Global Journal of Computer Science and Technology: F
Graphics & Vision
Volume 20 Issue 1 Version 1.0 Year 2020
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 2249-4596 & Print ISSN: 0975-5861

The Fast Integration of a Rotated Rectangle Applied to the Rotated Haar-like Features for Rotated Objects Detection

By Mohamed Oualla, Khalid Ounachad & Abdelalim Sadiq
Moulay Ismail University

Abstract- The Integral Image technique, used by Viola and Jones, is generally used to calculate the integral of a rectangular filter in an input picture. This filter is a rectilinear rectangle. We propose a method to integrate a rotated one by any angle of rotation inside an image based on the Bresenham algorithm of drawing a segment. We use some pixels – called key points - that forms the four segments of a rotated rectangle, to calculate its Integral Image. Our method focuses on three essential tasks; the first is to determine the rule for drawing a segment (SDR), the second is to identify all the key points of the rectangle r, and the third is to calculate the integral image. The speed of this method depends on the size and angle of rotation of the rectangle. To demonstrate the efficiency of our idea, we applied it to the rotated Haar-like features that we proposed in a later work [12], which had as objectives the improvement of the Viola and Jones algorithm to detect the rotated faces in a given image. We performed tests on more widespread databases of images, which showed that the application of this technique to rotated Haar-Like features improves the performance of object detectors, in general, and faces in particular.

Keywords: haar-like features, integral image, face detection, object detection, viola & jones algorithm, adaboost.

GJCST-F Classification: I.4.8

THEFASTINTEGRATIONOFAROTATEDRECTANGLEAPPLIEDTOTHEROTATEDHAARLIKEFEATURESFORROTATEDOBJECTSDETECTION

Strictly as per the compliance and regulations of:

The Fast Integration of a Rotated Rectangle Applied to the Rotated Haar-like Features for Rotated Objects Detection

Mohamed Oualla [α], Khalid Ounachad [σ] & Abdelalim Sadiq [ρ]

Abstract- The Integral Image technique, used by Viola and Jones, is generally used to calculate the integral of a rectangular filter in an input picture. This filter is a rectilinear rectangle. We propose a method to integrate a rotated one by any angle of rotation inside an image based on the Bresenham algorithm of drawing a segment. We use some pixels – called key points - that forms the four segments of a rotated rectangle, to calculate its Integral Image. Our method focuses on three essential tasks; the first is to determine the rule for drawing a segment (SDR), the second is to identify all the key points of the rectangle r, and the third is to calculate the integral image. The speed of this method depends on the size and angle of rotation of the rectangle. To demonstrate the efficiency of our idea, we applied it to the rotated Haar-like features that we proposed in a later work [12], which had as objectives the improvement of the Viola and Jones algorithm to detect the rotated faces in a given image. We performed tests on more widespread databases of images, which showed that the application of this technique to rotated Haar-Like features improves the performance of object detectors, in general, and faces in particular.

Keywords: haar-like features, integral image, face detection, object detection, viola & jones algorithm, adaboost.

I. Introduction

Adaboost learning algorithm, proposed by [1], and adapted by Viola and Jones for object detection [2][3], is one of the most widely used algorithms in detection. This algorithm, based on Haar-like features (Fig. 2), achieves a high detection rate and we use it widely in face and pedestrian detection [4] [5] [6].

Haar-like feature classifiers trained by Adaboost algorithm are often incapable of finding rotated objects (Fig. 1). Many experts have proposed several solutions to fill this problem; Viola and Jones [2] [3] [4] used rotated positive examples during training, but this approach may give some inaccurate classifier. Another process adopted by several researchers [5] [7] [9] [10] consists of training several classifiers which specialize in certain angle intervals, this approach provides classifiers with appropriate accuracy and efficiency, but it makes the training computationally more expensive. In [7][8] the image is physically or virtually rotated until the edges of the Haar-Like feature are aligned vertically or horizontally, this approach makes the detection computationally more expensive and also a loss of information when a set of pixels will be outside the region of interest.

Fig. 1: Face detection with the latest method [12]; The algorithm is unable to detect a few rotated faces.

In [12], we gave a solution allowing, firstly, to define, dynamically, a set of rotated Haar-Like Features by more than 50 angles formally expressed by $arctang$ $(A\ /\ B)$ such that A and B are two integers. And secondly, to calculate, approximatively, the rotated integral image of these rectangles to preserve the two advantages (speed and simplicity) for which Viola and Jones invented this technique for the first time. This method has shown a disadvantage, a little annoying, which consists of a false alarm rate, which is relatively high.

In this paper, we propose a method that calculates the real Integral Image of a rotated rectangle based on the set of pixels, called key-points, forming each of its four segments.

