Very Fast Template Matching*

H. Schweitzer, J.W. Bell, and F. Wu

[1] The University of Texas at Dallas, Richardson TX 75083, USA
[2] Voxar AG, 6516 Benchmark Dr., Plano TX 75023, USA
{haim,wufeng}@utdallas.edu, wes.bell@voxar.de

Abstract. Template matching by normalized correlations is a common technique for determine the existence and compute the location of a shape within an image. In many cases the run time of computer vision applications is dominated by repeated computation of template matching, applied to locate multiple templates in varying scale and orientation. A straightforward implementation of template matching for an image size n and a template size k requires order of kn operations. There are fast algorithms that require order of $n \log n$ operations. We describe a new approximation scheme that requires order n operations. It is based on the idea of "Integral-Images", recently introduced by Viola and Jones.

1 Introduction

Searching and locating shapes in images and video is an important component in many computer vision systems. Template matching by normalized correlations (e.g., [1]) is arguably the most common approach, and can be traced back to very early research in pattern recognition [2]. Suppose t is a template to be detected in an image f. Template matching by normalized correlations computes the following value at each point (u, v) of the image:

$$h(u,v) = \frac{\sum_{x,y} f(u+x, v+y)\ t(x,y)}{\sqrt(\sum_{x,y} f^2(u+x, v+y))} \tag{1}$$

where the summations are over all template coordinates. A large value of $h(u, v)$ indicates a likely match at the coordinate (u, v). It can be shown that a match that maximizes h is identical to the template t up to scaling.

To express the complexity of normalized correlations we denote by n the number of image pixels and by k the number of template pixels. A straightforward computation of (1) requires order of kn operations. Since both numerator and denominator can be implemented with correlations, computing (1) is directly tied to the complexity of computing convolutions, a well researched topic (e.g.,

* This material is based in part upon work supported by the Texas Advanced Research Program under Grant No. 009741-0074-1999 and the Texas Advanced Technology Program under Grant No. 009741-0042-2001

[3]). Using fast convolutions (e.g. by means of an FFT algorithm) it is possible to compute (1) in order of $n \log n$ operations for any value of k.

The computational cost of template matching may dominate the speed of computer vision systems (see, e.g., [4]). Typically such systems are built to detect multiple objects, with a search being performed at multiple scales and orientations. Systems that use fast $n \log n$ algorithms are rare. A possible reason might be the complex coding, and the fact that order $n \log n$ run time might be too high. Instead, common strategies include the development of specialized hardware (e.g., [5,6]) and special acceleration techniques that compute the value of $h(u, v)$ only at a subset of the image locations.

One such approach (e.g., [7]) is to perform a first stage where matching is computed for a small subset of the template pixels. The exact value of $h(u, v)$ is then computed in a second stage only at image locations that were ranked high in the first stage. A related approach (e.g., [8,9]) uses coarse-to-fine search, determining matching candidates first in a low resolution version of the image. The search is refined in higher resolutions only at areas where candidates were found in the lower resolution.

The motivation for our work is a recent result by Viola and Jones [10] that takes a different approach to speeding up matching. Unlike the approaches that compute $h(u, v)$ only at a subset of image locations, Viola and Jones show that these values can be computed very fast (order n operations) everywhere for a special type of template: axis parallel uniform rectangles. This can be used to create more complex templates as combinations of several rectangles. Matching can then be computed by using the $h(u, v)$ values computed from several templates as "features" that are analyzed by a learning algorithm.

1.1 The Main Contribution

The key to the speedup obtained by Viola and Jones is a pre-computed data-structure called "an integral image". Our main contribution is the observation that integral images can be used to compute algebraic moments in linear time. This enables us to compute in linear time the best least squares approximation polynomials at each location of the image. (A total of n polynomials are computed for an image of n pixels.) These approximations are used to compute accurate estimates to the values of $h(u, v)$.

The order of the polynomials used in the approximation affects the run time complexity. Using polynomials of order d requires order of $d^2 n$ operations. On the other hand, the estimates to $h(u, v)$ improve with the order of the polynomial approximation. Our experiments show that second order polynomial approximations give sufficiently accurate estimates in typical situations at significantly reduced run time.

1.2 Paper Organization

The paper is organized as follows. Integral images are introduced in Section 2, following Viola and Jones. It is shown in Section 3 that local algebraic moments of low order can be computed efficiently using integral images. These moments are used to compute local least squares polynomial approximations to the image. A closed form expression to the normalized matching computed between these polynomials and a template is derived in Section 4. The algorithm steps are described in Section 5, with an analysis of its complexity. Experimental results are described in Section 6.

