System for Optical Character Recognition Using Intelligent Template Matching

Swanand Joshi
Department of Information and Technology
Pune Institute of Computer and Technology
Pune, India

***Abstract*— Template matching techniques are one of the simplest and oldest techniques in optical character recognition (OCR). The proposed system uses template matching for character recognition. The proposed system does not require an entire input template to be matched but only the critical part of the input template. The proposed system does not require any mathematical formulae for selection of the template as well as for the actual comparison of templates. For any N by N size template a standard template matching needs N multiplied by N comparisons but the proposed system requires N multiplied by 1.5 comparisons at the maximum. The system comprises of a counting block which determine the count of the input template which is nothing but a matrix with binary values. The system further includes classification block which classifies the given input. A target character repository provides a sequence of matching functions to select and recognize the correct character. The matching function block consists of all the matching functions that require simple binary comparison and by doing so avoid complicated mathematical computations. The performance system is measured by the number of comparisons required for each character input template for accurate selection and final recognition.**

Keywords—OCR, template matching, OCR systems.

I. INTRODUCTION

Template matching for optical character recognition is a technique where input templates are compared with a set of system templates to recognize the character. The input template is kept on the system template and the difference between the templates is calculated. The system template which gives minimum difference is classified as the correct character for that input. These techniques are sensitive to noise as they give erroneous results in noisy templates. The main disadvantage of template matching is that it requires a large number of comparisons. The primary objective of the system proposed in this paper is to reduce the number of comparisons for character recognition. The selection of correct template also requires a lot of comparisons and mathematical modeling. The selection comparisons are also reduced as the characters are classified and template selection process is made easier in this system.

II. SYSTEM ARCHITECTURE

The proposed system aims to reduce the number of comparisons in template matching. The system classifies all the characters into 4 classes on the basis of count which reduces the overhead of comparing the input template with all other templates.

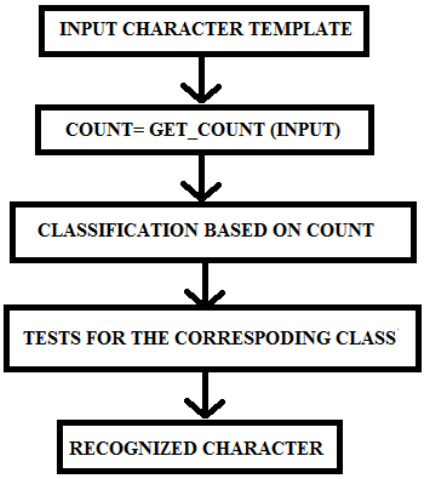

Fig. 1 System Architecture

IJERT

III. ASSUMPTIONS AND TESTING FUNCTIONS

A. Assumptions

The proposed system uses following assumptions and testing functions for Optical Character Recognition:

- The template size is 8 x 8 with 0 as low entropy value and 1 as high entropy value.
- System templates are generated according to the standard 'Times New Roman' font with size 160.
- The template is considered as a Cartesian co-ordinate system with standard four quadrants I to IV.
- The system employs divide and conquer strategy wherein the quadrants are divided into further four quadrants. The notation for these divided quadrants can is (main quadrant. Sub quadrant) for example if 1st quadrant is divided, the notation will be I.I, I.II, I.III and I.IV.
- The count of the template is obtained by a get_count() function which applies a simple matrix counting technique. The function returns a count value indicating number of 1's present.
- The algorithm uses four testing functions for comparisons. These are scan_line (), scan_column(), center_cluster(), region_match().

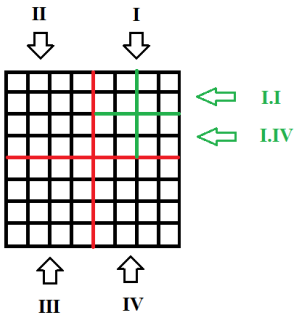

Fig. 2 Template and quadrants

B. *Testing Functions*

- The scan_line function calculates the no of 1's present in the given horizontal line. It takes line_no. as an input and returns the count of number 1's found in the horizontal line in the template. The no of comparisons performed by the function are 8.
- The scan_column function calculates the no of 1's present in the given vertical column. It takes column_no. as an input and returns the count of number 1's found in the vertical column in the template. The no of comparisons performed by the function are 8.
- The center_cluster function calculates the no of 1's present in the four locations around the center. It returns the count of number of 1's found around the center. The no of comparisons performed by the function are 4.
- The region_match function calculates the number of 1's present in the region specified. The function takes quadrant or a sub-quadrant as an input and returns the count of number of 1's found in the region. The no of comparisons performed by the function are m^2 where m is the size of the region.