Author α: Software Engineering \& Information Systems Engineering Team, Computer sciences department, Faculty of Sciences and Technology (FST), Moulay Ismail University, Errachidia, Morocco. e-mail: mohamedoualla76@gmail.com
Author σ ρ: Information System and Multimedia team, Faculty of Sciences (FS), Ibn Toufail University, Kénitra, Morocco.

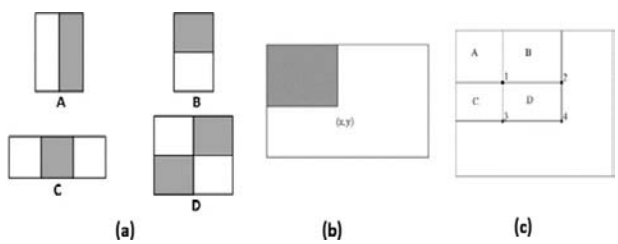


Fig. 2: (a) Haar-Like basic features, used by Viola-Jones in [2]. (b) Integral Image (c) Calculation of the sum of the rectangle D with the Integral Image.

Indeed, we determine these points by adapting the algorithm for drawing a segment [13].

We organize this paper as follows: Section II presents some work done in the literature. In section III we will propose our technique concerning the quickly integration of a rotated rectangle. Section IV explores the results obtained by our two experiments carried out for the case of face detection. In section V, we will give a summary of our work, some conclusions, and perspectives.

II. Related Work

The technique of the Integral Image, strongly associated with the famous algorithm of Viola and Jones [2], is a technique that goes back to the year 1984 when Crow [16] introduced it, for the first time, in computer graphics. Its use in computer vision began in 2001 when Viola and Jones used it to calculate the integral of the rectangular filters called HaarLike features, shown in Fig.2 (a), considered as one of the pillars of their real-time face detection algorithm. Therefore, most researchers have widely used and developed this technique to solve the problems related to the detection of objects, including faces detection [2][7], pedestrian detection [4][6], etc.

Let i be an image of dimension $M \times N$, $i(x, y)$ is the intensity of the pixel (x, y) of the image i. The *Integral Image* is another image ii such that for each pixel (x, y): (see Fig. 2(b)).

$$ii(x,y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x',y') \qquad (1)$$

Therefore, the integration of each axis-aligned (parallel to the axes of the image coordinate system) rectangle $R = [(x, y), w, h]$ is calculated only in four references: (see Fig. 2(c))

$$ii(R) = ii(x + w, y + h) - ii(x, y + h) - ii(x + w, y) + ii(x, y) \qquad (2)$$

As (x, y) is the upper left corner of R, w and h represent, respectively, its width and height. In the rest of this article, this type of rectangle is called a *normal rectangle*.

This technique was then the subject of several improvements, in order to detect rotated objects in an image. Indeed, several works have chosen to feed the original set of *HaarLike features*, initially proposed by Viola and Jones (Fig.2 (a)), by others which are rotated by a variety of angles and offering an ease of calculating their integration. In particular, Lienhart et al. [5] who introduced a 45° tilted integral image. Then, Barczak et al. [7] and Du et al. [8] retained the same technique of Viola and Jones to incorporate rotated features by 26.57° and 63.5°. Then Barczak and Mossom [9] tried to generalize this technique for angles having a tangent in the form of a rational number $1/n$ or $n/1$. Another work is that of Ramirez et al. [10], who introduced the asymmetric Haar Features. Pham et al. [14] have developed a technique called Polygonal integration to divide a polygon into a set of axis-aligned rectangles (*normal rectangles*), then determine its integration according to those of these rectangles. Doretto et al. [15] gave an extension of formula 1 to compute the integral of a domain $D \subset \mathbb{R}^n$, which consists of a finite unification of axis-aligned rectangles.

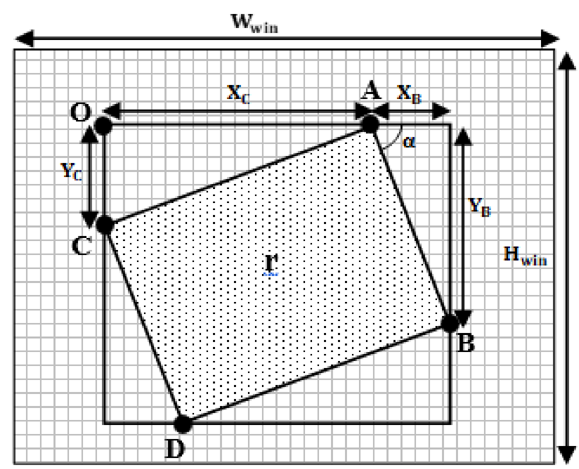


Fig. 3: Representation of a rotated rectangle.