2 Integral Images

This section describes the idea of integral images following Viola and Jones. Suppose $f(x, y)$ is an integrable function for nonnegative x, y. Define:

$$I(u,v) = \int_{x=0}^{u} \int_{y=0}^{v} f(x,y)dydx$$

then an explicit integral can be computed as:

$$\int_{x=x_a}^{x_b} \int_{y=y_a}^{y_b} f(x,y)dydx = I(x_b,y_b) + I(x_a,y_a) - I(x_a,y_b) - I(x_b,y_a)$$

The discrete version of this formula is very similar (but not identical). For an image $f(x, y)$ define the integral image $I(x, y)$ as:

$$I(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} f(x',y') \tag{2}$$

Then a summation of f over any axis parallel rectangle can be computed as:

$$\begin{aligned} \textstyle\sum_{x=x_a}^{x_b} \sum_{y=y_a}^{y_b} f(x,y) = \\ & I(x_b,y_b) + I(x_a-1,y_a-1) \\ & -I(x_a-1,y_b) - I(x_b,y_a-1) \end{aligned} \tag{3}$$

As shown by Viola and Jones the integral image (2) can be computed in one pass over the image, using the following recursive formulas:

$$s(x,y) = s(x,y-1) + f(x,y), \quad I(x,y) = I(x-1,y) + s(x,y)$$

where s(x,y) is the cumulative column sum.

This implies that once the integral image is computed, the sum of values over any rectangle can be computed very fast. Viola and Jones used this to compute "features" that were created from correlations with two, three, and four adjacent rectangles, as shown schematically in Fig. 1. The question of whether these results can be generalized to other functions, not created from uniform rectangles, was not addressed in [10].

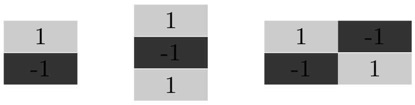


Fig. 1. Rectangle features used by Viola and Jones. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent. A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. A four-rectangle feature computes the difference between diagonal pairs of rectangles.

3 Fast Computation of Local Algebraic Moments

In this section we show that integral images can be used to compute low order local algebraic moments, generalizing the results of Viola and Jones. The algebraic moments of a rectangular region are defined as follows:

$$m_{pq} = \sum_{x=x_a}^{x_b} \sum_{y=y_a}^{y_b} x^p y^q f(x,y) \tag{4}$$

The order of the moment m_{pq} is $p+q$. Fig. 2 shows the templates corresponding to moments of order 1 and 2, with the origin taken at the center of the rectangle. Notice the similarity between these and the rectangle features of Viola and Jones.

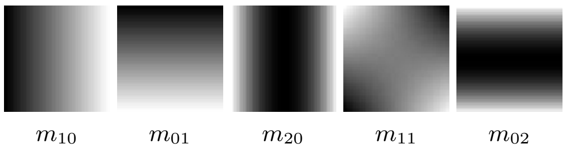


Fig. 2. Templates corresponding to algebraic moments

Replacing f with $x^p y^q f$ in equations (2),(3) it is clear that algebraic moments can be computed with the technique of integral images. This by itself is not useful since the value of m_{pq} (with the exception of m_{00}) depends on the choice of origin, and the origin in (2),(3) is the image origin. The reason for choosing algebraic moments is the observation that there is a simple relationship between m_{pq} and the centralized moments computed with respect to the center of the rectangle. Let μ_{pq} denote the centralized moments, and let (x_0, y_0) denote the center of the rectangle. Then:

$$\mu_{pq} = \sum_{x=x_a}^{x_b} \sum_{y=y_a}^{y_b} (x - x_0)^p (y - y_0)^q f(x,y)$$

A direct computation gives:

$$\mu_{pq} = \sum_{\substack{0 \le s \le p \\ 0 \le t \le q}} (-1)^{s+t} \binom{p}{s} \binom{q}{t} x_0^s y_0^t m_{p-s,q-t} \tag{5}$$

The important observation is that the moment μ_{pq} is computed from moments m_{pq} of the same or lower order. The explicit formulas for several low order moments are:

$$\begin{aligned}
\mu_{00} &= m_{00} \\
\mu_{10} &= m_{10} - x_0 m_{00} \\
\mu_{01} &= m_{01} - y_0 m_{00} \\
\mu_{20} &= m_{20} - 2x_0 m_{10} + x_0^2 m_{00} \\
\mu_{11} &= m_{11} - x_0 m_{01} - y_0 m_{10} + x_0 y_0 m_{00} \\
\mu_{02} &= m_{02} - 2y_0 m_{01} + y_0^2 m_{00} \\
\mu_{30} &= m_{30} - 3x_0 m_{20} + 3x_0^2 m_{10} - x_0^3 m_{00} \\
\mu_{21} &= m_{21} - y_0 m_{20} - 2x_0 m_{11} + 2x_0 y_0 m_{10} + x_0^2 m_{01} - x_0^2 y_0 m_{00} \\
\mu_{12} &= m_{12} - 2y_0 m_{11} - x_0 m_{02} + y_0^2 m_{10} + 2x_0 y_0 m_{01} - x_0 y_0^2 m_{00} \\
\mu_{03} &= m_{03} - 3y_0 m_{02} + 3y_0^2 m_{01} - y_0^3 m_{00}
\end{aligned}$$

In summary, this section gives an algorithm for computing the local (centralized) algebraic moments μ_{pq} up to a given order at any given rectangle in the image. The required preprocessing that needs to be computed just once per image is the integral images of the functions $x^p y^q f(x, y)$. The values of m_{pq} at the desired rectangle are computed from (3), and the values of μ_{pq} are computed from the m_{pq} values using (5).

4 Fast Template Matching

In this section we develop a match measure that uses the local moments. In developing this measure we focus on a single rectangle in the image. The pixel values are denoted by $f(x, y)$ as before, but the origin of the coordinate system is placed at the center of the rectangle.

