Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Haar Feature Analysis of Face-Non Face Images

Ravi Kant Parashar
Department of Computer Engineering, Poornima University,
Rajasthan, India

Abstract*— *Face detection is a challenging classification problem and involves a wide range of issues. People can quickly and accurately recognize faces in real life, but for a computer is a very tedious task. To detect the face accurately and rapidly within an image is the demand of time. In this paper, Haar features of Face and Non-Face images are analyzed. At first, an integral image of a binary image is computed having 19 x 19 pixel size. Then, it applies Haar feature extraction technique to extract Haar values of face and non-face images. Using this integral image, Haar features are calculated in a rapid manner. Multiple face and non face samples are taken from MIT standard database and extracted different features from theses samples. The feature extraction and integral image computation are performed on MATLAB 7 software. Features are calculated using the rectangular area calculation. Each Haar feature has a value that is calculated by taking the area of each rectangle, and then summing the results. To calculate the Haar feature value, an image having the size equal to frame size is taken as input. These distinctive features could be compressed into statistical model parameters which could be used as special property to classify different objects. Above all, this method has a certain theory and practical value.

Keywords*— *Face Detection, Haar Features, Integral Image, Classification, Face Recognition

I. INTRODUCTION

Images are classified based on the value of simple features in the face detection procedure. The high variability of the face class and complexity of the non - face class arises the need of efficient and fast feature extraction technique. It is advantageous to use the features rather than pixels directly as feature-based systems operate much faster than pixel based system. The other reason is that ad-hoc domain knowledge which is very hard to learn using a finite quantity of training data can be encoded using features.

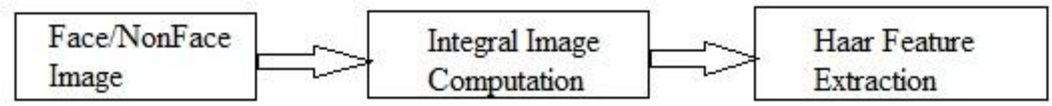

Fig. 1 Haar Feature Computation

In statistical model based training [3], multiple face and non-face samples are taken to extract different features from theses samples. These distinctive features are then compressed into statistical model parameters which are used as special property to classify different objects. By making adjustments in these parameters we can improve the accuracy of classification.

II. INTEGRAL IMAGE COMPUTATION

The integral image computation [2] is an effective way to evaluate the Haar features in a quick manner. Suppose, there is a gray scale image having some fixed size. A Pixel is the smallest element of an image. For a greyscale image, the pixel value is a single number that represents the brightness of the pixel. Each pixel has a value from 0 to 255. Integral image is computed for a given image calculating the sum of pixel values in a given image or a rectangular subset of the given image. To create an integral image, we need to create a Summed Area Table (SAT). In this table, at any point P(x,y) the value is the sum of all the pixel values above and to the left of this point, including the original pixel value of (x, y).

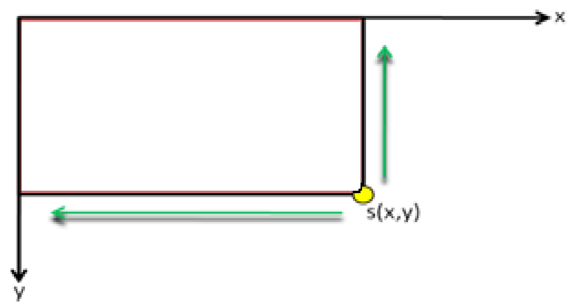

Fig. 2 The region represents the sum of pixels up to the position (x, y)

Let $i(x,y)$ is the original image, the integral image $ii(x,y)$ will be represented by

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y')$$

So it is really good that SAT is constructed in one pass over the original image. The integral value in the SAT at (x,y) is calculated by using the given formula

$$S(x,y) = i(x,y) + S(x-1,y) + S(x,y-1) - S(x-1,y-1)$$

Here $i(x,y)$ is the original pixel value from the image. Then add the values directly above this pixel$S(x,y-1)$, and directly left to this pixel $S(x-1,y)$. Finally, subtract the value directly top-left of $i(x,y)$, $S(x-1,y-1)$. This $S(x,y)$ is the sum of all the pixel values to the left and the above of this pixel and also the original pixel value of (x,y) itself. Using the integral image any rectangular sum can be computed in four array references. The difference between two rectangular sums can be computed in eight references. Therefore, double rectangular characteristic values consisted of two adjacent rectangles can be calculated by six array references. Similarly, the characteristic values of three rectangles can be calculated by eight array references, the characteristic values of four rectangles can be calculated by nine array references.