III. Fast Integration of the Rotated Haar-Like

a) Rotated Haar-Like features

A rotated Haar-Like feature is a rectangle rotated by a given angle. The rotated Haar-like features are based on the normal ones presented in [2]. In a scan window named Win - with $Hwin$ as height and $Wwin$ as width - a rotated rectangle is defined by its vertex A (xA, yA), its rotation angle α and its rectangle encapsulating $R[o(xA - xC, yA), w = XC + XB, h = YB + YC]$ as shown in Fig. 3.

Formally, a rotated rectangle is defined by a six-element vector as shown by the following formula:

$$r_A^t = (A, X_B, Y_B, X_C, Y_C, \alpha) \in Win \times (\mathbb{N}^*)^4 \times \mathbb{R}^*$$

$$such\ as\ n_\alpha = \frac{Y_B}{X_B} = \frac{X_C}{Y_C} \tag{3}$$

$$and\ \alpha = arctan\ (n_\alpha)\ /n_\alpha\ \in \mathbb{Q}^*$$

$$and\ Y_B + Y_C + y_A < H_{win}, \qquad X_B + x_A < W_{win}$$

The set of rotated rectangles $\mathcal{R}$ is divided into two categories $\mathcal{R}a$ and $\mathcal{R}b$. These two categories are defined by formulas 4 and 5.

$$\mathcal{R}a = \{r \in \mathcal{R} |\ (yB \geq yc\ \ ET\ xA \geq xD)\ OU\ (yB \leq yc\ ET\ xA \leq xD)\} \tag{4}$$

$$\mathcal{R}b = \{r \in \mathcal{R} |\ (yB > yc\ \ ET\ xA < xD)\ OU\ (yB < yc\ ET\ xA > xD)\} \tag{5}$$

Knowing that B, C and D are other vertices of the rotated rectangle r, as shown in Fig. 3.

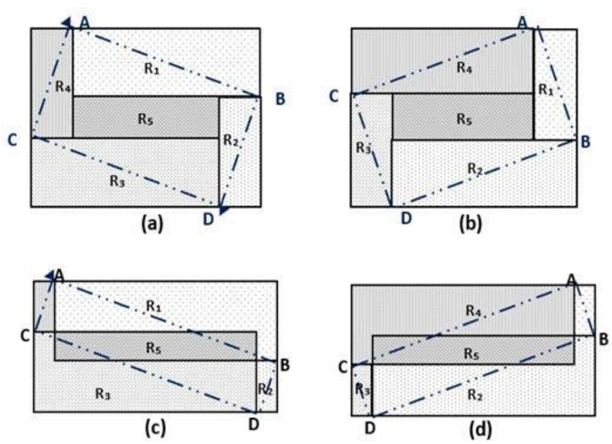


Fig. 4: Representation of a rotated rectangle of the category $\mathcal{R}a$ by (a) and (b) and $\mathcal{R}b$ by (c) and (d).

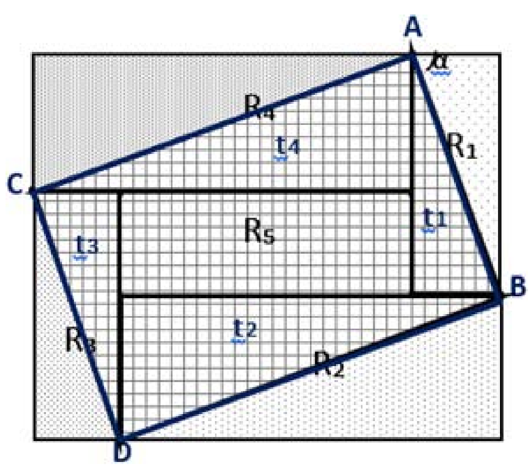


Fig. 5: A rotated rectangle divided into triangles.

b) The integration of a rotated rectangle

The principle of our method consists in dividing the rectangle that encapsulates r into five *normal* rectangles called Ri, as shown in Fig. 4 and 5. The result of the intersection of r with the rectangles Ri is the right triangle called ti . We illustrated this definition in Fig. 5. The following equation gives the identification of these triangles: $ti = r \cap Ri$ for $i \in \{1,2,3,4\}$.

Knowing that the integral of a rotated rectangle is:

$$I(r) = \iint_{(x,y)\in r} i(x,y) \tag{6}$$

such that $i(x, y)$ is the intensity of the pixel (x, y) belonging to the rectangle r. The exploitation of the integral image technique proposed by Viola and Jones [2] leads us to reformulate equation 6 in this form:

$$I(r) = \sum_{i=1}^{4} In(t_i) + S \times I(R_5)\ /\ \begin{cases} S = +1\ if\ r \in \mathcal{R}_a \\ S = -1\ if\ r \in \mathcal{R}_b \end{cases} \tag{7}$$

such that: $I(R5)$ is the *Integral Image* of $R5$ rectangle calculated according to the equation 2. $In(ti)$ is the integration of the triangle ti .