Let t be a template with the local moments $U_t = (\mu_{00}^t, \dots, \mu_{pq}^t)$. We denote by $U_f(u, v) = (\mu_{00}^f, \dots, \mu_{pq}^f)$ be the corresponding local moments computed over a rectangle centered at (u, v) of the same dimensions as the template. One may be tempted to use the Euclidean distance, or other direct comparisons between U_f and U_t as an indication of match. One choice is to use the same normalized measure as in (1), which gives:

$$h(u, v) = \frac{\sum_{p,q} \mu_{pq}^f(u, v)\ \mu_{pq}^t}{\sqrt{(\sum_{p,q} (\mu_{pq}^f(u, v))^2)}} \tag{6}$$

As in the case of (1) it can be shown that (6) is maximized by the exact match, but unlike (1) it does not degrade gracefully. As shown in Section 6 using these values for matching gives unsatisfactory results since the moments of order 2 and higher are very sensitive to noise. We proceed to derive an alternative measure based on a polynomial approximation to f over the rectangle.

4.1 Notation

We wish to approximate a rectangle of the image $f(x, y)$ with a low degree polynomial. As an example, the polynomial of total degree 2 can be written as:

$$p(x, y) = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2$$

Vector notation simplifies the derivations in this section. Let A be a coefficients vector and X a variables vector. U denotes a vector of centralized moments. For example, limiting the order to 2 we have:

$$A = \begin{pmatrix} a_{00} \\ a_{10} \\ a_{01} \\ a_{20} \\ a_{11} \\ a_{02} \end{pmatrix}, \quad X = \begin{pmatrix} 1 \\ x \\ y \\ x^2 \\ xy \\ y^2 \end{pmatrix}, \quad U = \begin{pmatrix} \mu_{00} \\ \mu_{10} \\ \mu_{01} \\ \mu_{20} \\ \mu_{11} \\ \mu_{02} \end{pmatrix}$$

In this notation the polynomial p above can be written as $p = A'X$. (The symbol " $'$ " indicates matrix transpose.) We use the "bar" notation to indicate summation over the rectangle. Writing Equation (1) in this notation gives:

$$U_f = \overline{Xf}$$

where U_f is the moments vector computed from f.

4.2 Approximation by Polynomials

In our notation the mean squared error of a polynomial approximation to f is given by:

$$E = \overline{(p - f)^2} = A'\overline{XX'}A - 2A'U_f + \overline{f^2} = A'BA - 2A'U_f + \overline{f^2}$$

where $B = \overline{XX'}$ is a positive-definite and symmetric matrix. For the example above we have:

$$B = \overline{XX'} = \overline{\begin{pmatrix} 1 & x & y & x^2 & xy & y^2 \\ x & x^2 & xy & x^3 & x^2y & xy^2 \\ y & xy & y^2 & x^2y & xy^2 & y^3 \\ x^2 & x^3 & x^2y & x^4 & x^3y & x^2y^2 \\ xy & x^2y & xy^2 & x^3y & x^2y^2 & xy^3 \\ y^2 & xy^2 & y^3 & x^2y^2 & xy^3 & y^4 \end{pmatrix}} \tag{7}$$

Therefore, the error E is quadratic in A, and the value of A that minimizes E is $A_f = B^{-1}U_f$. This gives the following formula for the best polynomial approximation to f:

$$p_f = U_f'B^{-1}X \tag{8}$$

Observe that $\overline{p_f^2} = U_f'B^{-1}U_f$.

4.3 The Match Measure

We proceed to compute our main result, the match measure obtained from Eq. (1) by replacing f with p_f. (No approximation is applied to the template t.) The approximation to the numerator of (1) can be written as:

$$\overline{p_f\ t} = \overline{U_f' B^{-1} X\ t} = U_f' B^{-1} \overline{X\ t} = U_f' B^{-1} U_t$$

From the note following (8) the approximation to the denominator of (1) can be written as:

$$\sqrt{(\overline{p_f^2})} = \sqrt{U_f' B^{-1} U_f}$$

Combining these expressions gives the following measure:

$$h(u,v) = \frac{U_f'(u,v) B^{-1} U_t}{\sqrt{U_f'(u,v) B^{-1} U_f(u,v)}} \tag{9}$$

where $U_f(u,v)$ are the local moments of f computed over a rectangle centered at (u,v).

We note that the measure given in (6) is up to a scaling factor of the cosine of the angle between the two vectors U_t and U_f. The same is true for (9), but in an elliptic coordinate system determined by the positive definite matrix B^{-1}. In particular, the value of (9) is maximized by the exact match.