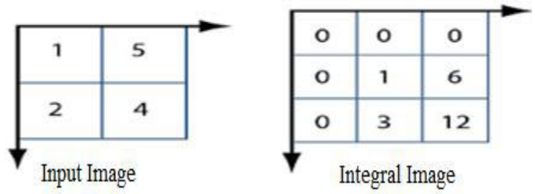

Fig. 3 Integral Image Calculation

Here,

$$i(x,y) = 4$$
$$s(x-1,y) = 1+2 = 3$$
$$s(x,y-1) = 1+5 = 6$$
$$s(x-1,y-1) = 1$$
$$s(x,y) = 4+3+6-1 = 12$$

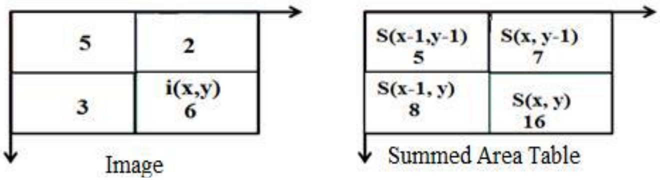

Fig. 4 Summed Area Table

Here,

i(*x*, *y*) □ □ 6
S(*x* □ 1,*y*) □ □ 8
S(*x*, *y* □ 1) □ □ 7
S(*x* □ 1,*y* □ 1) □ □ 5

Putting these values in the formula

$$S(x,y) = i(x,y) + S(x-1,y) + S(x,y-1) - S(x-1,y-1)$$

S(*x*, *y*) □ □ 6 □ □ 8 □ □ 7 □ □ 5 □ □ 16

Using the above methodology, a summed area table is filled with values. Now computation of sum of pixels in any subset of the original image can be done in constant time, in O(1) complexity.

In order to calculate the area contained by area D in an integral image, only four values are required. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A+B, at location 3 is A + C, and location 4 is A + B + C + D. The sum within D can be computed as

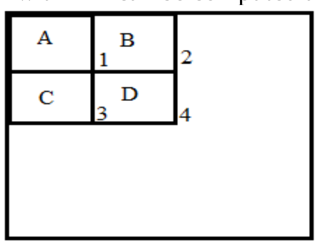

Fig. 5 The Sum of the Pixels

Parashar International Journal of Advanced Research in Computer Science and Software Engineering 6(5), May- 2016, pp. 295-299

From the previous example, the values in summed area table are as follows

$$s(A) = 1,$$
$$s(B) = 6,$$
$$s(C) = 3,$$
$$s(D) = 12$$
$$i(x', y') = s(A) + s(D) - s(B) - s(C)$$
$$i(x', y') = 1 + 12 - 6 - 3 = 4$$

Original image

1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1

Integral image of original image

1	2	3	4	5	6	7	8	9	10	11	12
2	4	6	8	10	12	14	16	18	20	22	24
3	6	9	12	15	18	21	24	27	30	33	36
4	8	12	16	20	24	28	32	36	40	44	48
5	10	15	20	25	30	35	40	45	50	55	60
6	12	18	24	30	36	42	48	54	60	66	72
7	14	21	28	35	42	49	56	63	70	77	84
8	16	24	32	40	48	56	64	72	80	88	96
9	18	27	36	45	54	63	72	81	90	99	108
10	20	30	40	50	60	70	80	90	100	110	120
11	22	33	44	55	66	77	88	99	110	121	132
12	24	36	48	60	72	84	96	108	120	132	144

Fig. 6 Integral Image

Here, an original image is shown which have pixel value 1 for all the pixels in the image. The corresponding integral image is presented on the right side that is computed according to the above mentioned steps.

