1-18-1995

Shape recognition method using morphological hit-or-miss transform

Prabir Bhattacharya

Weibin Zhu

Kai Qian

Shape recognition method using morphological hit-or-miss transform

Prabir Bhattacharya
Weibin Zhu
University of Nebraska–Lincoln
Department of Computer Science & Engineering
Lincoln, Nebraska 68588
E-mail: prabir@cherokee.unl.edu

Kai Qian
Georgia Southwestern College
Department of Computer Science
Americus, Georgia 31709

Abstract. We present a new shape recognition algorithm for binary images based on the morphological approach. The standard algorithms based on the morphological hit-or-miss transform use block structure patterns, but in many practical situations, this may cause problems because the size of the object is unknown. In the proposed method, we match the corners of the object instead of the block structure pattern. Because most objects have different structure elements for the corners, our method gives a more reasonable approach and the size of the object plays a less significant role in our recognition algorithm than in the standard algorithms. Our method can be easily adapted to recognize the features of different types of objects.

Subject terms: mathematical morphology; hit-or-miss transform; rank order filtering; polynomial approach for image representation; shape recognition.

Optical Engineering 34(6), 1718-1725 (June 1995).

1 Introduction

There have been extensive investigations to develop methods to process digitized pictures using mathematical morphology.[1–7] Among the various morphological operations, the morphological hit-or-miss transformation is a basic tool for the shape detection.[8–11] This transformation involves the erosion of an image with a foreground structuring element and the erosion of the complemented image with a background structuring element; the intersection of these two erosions is called the hit-or-miss transform.

Recently, Casasent et al.[9] developed an optical implementation of the morphological hit-or-miss transform, improving the standard optical pattern-recognition correlator techniques for the detection of targets in images containing clutter. They modified the standard hit-or-miss transformation to do rank order filtering, where the foreground structuring element can be the desired object, and the background structuring element is its complement with a border (background) around it. The erosion operation can be performed on an optical correlator with output correlation threshold chosen to select the desired rank order filter. The advantage of their approach over the standard hit-or-miss transform is that it gives some flexibility in choosing the size of the templates. Their approach, however, still requires some preprocessing to get the approximate size of the target, and then one has to choose a set of structure elements whose sizes are close to the size of the desired object. Thus, it is necessary for each object to maintain a large database containing templates of varied sizes. Because the size of an object changes, in general, from image to image, we cannot always apply this algorithm efficiently.

In this paper, we propose an extension of the hit-or-miss algorithm of Casasent et al.[9] Our proposed method does not require any preprocessing to compute the size of the target. Our approach is to recognize the corners of an object instead of the whole object itself. By recognizing the corners of an object, we are able to recognize the shape of the object itself, particularly, when the object has a boxlike shape. Detecting corners of images (binary and gray) has been investigated by several authors.[12–15] We think, however, that there has been no previous work to detect corners using the hit-miss transform and to apply the information to recognize shapes. A novel feature of our algorithm is that the size of the object is relatively unimportant in the recognition process. Thus, the same set of templates could be used to recognize objects of similar shapes but of different sizes. In addition, it can be applied to match the unknown object with the given models (e.g., Ref. 16)—the recognition process would involve the calculation of the highest percentage of match when we apply the hit-or-miss transform on the unknown image for the templates corresponding to each model. Our proposed method works particularly well for the recognition of objects with boxlike shapes (such as military vehicles, submarines, and

Paper 22104 received Oct. 27, 1994; revised manuscript received Jan. 17, 1995; accepted for publication Jan. 18, 1995.

certain industrial parts). For curved surfaces also, the algorithm can detect the corners (if any). Because the templates designed for recognizing a given shape cannot be applied to match other shapes, the proposed algorithm is expected to recognize the shape correctly. We have included the implementation of the proposed algorithm for the recognition of the muzzle of a tank gun and also the sail (i.e., the funnel-like top part) of a submarine.

In the proposed method, we have used a certain polynomial representation of images.[16–25] It is not necessary to use this algebraic formulation to develop the proposed method and the standard array representation of images would work equally well. The possible advantage of the polynomial approach, however, lies in the fact that if we use the fast Fourier transform (FFT) to compute the convolutions of polynomials involved in the processing, considerable speed-up is obtained (theoretical estimates of the speed-up have been given in Refs. 16 and 17). The polynomial approach has been used extensively in the past to process binary, gray, colored, and 3-D voxel-based images. Various image processing operations have been defined in terms of a combination of algebraic operations on the polynomial representations of the image and a set of suitably chosen template polynomials. Thus the polynomial approach provides a versatile methodology for performing low-level vision tasks. The morphological operations of dilation and erosion can be performed very conveniently in the polynomial approach.[16,17,21] Because the hit-or-miss transform uses dilation and erosion as the basic operations, we can implement the hit-or-miss transform (and its possible extensions) using the algebraic approach.