4.4 Structure of the Matrix *B*

The dominant computation steps in (9) are the matrix/vector products. Since the matrix B depends only on the moments order and the dimensions of the rectangle, but not on f, it is possible to take advantage of its structure and simplify the computation. Define:

$$X_p = \sum x^p, \quad \text{where the sum is over one row of the rectangle.}$$
$$Y_q = \sum y^q, \quad \text{where the sum is over one column of the rectangle.}$$

The entries of B are terms of the form $X_p Y_q$. When measured with respect to the rectangle center, X_p and Y_q are 0 for odd p, q. Thus, the matrix B in Eq. (7) can be written as:

$$B = \begin{pmatrix} X_0Y_0 & 0 & 0 & X_2Y_0 & 0 & X_0Y_2 \\ 0 & X_2Y_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & X_0Y_2 & 0 & 0 & 0 \\ X_2Y_0 & 0 & 0 & X_4Y_0 & 0 & X_2Y_2 \\ 0 & 0 & 0 & 0 & X_2Y_2 & 0 \\ X_0Y_2 & 0 & 0 & X_2Y_2 & 0 & X_0Y_4 \end{pmatrix}$$

A direct computation gives:

$$B^{-1}\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \end{pmatrix} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{pmatrix}$$

where

$$\begin{aligned}
w_1 &= \frac{(X_0X_4Y_0Y_4 - X_2^2Y_2^2)z_1 + X_0X_2(Y_2^2 - Y_0Y_4)z_4 + (X_2^2 - X_0X_4)Y_0Y_2z_6}{K} \\
w_2 &= \frac{z_2}{X_2Y_0} \\
w_3 &= \frac{z_3}{X_0Y_2} \\
w_4 &= \frac{X_0X_2(Y_2^2 - Y_0Y_4)z_1 + X_0^2(Y_0Y_4 - Y_2^2)z_4}{K} \\
w_5 &= \frac{z_5}{X_2Y_2} \\
w_6 &= \frac{(X_2^2 - X_0X_4)Y_0Y_2z_1 + (X_0X_4 - X_2^2)Y_0^2z_6}{K} \\
K &= X_0(X_2^2 - X_0X_4)Y_0(Y_2^2 - Y_0Y_4)
\end{aligned}$$

These formulas can be used to simplify the computation of both numerator and denominator of (9). They show that for this case the matrix/vector product requires only 8 multiplications. Similar simplifications can also be computed for the matrix needed for the third order approximation.

5 The Algorithm

This section summarizes the algorithm. Given a template t and an image f our goal is to estimate the matching values $h(u, v)$ using polynomial approximations of order d. Observe that there are l moments $\mu_{p,q}$ satisfying $p + q \leq d$, where $l = (d + 1)(d + 2)/2$.

Step 1. Compute U_t, the moments of the template. U_t is a vector of length l.
Step 2. Compute the integral images for the l moments m_{pq}.
Step 3. For each location (u, v) of the image compute the centralized moments $U_f(u, v)$ using equations (3) and (5).
Step 4. For each location (u, v) of the image compute the values of $h(u, v)$ from U_t and $U_f(u, v)$ using (9).

The heavy computation steps are 2,3,4. With n the number of image pixels the number of operations in steps 2,3,4 is order of nd^2. Observe that when matching multiple templates over the same rectangle dimensions it is enough to repeat only steps 1 and 4.

6 Experimental Results

All experiments described in this section were performed on a single image and a single template, created from the ORL collection of faces. The image to be searched was created as a mosaic of 40 face images, and the template was chosen as one of the faces in the mosaic. (It is the second face from the right in the fourth row from the top.) The template size is 91×111, and the image size is 826×670.

Run time: We observed a huge saving in time when compared to the order kn implementation of the classic technique. Our technique was faster by about a factor of 100.

Quality of the results: Several criteria were used to evaluate the results:

- Location of faces should have higher values than location of non-faces.
- Location of faces of the individual shown in the template (all faces on fourth row) should have higher values than the location of other faces.
- The location of exact match should have the highest value.

It is known (e.g., [1]) that template matching by normalized correlations produces smooth output, so that matches appear as peaks of the match measure. Therefore, in evaluating the performance we consider only the peak maxima. A peak maximum in the correlation output was computed as a local maximum, defined as a pixel whose matching value is greater than or equal to the matching values of all other pixels in its 8-neighborhood.

Poor quality was observed for the moments method, as given by Eq. (6), and for the proposed technique implemented with 1st order polynomials. These cases will not be discussed further. Implementations of the proposed technique with 2nd and 3rd order polynomials produced very good results even in the presence of large amounts of noise.

What is being shown: We show results of experiments performed with noise-free and noisy images. In each case the local maxima of the measure were detected, and the top 20 are plotted using the following graphic code:

8-pointed star: the location of the highest match
4-pointed cross: the location of matches ranked 2 and 3
Diamond: the location of matches ranked 4 to 10
Square: the location of matches ranked 11 to 20

Color coded images of match values (not limited to local maxima) are also shown, using the following color code:

White: Top 0.1% of the match values.
Blue: Top match values between 0.1% and 1.0%
Magenta: Match values corresponding to the range 1.0% to 5.0%
Black: Match values below the top 5%