IV. SYSTEM TEMPLATES AND CLASSES

A. *System Templates*

A standard template is a matrix of size 8 x 8. Templates are created for every character 'A' through 'Z' and are saved in system as standard system templates. The sample system template for 'A' is as shown in the figure. The white boxes in the figure represent 0 the grey boxes represent 1.

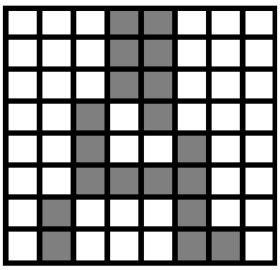

Fig. 3 Sample System Template

B. *Classes*

Get_count () function is used to calculate the count of the template. The number of grey boxes is tantamount to the count of the template. The count of the above system template for 'A' is 19.

The classification of the characters is done on the basis of count. This will help the recognition module to eliminate some decisions straight away and thus reducing the comparisons. The following table gives the character classes with their counts.

TABLE I. CHARACTER CLASSIFICATION

CLASS	RANGE OF COUNT	CHARACTERS
1	Less than 15	F (13), I (12), J (13), L (11), T (12), Y (12)
2	Between 15 to 17	C (16), K (17), P (15), V (15), X (17)
3	Between 18 to 22	A (19), D (20), E (19), G (21), H (19), O (20), R (19), S (18), U (18), Z (18)
4	Greater than 22	B (24), M (28), N (24), Q (24), W (26)

Once the given input template is classified into one of the classes, actual recognition tests are performed class-wise to obtain the result. The class-wise tests are explained in the subsequent topic.

V. CLASS-WISE TESTS

Once the input template is classified, the test sequence for the particular class is employed on the template for character recognition.

A. *Class 1 Tests*

For illustrative purpose, the class 1 test sequence is explained in detail. Other classes consist of different testing sequences for recognition. Class 1 has all the characters having count less than 15. The characters are 'F', 'I', 'J', 'L', 'T', 'Y'. If the input is classified into class 1 then following sequence of tests are conducted in order to recognize the input character.

1) Test 1.1- Test for 'L'

The test includes region_match() function as the testing function. The regions to be matched are I.II and I.III. If the testing function returns 0 on both the regions then the input character is 'L' otherwise input is sent to next test.

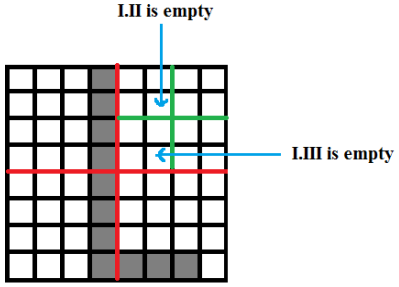

Fig. 4 Test 1.1

IJERT

2) Test 1.2- Test for 'I'

The test includes region_match() function as the testing function. The region to be matched is IV.III. If the testing function returns a non-zero value then the input character is 'I' otherwise input is sent to next test.

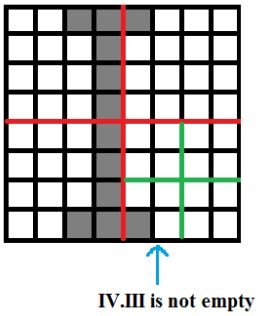

Fig. 5 Test 1.2

3) Test 1.3- Test for 'J'

The test includes region_match() function as the testing function. The region to be matched is III.III. If the testing function returns a non-zero value then the input character is 'J' otherwise input is sent to next test.

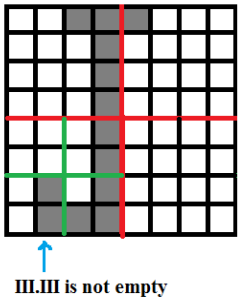

Fig. 6 Test 1.3

4) Test 1.4- Test for 'Y'

The test includes scan_line() function as the testing function. The line to be scanned is the 1st horizontal line. If the testing function returns value equal to 2 then the input character is 'Y' otherwise input is sent to next test.