Indeed, we can divide a triangle t into several *normal* rectangles Ri (axis-aligned rectangles) (Fig. 6). The two vertices (Mi, Ni) of each rectangle Ri crossed by the hypotenuse of the triangle t are named *key points*. The set of these *key points*, $\mathcal{PC}$, are found by applying the Segment Drawing Rule (SDR) [13]. In fact, $\mathcal{PC} = \mathcal{M} \cup \mathcal{N}$ such that:

$$\mathcal{M} = \mathcal{M}t1 \cup \mathcal{M}t2 \cup \mathcal{M}t3 \cup \mathcal{M}t4$$

$$\mathcal{N} = \mathcal{N}t1 \cup \mathcal{N}t2 \cup \mathcal{N}t3 \cup \mathcal{N}t4$$

Knowing that $\mathcal{M}ti$ and $\mathcal{N}ti$ are, respectively, the set of key points Mi and Ni of the triangle ti.

Mathematically speaking, let $f: \mathcal{R} \rightarrow \mathcal{PC}^n$ be a function that determines the vector $vp = (P0, \ldots, Pn)$ of the key points, for each rectangle r of R, therefore: $f(r) = vp = (P0, \ldots, Pn)$. And $g: \mathcal{PC}^n \rightarrow \mathbb{R}$ a function that computes the *Integral Image* from a vector of points. In other words, $g(vp) = I(r)$. Indeed, we define the function I as a function composed of f and g: $I(r) = fog(r)$.

Therefore, the method proposed in this article, to calculate the *Integral Image* of a rectangle, is based on three essential tasks:

- Determine the rule for drawing a segment (SDR);
- Determine all the key points (Mi, Ni) of the rectangle r;
- Calculate the integral image of r according to its key points.

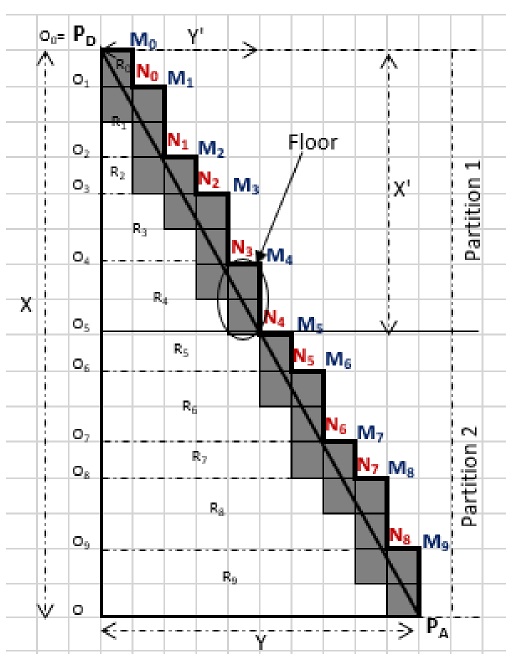
c) Triangle integration

The major problem of the computation part of the integral of a right triangle t is that each slope of its hypotenuse presents different difficulties. Is very difficult to integrate a line with an irrational slope because at the level of each pixel intersected by the line, the partition model is always different. For our case, the slope of the hypotenuse of the triangle t is rational. Specifically, a line with a rational slope produces a finite number of pixel partition models.

Consider, for example, a positive slope n of a line L, when this line crosses a set of pixels, the model of partitions of the pixels is repeated at each n pixels intersected by the line L [14]. This repetition is the key of our solution proposed in this article. More generally, let $d = n/m$ $(n, m \in Z \text{ and } m \geq 0)$ be a rational slope, the integer vector $vd = (m, n) / gcd(|n|, |m|)$ represents a 2D interval such as the partition model is periodic. Otherwise, the partition model at any pixel (x, y) is the same at $(x, y) + vd$. To determine the pixels traversed by a line L, we used the Bresenham algorithm [13], which makes it possible to define the Segment Drawing Rule (SDR).

Segment Drawing Rule (SDR): Based on the Bresenham algorithm [13], which allows determining all the pixels forming a segment, we have defined the rule allowing to go through all the pixels forming the hypotenuse of a iangle t (Fig. 6 (a)). This rule $RSDR$ is a vector formally defined as follows: $RSDR = (n1, n2, \dots, ne)$ such that e represents the number of floors which is equal to $min(Y', X')$ and ni that of pixels for the ith floor; Fig. 6 (b) gives an example of these values. A floor is a set of ni pixels aligned either vertically $(if\ \alpha < 45°)$ or horizontally $(if\ \alpha > 45°)$. The transition from one floor to another is done by moving a pixel to the right for sides $[AB]$ and $[CD]$ and left for $[AC]$ and $[BD]$. A particular case is that of $\alpha = 45°$, the number of pixels for each floor is always equal to 1, and the displacement can be done in both directions.