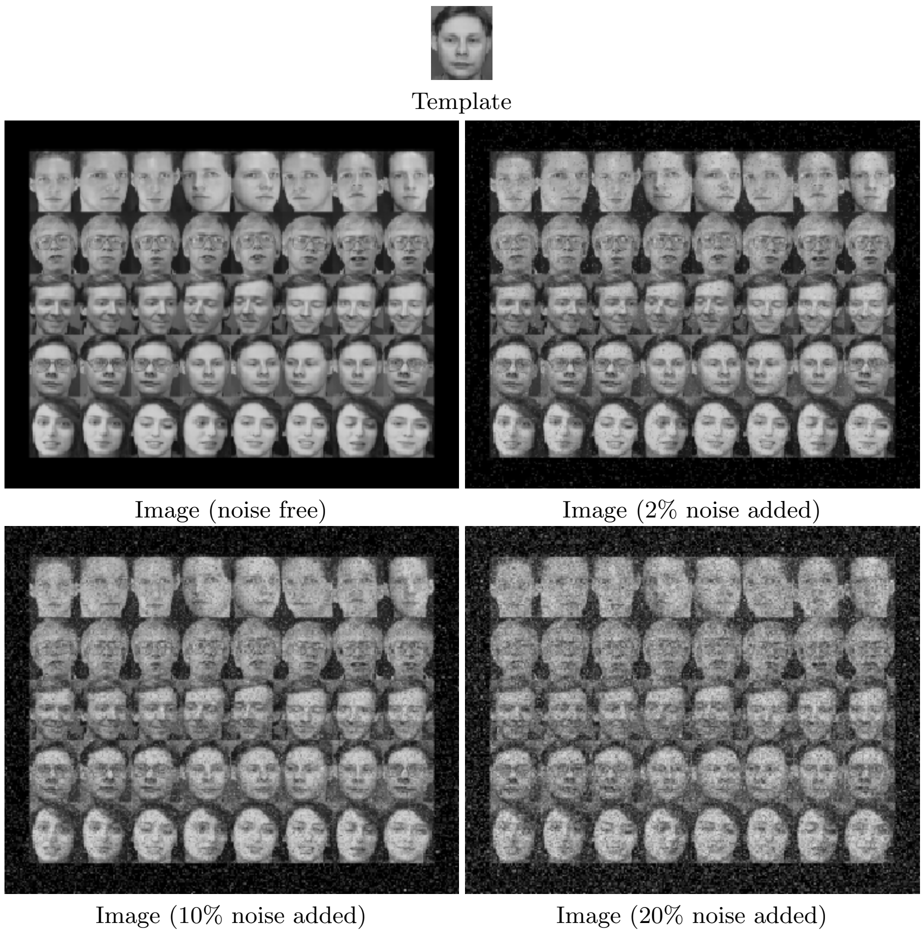
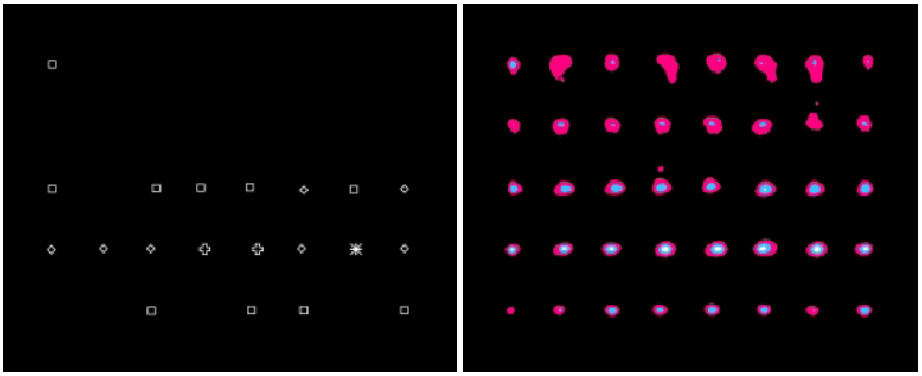


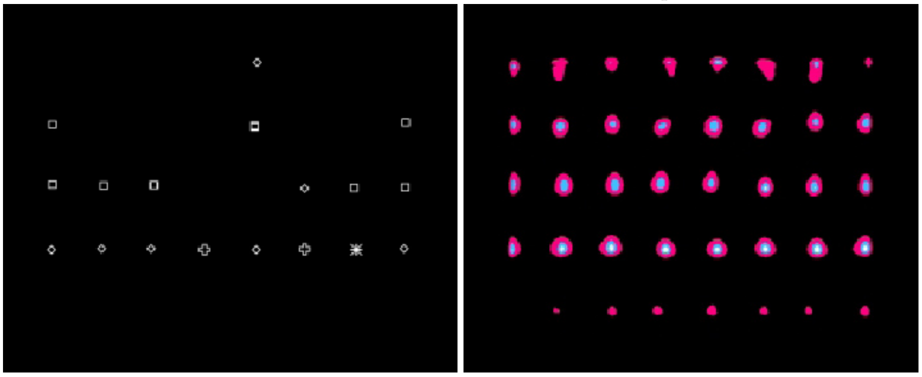
Fig. 3. The template and the image used in the experiments

Experiments: Four experiments have been run. The template and images used are shown in figure 3. Experimental results are shown in figures 4, 5, 6 and 7. The crude detection of local maxima, as shown in the left column of these figures, sometimes produces adjacent pixels as local maxima. But in all cases, the location of the top 20 maxima always detect very accurately the location of a face. In most cases, the majority of the faces on the fourth row are detected (these are the faces of the individual shown in the template).