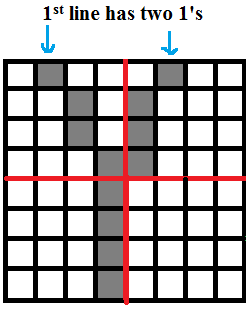

Fig 7 Test 1.4

5) Test 1.5- Test for 'F' and 'T'

The test includes scan_column() function as the testing function. The column to be scanned is determined as follows. The first horizontal line is scanned. When the first 1 is found, the column in which it is found is considered for testing. If the testing function returns value equal to 1 then the input character is 'T' otherwise it is 'F'.

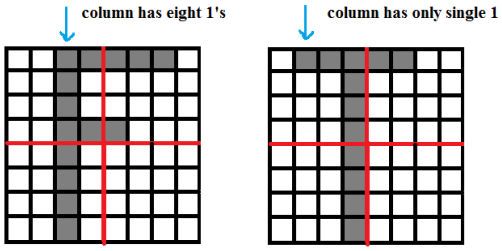

Fig 8 Test 1.5

B. Class 2 Tests

Class 2 has all the characters having count from 15 to 17. The characters are 'C', 'K', 'P', 'V', 'X'. If the input is classified into class 2 then following sequence of tests are conducted in order to recognize the input character.

1) Test 2.1

The test includes center_cluster() function as the testing function. If the function returns value equal to 0 then the input is either 'C' or 'V' and input is sent to test 2.2 else the input is sent to test 2.3.

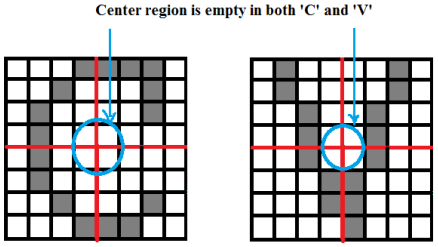

Fig. 9 Test 2.1

2) Test 2.2- Test for 'C' and 'V'

The test includes region_match() function as the testing function. The region to be matched is II.II. If the testing function returns value equals to 0 then the input character is 'C' otherwise it is 'V'.

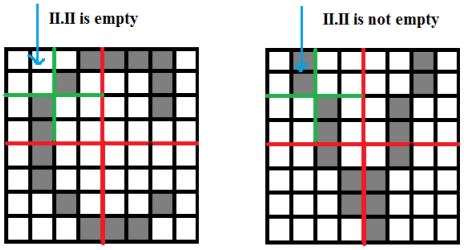

Fig. 10 Test 2.2

3) Test 2.3- Test for 'X'

The test includes region_match() function as the testing function. The region to be matched is III.III. If the testing

IJERT

function returns value greater than 0 then the input character is 'X' otherwise input is sent to next test.

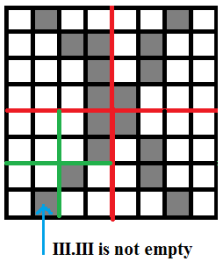

Fig. 11 Test 2.3

4) Test 2.4- Test for 'K' and 'P'

The test includes region_match() function as the testing function. The region to be matched is IV.III. If the testing function returns value equals to 0 then the input character is 'P' otherwise it is 'K'.

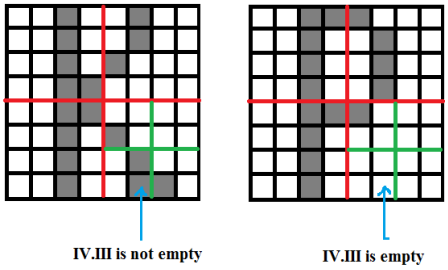

Fig. 12 Test 2.4

C. Class 3 Tests

Class 3 has all the characters having count from 18 to 22. The characters are 'A', 'D', 'E', 'G', 'H', 'O', 'R', 'S', 'U', 'Z'. If the input is classified into class 3 then following sequence of tests are conducted in order to recognize the input character.

1) Test 3.1

The test includes center_cluster() function as the testing function. If the function returns value equal to 0 then the input can be 'D', 'O', 'U', 'G' and input is sent to test 3.2 else the input is sent to test 3.5.

2) Test 3.2- Test for 'U'

The test includes scan_line() function as the testing function. The line to be scanned is the 1st horizontal line. If the testing function returns value equal to 2 then the input character is 'U' otherwise input is sent to test 3.3.

