

1. start at the first x and first y pixel of the image pixel(0,0), Set number of lines to 0
2. scan up to the width of the image on the same y-component of the image
   a. if a black pixel is detected register y as top of the first line
   b. if not continue to the next pixel
   c. if no black pixel found up to the width increment y and reset x to scan the next horizontal line
3. start at the top of the line found and first x-component pixel(0,line_top)

ISSN: 2233-7857 IJFGCN

4. scan up to the width of the image on the same y-component of the image
    a. if no black pixel is detected register y-1 as bottom of the first line. Increment number of lines
    b. if a black pixel is detected increment y and reset x to scan the next horizontal line
5. start below the bottom of the last line found and repeat steps 1-4 to detect subsequent lines
6. If bottom of image (image height) is reached stop.

ii. Detecting Individual symbols

Detection of individual symbols involves scanning character lines for orthogonally separable images composed of black pixels.

Algorithm:

1. start at the first character line top and first x-component
2. scan up to image width on the same y-component
    a. if black pixel is detected register y as top of the first line
    b. if not continue to the next pixel
3. start at the top of the character found and first x-component, pixel(0,character_top)
4. scan up to the line bottom on the same x-component
    a. if black pixel found register x as the left of the symbol
    b. if not continue to the next pixel
    c. if no black pixels are found increment x and reset y to scan the next vertical line
5. start at the left of the symbol found and top of the current line, pixel(character_left, line_top)
6. scan up to the width of the image on the same x-component
    a. if no black characters are found register x-1 as right of the symbol
    b. if a black pixel is found increment x and reset y to scan the next vertical line
7. start at the bottom of the current line and left of the symbol, pixel(character_left,line_bottom)
8. scan up to the right of the character on the same y-component
    a. if a black pixel is found register y as the bottom of the character
    b. if no black pixels are found decrement y and reset x to scan the next vertical line

Architecture Of Object Detection

The salient feature, Now, there are various methods available for detection.

1) Tracing on trace paper using paper and pencil.
2) Eight Neighbors algorithm
3) Scanning top to bottom and then bottom to top and considering only pixels.

Here, we have opted for the first option. The image is used to trace out the outline on a trace paper using a pencil. This is then fed as input to mouse. This method is easy to carry out, but it lacks efficiency in certain terms:

a) The trace is done using free hand, which is never very perfect.
b) The time complexity of overall process increases, as we have to wait for scanning every time we have a new image.

However, this method had been successfully implemented to differentiate various faces inspite of its limitations.

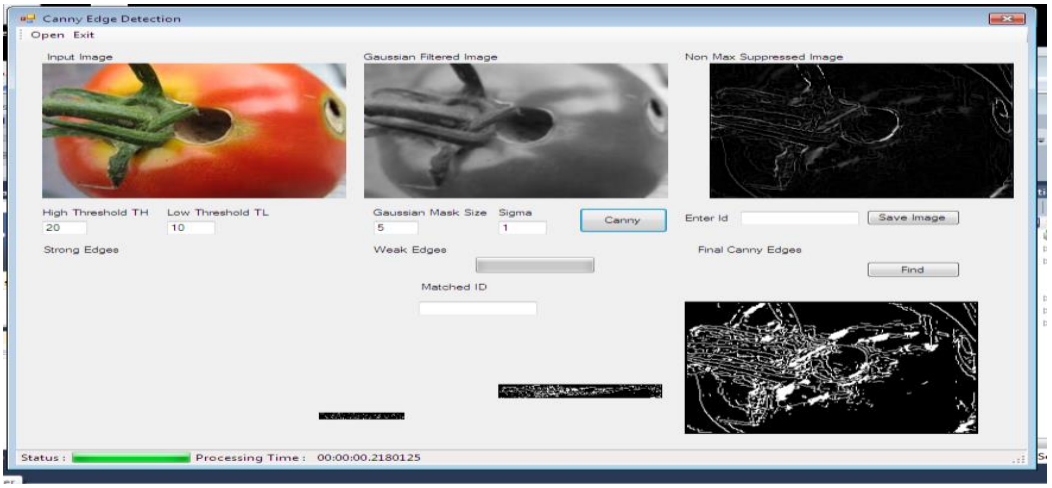
To overcome the limitations of above method, we have gone to develop a software for detection.

Filtering:-

The output obtained after applying detection is not exactly as per our need. It contains the as a part of the whole outline obtained, and not merely the side profile. So, now we have to extract profile from

ISSN: 2233-7857 IJFGCN

this image. This can be achieved by filtering out the unwanted pixels. To achieve this we can first find those pixels, which are single and scattered using 8-neighbouring algorithm. All the 8 neighbors of each pixel are found out and those with no neighbors are deleted. Similarly, we can omit the outside rectangle obtained by negating certain portions from ends.



RESULT

Based on experimental results, it is concluded that the accuracy of SURF algorithm is higher than SIFT and HOG.

Process	Percentage Accuracy
SIFT	56.66%
HOG	76.66%
SURF	81.66%

Table 1. Percentage accuracy by Feature Matching

We will be able to determine the object properly and know the information related to it using internet. The second phase of project will implement the process of object detection in an image. The process involve binarization, feature Extraction using SURF algorithm, and finally matching of features using CDS Context Dependent Similarity algorithm. Execution will be performed using C# in Microsoft Visual Studio. Finally the project will be able to detect the object in an image.

CONCLUSION

In the segmentation stage, an image of sequence of characters is decomposed into sub-images of individual character. The pre-processed input image is segmented into isolated characters by assigning a number to each character using a labelling process. This labelling provides information about number of characters in the image It will to detect the object so dat we can be able recognize and know about the object.

ISSN: 2233-7857 IJFGCN

REFERENCES

[1] Chinmoy Biswas, Jaydeep Mukherjee *Object Recognition Technique using Sift Descriptor, Surf Descriptor and Hog Descriptor*, , International Journal of Computer Applications (0975 – 8887) Volume 117 – No. 22, May 2015.

.[2] MeeraSambhajiSawalkar, MrunalineePatole *Object Detection and Recognition from the Images as Well as Video* , International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Impact Factor (2012): 3.358.

.[3] KhushbooKhurana, ReetuAwasthi *Techniques for Object Recognition in Images and Multi-Object Detection*,International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 4, April 2013.

[4] Remya Ramachandran, Andrews Jose *Object Matching And Recognition System Using Surf* , International Journal of Research in Computer and Communication Technology, Vol 3, Issue 9, September – 2014.

[5] P M Panchal, S R Panchal, *S K Shah, A Comparison of SIFT and SURF*, International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 2, April 2013.

[6] D. G. Lowe , *Object recognition from local scale-invariant features* , in Proc. of the International Conference on Computer Vision, Corfu, 1999.

[7] H. Bay, T. Tuytelaars, and L. J. V. Gool *Surf: Speeded up robust features , in ECCV* (1) (A. Leonardis, H. Bischof, and A. Pinz, eds.), vol. 3951 of Lecture Notes in Computer Science, pp. 404_417, Springer, 2006.

ISSN: 2233-7857 IJFGCN