7 Concluding Remarks

This paper describes a very fast template matching technique, generalizing the integral images approach of Viola and Jones. The implementation of the proposed technique is straightforward. It produces highly accurate approximations


Results obtained with the classic technique


Results obtained with 3rd order interpolation polynomials

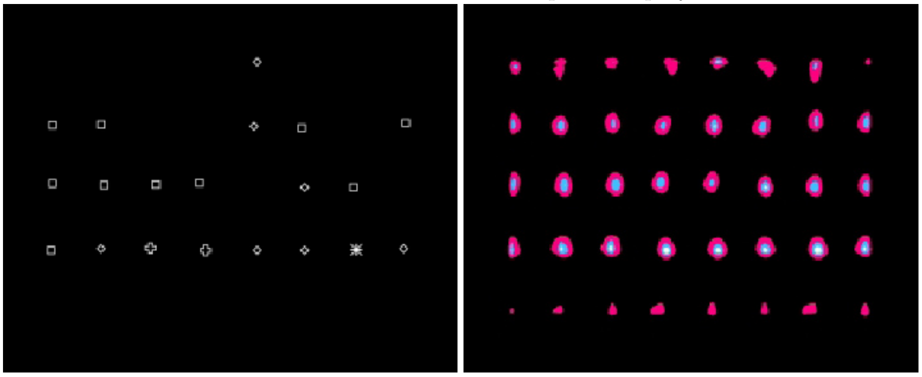
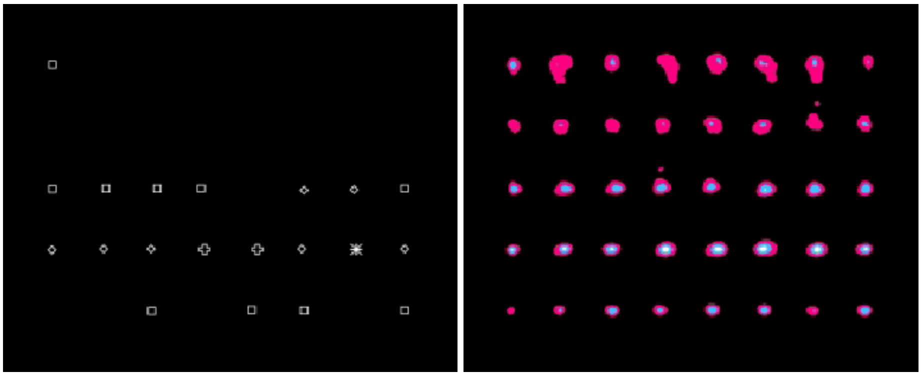
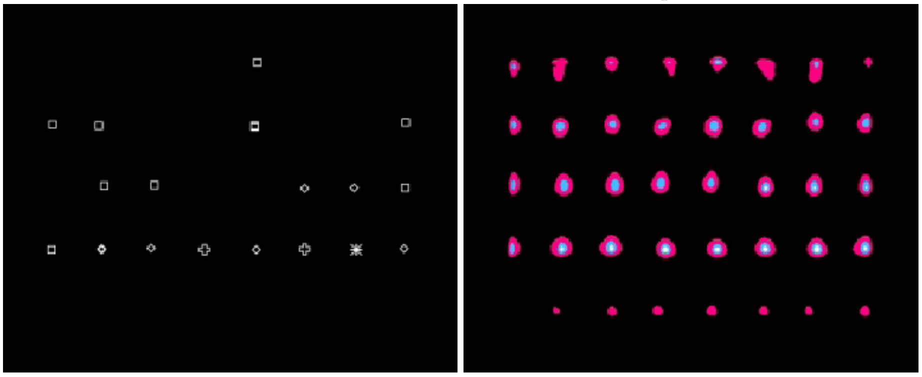

Results obtained with 2nd order interpolation polynomials

Fig. 4. Noise free results. All the top 20 local maxima (shown in the left column) are exact face locations. All faces of the individual being targeted (fourth row) are found among the top 20 matches. The color coded images in the right-hand column show that seven of the eight faces of the individual are within the top 0.1% of all match values. The best match value shown by the 8-pointed star is the correct location of the exact template.