In Sec. 2, we review some relevant background material. In Sec. 3, we present the proposed hit-or-miss transform. In Sec. 4, we consider the implementation of the proposed method. Section 5 gives our summary and conclusions.

2 Background

2.1 *Morphology and Hit-Miss Transformation*

The morphological operations can be viewed as shape filtering operations by successive transformations that remove the information from image and see how they relate to the shape of structuring element and retaining only the information of interest in the image. For a detailed background in morphology consult Refs. 1 to 7. Here we give the basic definitions only. The two basic operations in mathematical morphology are the erosion and dilation.[1] The definition for the erosion of an image X with a structuring element B is

$$X \ominus B = \{a : B_a \subseteq X\} \ , \tag{1}$$

where $\ominus$ denotes the erosion, B_a is a set B shifted by a, and $\subseteq$ denotes a subset. The definition for the dilation of an image X with a structuring element B is

$$X \oplus B = \{a : B_a \cap X \neq \emptyset\} \ , \tag{2}$$

where $\oplus$ denotes dilation and $\cap$ is the intersection (i.e., the AND operation).

The traditional hit-or-miss transformation is defined[1] as

$$X(B,D) = [(X \ominus B) \cap (X^c \ominus D)] \ , \tag{3}$$

where we denote a binary image by X, its complement by X^c, and a foreground structuring element by D. The hit-or-miss transformation uses erosion and dilation operations, and it also uses the foreground and background images. Figures 1(a) and 1(b) give an illustration of the hit-or-miss transform, where Fig. 1(a) is the foreground image of the rectangle X (basically, it is the binary image of the rectangle itself), and Fig. 1(b) is the background image X^c of X [note that in Fig. 1(b) the corners of the outer rectangle appear to be slightly curved because of low resolution, but it should be regarded as rectangular]. Compared to X, the black rectangle in Fig. 1(a) turns into a white rectangle but a few pixels width boundary around X is presented. We get the boundary pixels of the object, but they are not ordered along the contour. The width of the boundary is application dependent.

In Casasent et al.,[9] a new hit-or-miss transformation is proposed for noisy objects, using rank order filtering. According to their method, with noisy images, one performs the operation $(X \ominus B)$ with a slightly larger object than the actual object itself. For example, in Fig. 1(a), we can have a bigger rectangle with two or three pixels wider; and for the $(X^c \ominus D)$ operation the background image can be slightly smaller, $B \cap D \neq \emptyset$. Their implementation indicates good results. The advantage of their approach is that it allows more flexibility in choosing the size of templates compared to the traditional hit-or-miss operation.

2.2 *Image Representation by Polynomials*

To represent the templates and binary images, we use a polynomial representation. In this approach, a binary image of size $r \times s$ is represented by a polynomial in two variables:

$$\sum_{ij} a_{ij} X^i Y^j \ , \tag{4}$$

where the summation is over all the pixels of the image and a_{ij} is 1 or 0 according to whether the pixel is black or white, respectively. Similar algebraic representation is also used for the templates. The morphological operations of dilation and erosion can be performed using a polynomial approach; the details are given in Refs. 17 and 21. We review it briefly in the following paragraph.

Let A and B be the picture polynomials of a binary image and a structuring element (SE) respectively. Let

$$P = A * B \equiv \sum_{ij} p_{ij} X^i Y^j \tag{5}$$

denote the usual polynomial product of A and B. Then the dilation $\oplus$ and erosion $\ominus$ are obtained as follows:

$$A \oplus B = \sum_{ij} c_{ij} X^i Y^j, \quad A \ominus B = \sum_{ij} d_{i,j} X^i Y^j \ , \tag{6}$$

where $c_{ij} = 1$ if $p_{ij} \geq 1$ and 0 otherwise, and $d_{ij} = 1$ if $p_{ij} = n$ and 0 otherwise, where n is the number of nonzero terms in the polynomial B. Because the primitive operations of hit-or-miss transformation are dilation and erosion, it follows that we can use the polynomial representation for the hit-or-miss transformation. The polynomial approach has been used extensively for image processing and recognition of binary, gray, colored, and voxel-based 3-D images.[16–24]