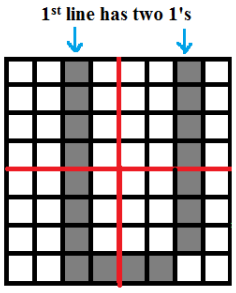

Fig. 13 Test 3.2

3) Test 3.3- Test for 'D'

The test includes scan_column() function as the testing function. The column to be scanned is determined as follows. The first horizontal line is scanned. When the first 1 is found, the column in which it is found is considered for testing. If the testing function returns value equal to 8 then the input character is 'D' otherwise the input is sent to test 3.5.

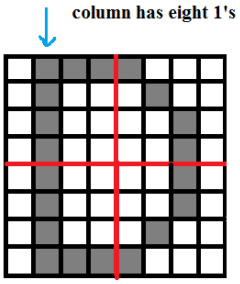

Fig. 14 Test 3.3

4) Test 3.4- Test for 'G' and 'O'

The test includes region_match() function as the testing function. The region to be matched is IV.IV. If the testing function returns 0 then the input character is 'O' otherwise it is 'G'.

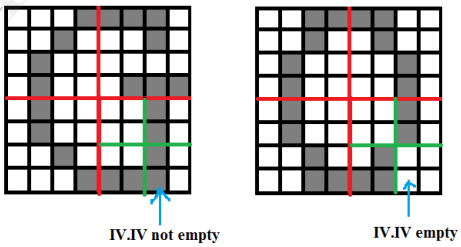

Fig. 15 Test 3.4

5) Test 3.5

The test includes region_match() function as the testing function. Region to be matched is II.II. If the function returns value equal more than 0 then the input can be 'Z' or 'H' and input is sent to test 3.6 else the input is sent to test 3.7.

6) Test 3.6- Test for 'Z' and 'H'

The test includes scan_column() function as the testing function. The column to be scanned is determined as follows. The first horizontal line is scanned. When the first 1 is found, the column in which it is found is considered for testing. If the testing function returns value equal to 8 then the input character is 'H' otherwise it is 'Z'.

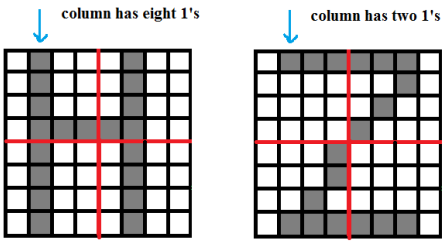

Fig. 16 Test 3.6

IJERT

7) Test 3.7- Test for 'A'

The test includes region_match() function as the testing function. The region to be matched is III.III. If the testing function returns value more than 0 then the input character is 'A' otherwise the input is sent to test 3.8.

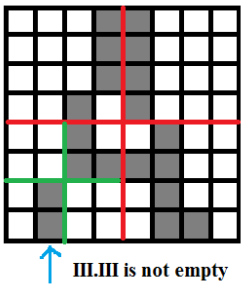

Fig. 17 Test 3.7

8) Test 3.8- Test for 'S'

The test includes scan_column() function as the testing function. The column to be scanned is determined as follows. The first horizontal line is scanned. When the first 1 is found, the column in which it is found is considered for testing. If the testing function returns value less than 8 then the input character is 'S' otherwise the input is sent to test 3.9.

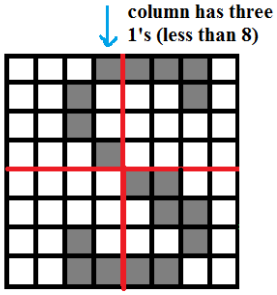

Fig. 18 Test 3.8

9) Test 3.9- Test for 'E' and 'R'

The test includes scan_line() function as the testing function. The line to be scanned is the last horizontal line. If the testing function returns value less than 4 then the input character is 'R' otherwise it is 'E'.

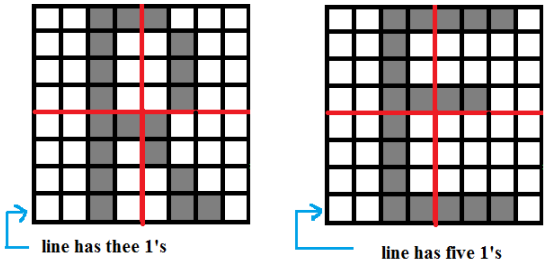

Fig. 19 Test 3.9

D. Class 4 Tests

Class 4 has all the characters having count greater than 22. The characters are 'B', 'M', 'N', 'Q', 'W'. If the input is classified into class 4 then following sequence of tests are conducted in order to recognize the input character.