Results obtained with the classic technique


Results obtained with 3rd order interpolation polynomials

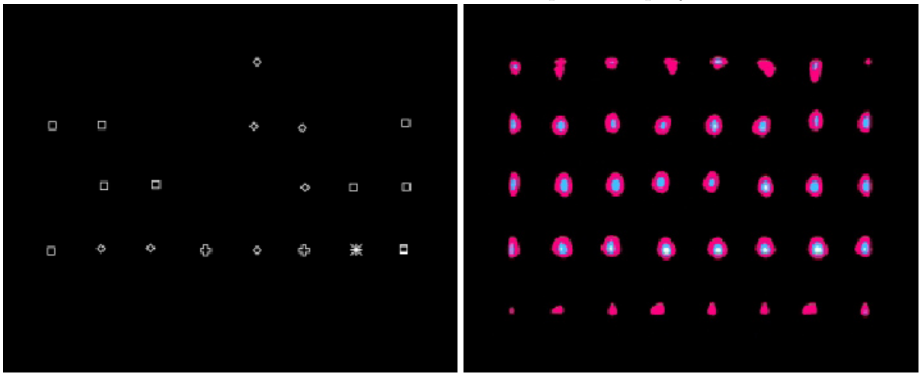
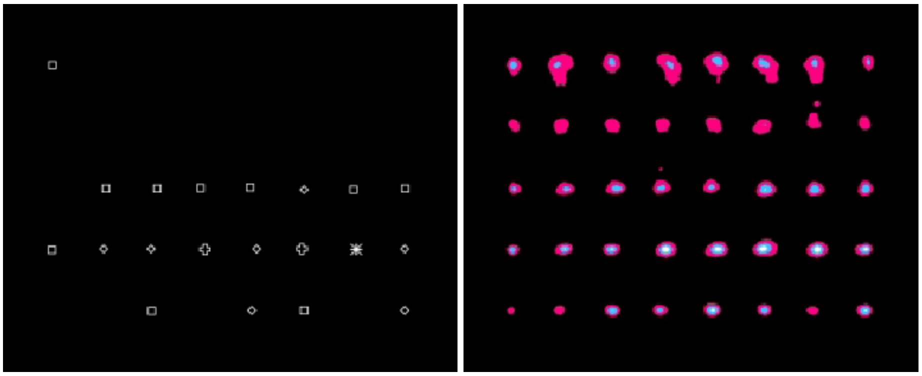
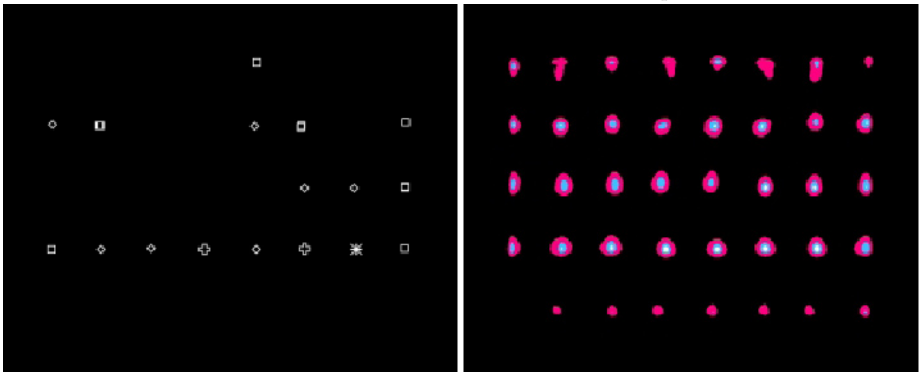

Results obtained with 2nd order interpolation polynomials

Fig. 5. Results for 2% noise. All the top 20 local maxima (shown in the left column) are exact face locations. All faces of the individual being targeted (fourth row) are found among the top 20 matches. The color coded images in the right column show that seven of the eight faces of the individual are within the top 0.1% of all match values. The best match value shown by the 8-pointed star of all match values is the correct location of the exact template.


Results obtained with the classic technique


Results obtained with 3rd order interpolation polynomials

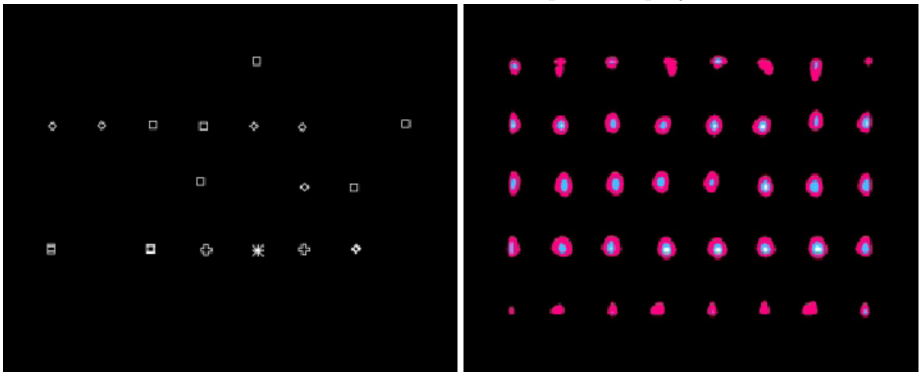
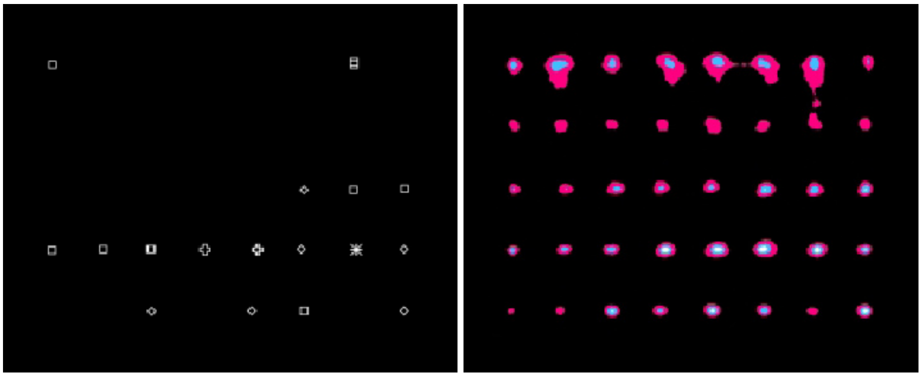
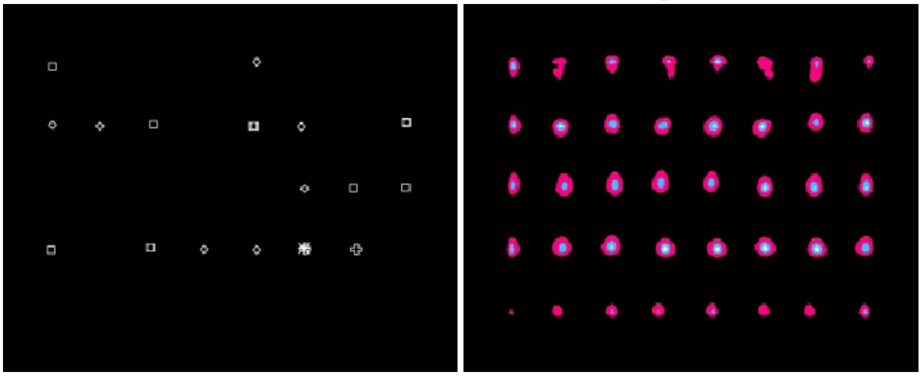