III. HAAR FEATURES EXTRACTION

Haar-like feature [1] reflects the partial gray change of image by describing the gray level difference between the two adjacent rectangular regions.

$$F_{Haar} = E(R_{White}) - E(R_{Black})$$

To compute these Haar features very rapidly at many scales, integral image representation for image is computed. Haar-like feature could divide into an edge feature, line features and the center surround feature.

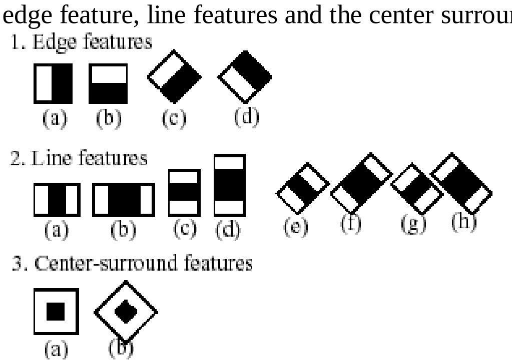

Fig. 7 Extended Set of Haar Features

1) ***Two-Rectangle Feature:*** It is the difference between the sum of the pixels within two rectangular regions. The regions are of the same size and shape and are horizontally or vertically adjacent.
2) ***Three-Rectangle Feature:*** It computes the sum within two outside rectangles subtracted from the sum in a center rectangle.
3) ***Four-Rectangle Feature:*** It computes the difference between diagonal pairs of rectangles.

The sum of pixels in white rectangles is subtracted from those in the black rectangles. These rectangles map over the image and tell if it is a face or non face.

Here Fig. 8(a) shows two-rectangle features, 8(b) shows three-rectangle features and 8(c) shows four-rectangle features.

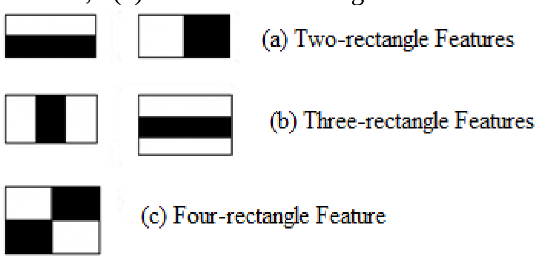

Fig. 8 Haar Features

Here, the main problem arises to handle such a huge amount of derived features. For a 20 x 20 pixel image size, if the above five Haar like features are used, the total number of features will be up to 51705.

Table I Total No. of Features against different Frame Size as Feature Type

Frame Size	Types of Feature	Total Number of Features
19 x 19	5 (Fig. 8a, 8b,8c)	40986
20 x 20	5 (Fig. 8a, 8b,8c)	51705
24 x 24	5 (Fig. 8a, 8b,8c)	114829
20 x 20	1 (Fig. 8a(i))	13851
19 x 19	1 (Fig. 8a(i))	11016

That's a large number and the time to calculate such a huge number of features is relatively large also. The improved extraction of Haar-like features could be applied by using size constraints or implementing any Haar feature reduction technique.

Features are extracted from sub windows of a sample image. The base size for a sub window is 19 by 19 pixels. Each of the five feature types are scaled and shifted across all possible combinations.