Indeed, the Segment Drawing Rule (SDR) is the result of the function s: $\mathfrak{S} \rightarrow \mathbb{N}^e$ which takes in parameter a segment S and returns a vector.



(a)

(X, Y)	(16,10)
$N = \gcd(\lvert X \rvert, \lvert Y \rvert)$	2
$(X', Y') = (X, Y) / N$	(8,5)
$e = min(X', Y')$	5
$R_{SDR} = (n_1, n_2, \dots, n_e)$	$R_{SDR} = (2{,}3{,}2{,}3{,}2)$
k	10
d	$vertical - right$

(b)

Fig. 6: (a) Rule for drawing a segment, (b) Example of values for $X = 16$, $Y = 10$.

$RSDR = (n1, n2, \dots, ne)$. S is the set of segments of a polygon; for our case, we are talking about a rectangle with four segments. Each segment is formally defined as follows: $S = (PD, PA, X', Y', N, k, d)$, such that:

- PD and PA are, respectively, the starting point and the arrival point of the segment S;
- $N = \gcd(|X|, |Y|)$ is the number of partitions or repetitions of $RSDR$;
- $(X', Y') = (X, Y) / N$;
- $k = min(X, Y) = min(X', Y') * N = e * N$: the number of floors composing the hypotenuse of the triangle;
- d: Represents the direction of the path to follow, to browse all the pixels:

if α<45° then d∈{vertical-right,vertical-left}

if α>45° then d∈{horizontal-right,horizontal-left}

Algorithm 1 shows the definition of the segment drawing rule (SDR) for both segments $[AB]$ and $[CD]$ with a direction $d = Vertical - Right$.

The authors of [14] have shown that the number of pixels, denoted np, traversed by the hypotenuse of the right triangle t for each partition is $np = |X'| + |Y'| - 1$, indeed, we easily deduce that the total number of pixels traversed by the hypotenuse of t is: np

$\times$ N. Therefore, to determine the SDR, algorithm 1 traverses $(np + e) \times N$ pixels.

Integral image of a right triangle: As mentioned above, the principle of our method is based on the division of a triangle into k normal rectangles Ri. A normal rectangle Ri is defined by the following quadruple: (see Fig. 6 (a)) $Ri = [OiMiNiOi{+}1]$ for $t1$, such us :

- $t1$ is the triangle with the hypotenuse as segment $[AB]$, shown in Fig. 5;
- k: Number of floors;
- $i = 0,1, \dots, k-1$;

Algorithm 1 Get SDR

Require

- P_D: $Starting\ point\ (A\ or\ C)$
- $(X', Y') = (X, Y)\ /$ $\gcd(|X|, |Y|)$ and $(X, Y) =$ (X_B, Y_B) $see\ Figure\ 2$
- $e = min(X', Y')$ number of floors
- $n_i = umber\ of\ pixels\ on\ the\ i^{th}\ floor$

Ensure: $SDR\ for\ segment\ [AB]\ and\ [CD]$

Initialisation:

$p(x_p, y_p) = P_D + 1$

Do:

```
For i = 1 to e − 1 do
    c = x_p, n_i = 0
    While c ≥ 0 do
        x_p = c
        if p in segment (AB or CD)
        and p inside rectangle then
            −increment n_i
            −increment c
        Else
            x_p = c − 1
            y_p = y_p + 1
            Break;
    End While
End For
Return SDR
```

- $Nk{-}1 = PA$ and $O0 = PD$;
- $Ok = O$.

Therefore, the *Integral Image* of a triangle t will be the sum of those of all rectangles Ri, formally:

$$In(t) = \sum_0^{k-1} Im(R_i) / \ k\text{: number of rectangles } R_i. \qquad (8)$$

Indeed, the Integral Image of a normal rectangle Ri, $Im(R\boldsymbol{i})$, according to formula 2 is:

$$Im(\boldsymbol{R_i}) = ii(N_i) - ii(O_{i+1}) - ii(M_i) + ii(O_i) \qquad (9)$$

Such as the function ii represents the Integral Image of a given pixel.

Therefore, from equation 8 and 9, we deduce the formula to calculate the Integral Image of a triangle t (case of triangle $t1$):

$$In(t) = \sum_0^{k-1} Im(\boldsymbol{R_i}) = ii(P_D) + ii(P_A) - ii(O) + \sum_0^{k-2} ii(N_i) - \sum_0^{k-1} ii(M_i) \qquad (10)$$

Knowing that for $t1$: $PD = A\ et\ PA = B$.