1) Test 4.1- Test for 'Q'

The test includes center_cluster() function as the testing function. If the function returns value equals to 0 then the input is 'Q' else the input is sent to the next test.

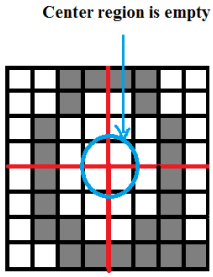

Fig. 20 Test 4.1

2) Test 4.2

The test includes region_match() function as the testing function. The region to be matched is III.III. If the testing function returns 0 then the input character is 'B' or 'W' and the input ois sent to test 4.3 otherwise the input is 'M' or 'N' and the input is sent to test 4.4.

3) Test 4.3- Test for 'B' and 'W'

The test includes region_match() function as the testing function. The region to be matched is II.II. If the testing function returns 0 then the input character is 'B' otherwise it is 'W'.

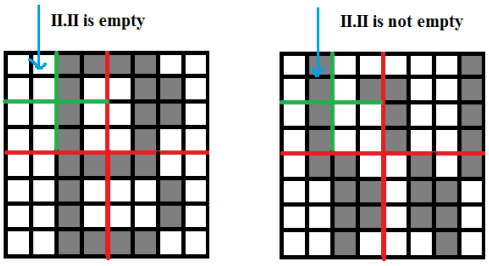

Fig. 21 Test 4.2

4) Test 4.3- Test for 'M' and 'N'

The test includes region_match() function as the testing function. The region to be matched is III.IV. If the testing function returns 0 then the input character is 'N' otherwise it is 'M'

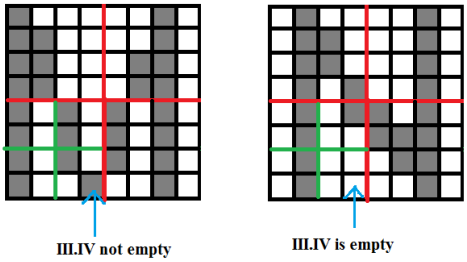

Fig. 22 Test 4.3

IJERT

VI. RESULTS

The following table shows the results of the proposed system. The number of comparisons required for every character template selection and recognition indicate the total number of comparisons required for a particular character.

TABLE II. COMPARISONS REQUIRED

CHARACTER	TEMPLATE SELECTION	TEMPLATE MATCHING	TOTAL
A	20	4	24
B	8	4	12
C	4	4	8
D	12	12	24
E	36	8	44
F	24	12	36
G	24	4	28
H	8	4	12
I	8	4	12
J	12	4	16
K	12	4	16
L	0	8	8
M	12	4	16
N	12	4	16
O	24	4	28
P	12	4	16
Q	0	4	4
R	36	8	44
S	24	12	36
T	24	12	36
U	4	8	12
V	4	4	8
W	8	4	12
X	8	4	12
Y	16	8	24
Z	8	12	20

From above table it is clear that out of 26 characters 21 need less than 30 comparisons for both selection and matching, 3 characters need 36 comparisons and only 2 characters need 44 comparisons.

Without using any feature vector or mathematical formulae, the algorithm reduces number of comparisons for template selection as well as matching.

CONCLUSIONS

- The proposed system is developed for an 8 x 8 template but it can be extended to any 2^n x 2^n template.
- The proposed system is developed for only capital 'A' to 'Z'. It can be extended to all the characters by adding more classes and tests.
- The distinction between the classes can become clearer if larger templates are used.
- The number of comparisons for every character is reduced to a large extent.
- A standard template matching requires n^2 comparisons for an n x n template but in this system the maximum comparisons required for any test are (3n/2).

IJERT

REFERENCES

[1] B. P. Gaikwad, R. R. Manza, G. R. Manza, "Video Scene Segementation to Separate Script", 3rd IEEE International Advance Computing Conference, 2013.

[2] Roberto Brunelli, Template matching techniques in computer vision: theory and practice, WILEY Publications, ISBN 978-0-470-51706-2, March 2009.

[3] Haojin Yang, Bernhard Quehl, Harald Sack, "Text Detection in Video images using Adaptive Edge Detection and stroke width verification", IEEE, 2012.