Results obtained with 2nd order interpolation polynomials

Fig. 6. Results for 10% noise. All the top 20 local maxima (shown in the left column) are exact face locations. There are only 2 misses among the faces of the individual in the template, when 2nd order interpolation is used. The color coded images in the right column show that all of the faces of the individual shown in the template are within the top 1.0% of all match values.


Results obtained with the classic technique


Results obtained with 3rd order interpolation polynomials

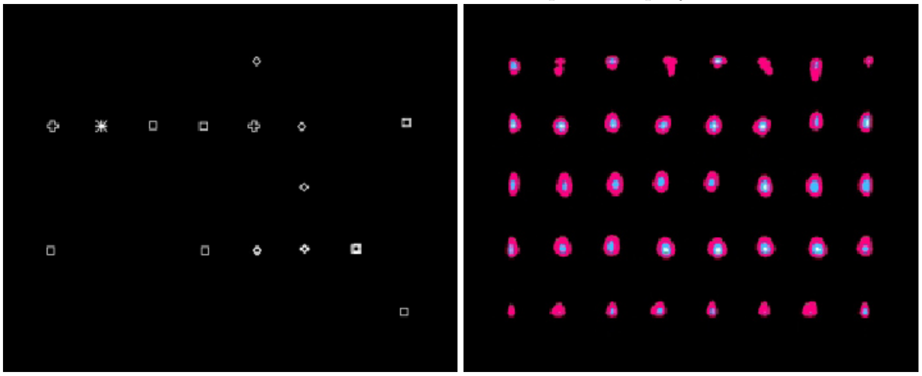

Results obtained with 2nd order interpolation polynomials

Fig. 7. Results for 20% noise. All the top 20 local maxima (shown in the left column) are exact face locations. There are only 5 misses among the faces of the individual in the template, mostly when when 2nd order interpolation is used. The color coded images in the right column show that all of the faces of the individual shown in the template are within the top 1.0% of all match values.

to the matching computed by the classic normalized correlations method in a fraction of the run time.

We have shown that the set of functions that can be computed from integral images includes the algebraic moments of low order. The results of Viola and Jones are the special case of the m_{00} moment. The question of whether or not this set of functions can be further expanded remains open. To compute other functions that are not algebraic combinations of moments one must show that they can be analytically "shifted", as shown by Eq. (5) for moments.

References

1. Rosenfeld, A., Kak, A.C.: Digital Picture Processing. Second edn. Academic Press (1982)
2. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley & Sons (1973)
3. Winograd, S.: Arithmetic Complexity of Computation. Volume 33 of SIAM CBMS-NSF. SIAM, Philadelphia (1980)
4. Davis, L., Bajcsy, R., Herman, M., Nelson, R.: RSTA on the move: Detection and tracking of moving objects from an autonomous mobile platform. In: Proceedings of the ARPA Image Understanding Workshop, Palm Springs, CA (1996) 651–664
5. Armstrong, J.B., Maheswaran, M., Theys, M.D., Siegel, H.J., Nichols, M.A., Casey, K.H.: Parallel image correlation: Case study to examine trade-offs in algorithm-to-machine mappings. The Journal of Supercomputing **12** (1998) 7–35
6. Fang, Z., Li, X., Ni, L.M.: Parallel algorithms for image template matching on hypercube SIMD computers. IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-9** (1987) 835–841
7. Goshtasby, A., Gage, S.H., Bartholic, J.F.: A two-stage cross correlation approach to template matching. IEEE Transactions on Pattern Analysis and Machine Intelligence **6** (1984) 374–378
8. Rosenfeld, A., Vanderburg, G.J.: Coarse-fine template matching. IEEE Transactions on Systems, Man, and Cybernetics (1977) 104–107
9. Yoshimura, S., Kanade, T.: Fast template matching based on the normalized correlation by using multiresolution eigenimages. In: International Conference on Intelligent Robots and Systems, Munchen, Germany (1994) 2086–2093
10. Viola, P., Jones, M.: Robust real-time object detection. Presented in the Second International Workshop on Statistical and Computational Theories of Vision, ICCV'2001 (2001) www.ai.mit.edu/people/viola/research/publications/ICCV01-Viola-Jones.ps.gz.