According to this principle, we deduce the formulas for calculating the integral image of all types of triangles of the rotated rectangle r: (Fig. 5)

$$In(t_1) = ii(A) + ii(B) - ii(O) - \sum_0^{k-1} ii(M_i) + \sum_0^{k-2} ii(N_i) \qquad (11)$$

$$In(t_2) = ii(O) - \sum_0^{k-1} ii(M_i) + \sum_0^{k-2} ii(N_i) \qquad (12)$$

$$In(t_3) = -ii(O) + \sum_0^{k-1} ii(M_i) - \sum_0^{k-2} ii(N_i) \qquad (13)$$

$$In(t_4) = -ii(A) - ii(C) + ii(O) + \sum_0^{k-1} ii(M_i) - \sum_0^{k-2} ii(N_i) \qquad (14)$$

d) Rotated Rectangle Integration

In the end and according to equation 7, formula 15 shows the Integral Image of a rotated rectangle. That of the rectangle of the medium $R5$ is easily found via the equation 2.

$$I(r) = ii(B) - ii(C)) - \sum_0^{k-1} ii(M_{1i}) + \sum_0^{k-2} ii(N_{1i}) - \sum_0^{l-1} ii(M_{2i}) + \sum_0^{l-2} ii(N_{2i}) + \sum_0^{k-1} ii(M_{3i}) - \sum_0^{k-2} ii(N_{3i}) + \sum_0^{l-1} ii(M_{4i}) - \sum_0^{l-2} ii(N_{4i}) \qquad (15)$$

Knowing that k is the number of floors of the hypotenuses of triangles $t1$ and $t2$ and l that of triangles $t2$ and $t4$. So, the total number of key points Pi of a rotated rectangle r, adding the two points B and C, is $n = 4\ (k + l) - 2$.

In general, the *Integral Image* of a rotated rectangle r is:

$$I(r) = \sum_0^{n-1} \rho_i * ii(P_i) / \rho_i \in \{+1, -1\} \qquad (16)$$

Such as ρi is the parity corresponding to the pixel Pi. Fig. 7 shows the variation of the sign ρi according to the type of the triangle and the key point ($+1$, for B, and -1 for C)

Algorithm 2 illustrates the process proposed to find these points for the two segments $[AB]$ and $[CD]$ with the direction of the path $d = Vertical - Right$.

Indeed, to compute the *Integral Image* of a rotated rectangle, we need 4 $(k + 1) - 2$ access to the memory, which means that the associated algorithm has a linear complexity O(n). In practice, the worst-case complexity varies between 40 and 74 operations for only 12% of cases.

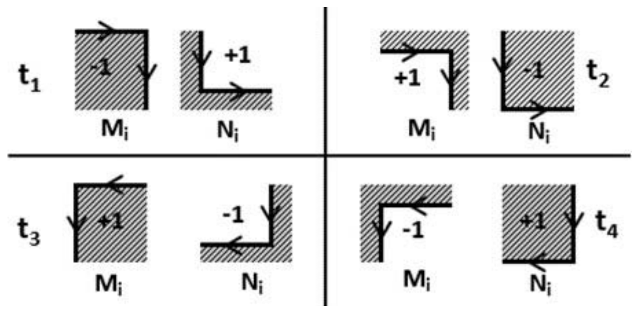


Fig. 7: The variation of the sign ρi according to the type of the triangle and the key point (M or N)

Algorithm 2 Get key points

Require

- $S = (P_D, P_A, X', Y', N, k, d)$ *such as* $P_D =$ A or C and $P_A =$ B or D
- $R_{SDR} = (n_1, n_2, \dots, n_e)$
- $d = Vertical - Right$

Ensure: *Key points for segment* $[AB]$ *and* $[CD]$

Initialisation:

$$M_0(x_{M_0}, y_{M_0}) = \begin{cases} x_{M_0} = x_{P_D} + 1 \\ y_{M_0} = y_{P_D} \end{cases}$$

Do:

```
For i = 1 to k − 1 do
    x_{M_i} = x_{M_{i−1}} + 1
    c = i mod e
    if c <> 0 then
        y_{M_i} = y_{M_{i−1}} + n_{c−1} − 1
    Else
        y_{M_i} = y_{M_{i−1}} + n_{e−1}
    x_{N_{i−1}} = x_{M_{i−1}}
    y_{N_{i−1}} = y_{M_i}
End For
Return key points
```

The complexity in the best case varies between 6 and 20 operations for more than 45% of cases, and 23 on average for all cases.

IV. Experiences

During the learning phase, we adopted the use of two of the most known and most available databases in our field, which are: UMIST Face Database [17] and CMU-PIE Face Database [18]. The base of faces UMIST consists of 564 images - cut and trimmed - of 20 people (mixed race, kind, and appearance) [17]. In this database we represent each person's frontal profile by a multitude of images illustrating a range of poses from different angles; their number is about 19 up to 36 for each person. These images are sampled, in our experience, up to a resolution of 20 × 20. Their number is 6900 images of faces. The CMU-PIE database contains 41,368 images obtained from 68 people. These images were taken in the CMU 3D room using a set of 13 high-quality color cameras synchronized with 21 flashes. The resulting RGB color images are 640 × 486 in size. These images are sampled, in our experiment, up to a resolution of 18 × 20. And for the training of our detector, we used 9996 images of faces. For the test phase, we used the standard MIT + CMU base which contains 117 images with 511 faces.