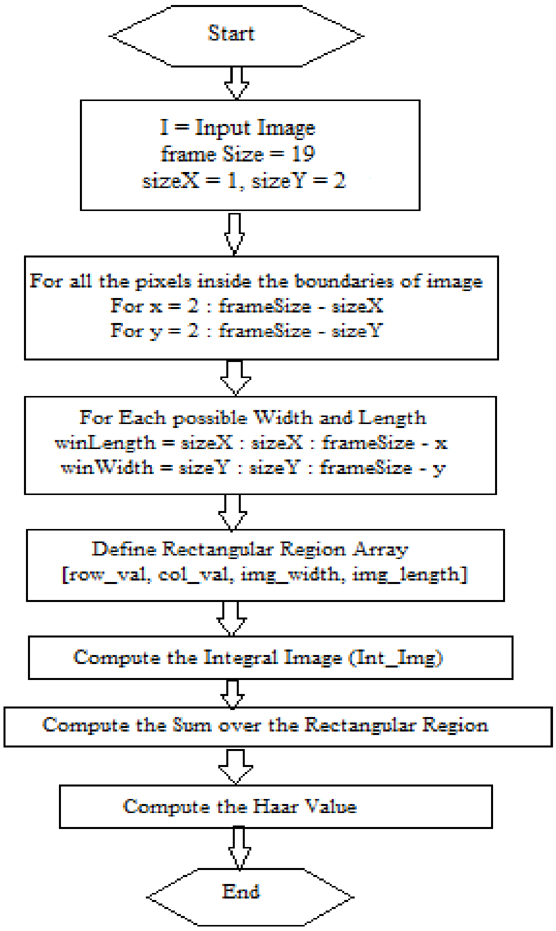

Fig. 9 Flow Chart: Haar Feature Computation

Let the face or non-face image have size (Frame Size): 8 x 8

The original image has these pixel values for corresponding pixels.

138	93	51	50	57	43	31	62
189	171	136	122	108	84	101	166
91	95	103	98	79	67	73	97
37	35	36	31	34	39	30	23
16	13	9	8	12	21	18	13
1	1	2	4	10	15	15	13
48	43	56	68	88	91	84	69
193	187	197	201	197	191	186	189

A haar-like feature is calculated using:

Feature Value = RectSum1 – RectSum2

Feature Value = ∑(Pixel in White Area) - ∑(Pixel in Black Area)

138	231	282	332	389	432	463	525
327	591	778	950	1115	1242	1374	1602
418	777	1067	1337	1581	1775	1980	2305
455	849	1175	1476	1754	1987	2222	2570
471	878	1213	1522	1812	2066	2319	2680
472	880	1217	1530	1830	2099	2367	2741
520	971	1364	1745	2133	2493	2845	3288
713	1351	1941	2523	3108	3659	4197	4829

Let calculate feature value for feature type 8 a(ii) on this image values.
Feature Value = ∑ (Pixel in White Area) – ∑ (Pixel in Black Area)
RectSum1 = ∑ (Pixel in White Area)
= 778 + 138 – (282 + 327) = 307
RectSum2 = ∑ (Pixel in Black Area)
= 282 + 1115 - (389 + 778) = 230
Feature Value = RectSum1 – RectSum2
= 307 – 230
= 77

Haar features extracted for each image leading to more complex design of any system. Therefore features were reduced through taking mean of 100, 200, 300, 400 and 500 values of Haar features of each image.

IV. CONCLUSION

With the successful extraction of Haar features, the next goal is to research the ability for gathering more precise details of the facial features. These details will be used to differentiate general human emotions and individual's expression. This research will be used in the field Face detection, human-computer interaction to analyze the emotions one exhibits while interacting with a user interface.

ACKNOWLEDGMENT

The author would like to thank Prof. (Dr.) Mahesh Bundele, Dean, Research & Development, Poornima University, Jaipur for encouragement and support to carry out the present work.

REFERENCES

[1] Lienhart R. and Maydt J. (2002), "An Extended Set of Haar-like Features For Rapid Object Detection," *Image Processing. 2002. Proceedings. 2002 International Conference on* , vol.1, pp.I-900,I-903.

[2] Viola, P. and Jones, M.J. (2004), Robust Real-Time Face Detection, *International Journal of Computer Vision*, Vol. 57(2), pp. 137-154.

[3] Yang, M.H.; Kriegman, D.J.; Ahuja, N. (2002), Detecting faces in images: a survey, *Pattern Analysis and Machine Intelligence. IEEE Transactions on*, vol.24, no.1, pp.34, 58.