The result of our work is the creation of two detectors UMIST-Detector and PIE-Detector from the process of learning using the two image bases, respectively, UMIST and PIE CMU described above.

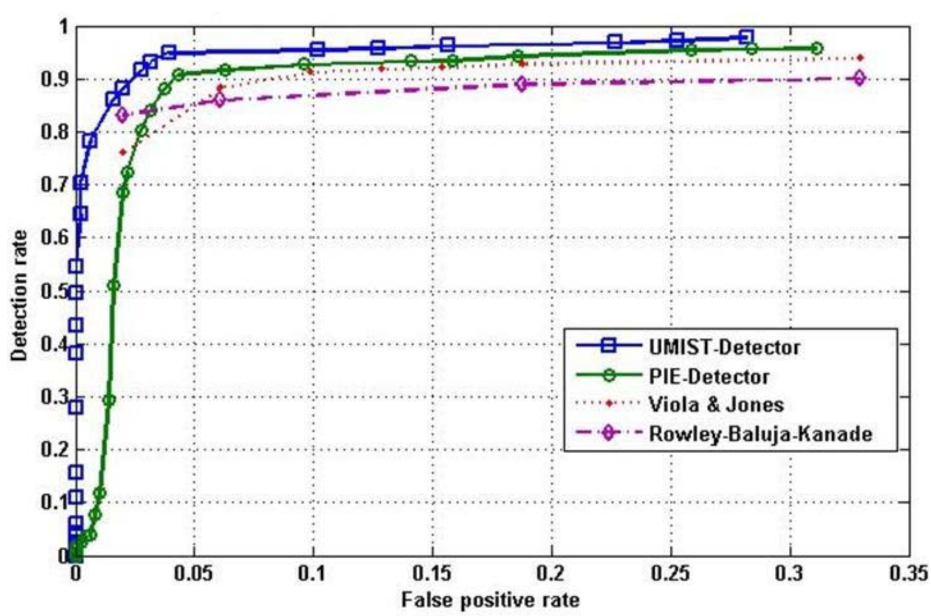


Fig. 8: The ROC curve of UMIST-Detector, PIE-Detector, and other detectors using the MIT-CMU test base.

The learning phase uses more than 300,000 features (or weak classifiers) divided into two types: normal and rotated. The storage of this large number of features took on a large amount of memory. The result of this phase is a detector composed of a group of features selected by the AdaBoost algorithm. For each feature selected, it takes about 3 to 4 hours - sometimes more - calculating time using an HP Notebook PC with a 2.6 GHz frame rate and 4 GB of memory. This slowness of learning was considered a major drawback of the Viola & Jones method.

Table 1: Number of normal and rotated Haar-Like features selected by AdaBoost for each Data Base

	UMIST	CMU-PIE	VIOLA & JONES
Normal	4103	4075	4297
Rotated	6424	4393	0
TOTAL	10527	8468	4297

Our program was run for 21 days for the first experience and 15 days for the second. The first detector consists of 10527 features distributed over 53 stages, and the second detector contains 8468 features spread over 46 stages.

The Viola-Jones detector contains 4297 features distributed over 32 stages, as is presented in the article of its authors [2][3]. The difference in the number of weak classifiers of our detectors and that of Viola and Jones is because we use, at most of the original features, those rotated by different angles. Table I shows the numbers of normal and rotated features selected by AdaBoost for each experiment, compared by the number obtained by Viola and Jones. Indeed, this added value has allowed us to achieve good results. These results clearly show that our detectors have correct detection rates greater than or nearly equal to those reported by Viola-Jones. UMIST-Detector has a detection accuracy of up to 97.8%, and this amounts to the fact that the images of the UMIST database are better pre-processed and standardized at the level of illumination, rotation of faces, different human races, etc. With PIE-Detector the detection rate cannot exceed the 95.9% threshold. To have performance tests comparable to those performed by Viola & Jones and Rowly [2][3][11], we used the MIT-CMU test database, which consists of 117 images and 511 faces. Viola and Jones used 130 images and 507 faces. A difference that we neglected because most of the images were the same or have undergone some slight modifications. Fig. 8 illustrates these results as a ROC.

The results are generally better, especially those obtained by Viola and Jones. Our two detectors have lower false alarm rates than the other methods; 28% for the first experience and 31% for the second. By using a 2.3 GHz core i3 and a 4 GB memory capacity, our detectors can scan an image of 252 $\times$ 426 pixels in about 0.9 seconds with a scaling factor of 1.2. what makes them a little bit slow compared to that of Viola and Jones (0.7 seconds to scan an image of 384 $\times$ 288), and this is mainly due to the large number of weak classifiers used by our method. The average speed to determine all the key points for each rotated feature does not exceed 62.4 µs.

V. Conclusion

In this work, we have proposed a simple, and effective method to integrate a rectangle rotated by any angle of rotation. The principle of this technique is to find specific pixels (key points) for each segment of the rectangle, then calculate the sum of its pixels according to these points. To find these points, we adopted the Bresenham algorithm for drawing a segment. This technique applied to the Viola and Jones algorithm with rotated HaarLike features added to the core set has proven efficiency and a high detection rate for detecting rotated faces in an image. In fact, we performed two experiments using, for the training phase, two image databases known in the literature, notably UMIST and CMU- PIE, and the MIT-CMU database for the test phase. Our perspective is to apply this technique for real-time detection and also for face and emotion recognition.

References Références Referencias

1. Y. Freund et R. Schapire, “A desicion-theoretic generalization of on-line learning and an application to boosting”, Journal of Computer and System Sciences(JCSS), vol. 55, no 1, 1997, p. 119-139.
2. Viola P, Jones M. Robust real-time object detection. International Journal of Computer Vision (IJCV), 2001, 57(2), p. 137-154.
3. P. Viola, M. Jones. "Rapid Object Detection Using a Boosted Cascade of Simple Features", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 511-518, 2001.
4. P. Viola, M. Jones, D. Snow. "Detecting pedestrians using patterns of motion and appearance", International Journal of Computer Vision, Vol. 63, Issue 2, pp. 153–161, July 2005.
5. Lienhart R, Maydt J. “An extended set of haar-like features for rapid object detection”, International Conference on. IEEE, 2002, vol. 1, pp: 900-903.
6. Miyamoto R, Sugano H, Saito H, et al. “Pedestrian recognition in far- infrared images by combining boosting-based detection and skeleton- based stochastic tracking”, Advances in Image and Video Technology. Springer Berlin Heidelberg, 2006, pp: 483-494.
7. A.L.C. Barczak. “Toward an Efficient Implementation of a Rotation Invariant Detector using Haar-Like Features”, Proceedings of the IEEE/RSJ

international conference on Intelligent robots and systems, Nouvelle-Zélande, Dunedin, 2005.

8. S. Du, N. Zheng, Q. You, Y. Wu, M. Yuan, J. Wu. "Rotated Haar-Like Features for Face Detection with In-Plane Rotation", 12th International Conference, VSMM 2006, Xi'an, China, October 18-20, Proceedings, pp 128-137, 2006.
9. C. H. Messom, A. L. C. Barczak, "Stream processing for fast and efficient rotated Haarlike features using rotated integral images". IJISTA 7(1): 40-57 (2009).
10. G. A. Ramirez, O. Fuentes, "Multi-Pose Face Detection With Asymmetric Haar Features", Applications of Computer Vision, WACV, 2008, IEEE Workshop in Copper Mountain,CO.
11. H.A. Rowley, S. Baluja; T. Kanade, "Neural network-based face detection", In IEEE Trans. Pattern Analysis and Machine Intelligence, vol.20 (1998), p. 23–38.
12. M. Oualla, A. Sadiq, S. Mbarki. "A new approach to represent rotated haar-like features for objects detection", Journal of Theoretical & Applied Information Technology (JATIT), Vol.78. N° 1, pp 15 – 23, 10th August 2015.
13. Jack E. Bresenham, "Algorithm for computer control of a digital plotter", IBM Systems Journal, vol. 4, no 1, 1th January 1965, p. 25–30.
14. M.T. Pham, Y. Gao, V.D.D. Hoang, T.J. Cham, "Fast Polygonal Integration and Its Application in Extending Haar-like Features to Improve Object Detection", In proceeding of the IEEE conference on Computer Vision and Pattern Recognition (CVPR); San Francisco, CA, 2010.
15. G. Doretto, T. Sebastian, P. Tu, and J.Rittscher. "Appearance-based person reidentification in camera networks: Problem overview and current approaches". In Journal of Ambient Intelligence and Humanized Computing, pp. 1–25, Springer Berlin / Heidelberg, 2011.
16. Crow, F. "Summed-area tables for texture mapping", SIGGRAPH, 84, pp. 207–212, 1984.
17. D.B. Graham and N.M. Allinson, "Characterizing Virtual Eigen signatures for General Purpose Face Recognition", NATO Advanced Study Institute, Face recognition: from theory to applications; 1997; Stirling in NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES, vol. 163, pp. 446-456, (Springer), 1998.
18. T. Sim, S. Baker, M. Bsat. "The CMU Pose, Illumination, and Expression (PIE) Database of Human Faces", Transactions on Pattern Analysis and Machine Intelligence, Vol. 25(12): pp. 1615–1618, 2003.