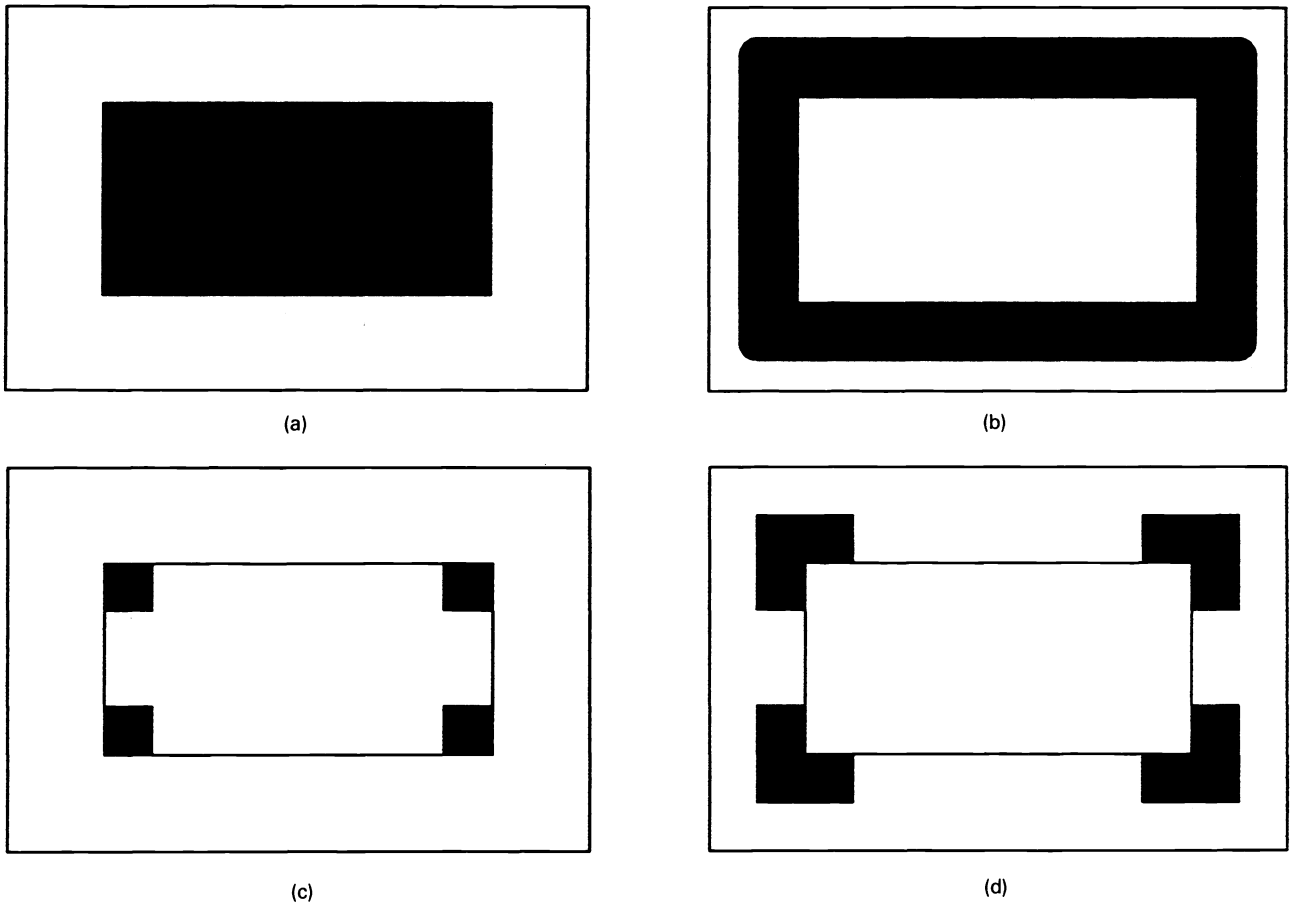


Fig. 1 Illustration of the hit-or-miss transform. For the traditional transform, (a) is an image (foreground template), (b) is the background template. For the proposed transform, (c) and (d) are the foreground and background templates for image (a).

3 Proposed Method

3.1 *Outline*

The main idea of the algorithm is to match an unknown object of dynamic size by matching its corners with a set of templates, instead of matching the entire object itself. Each corner of the object corresponds to two templates, one for the foreground and the other for the background. We apply the standard hit-or-miss operation using this set of templates to match each corner separately—the foreground templates are used for the hit operation and background templates are for the miss operation. We illustrate the basic idea by taking a straightforward model: Figs. 1(c) and 1(d) are the foreground and background templates, respectively, for the proposed algorithm to recognize four corners of the rectangle in Fig. 1(a). Figure 2 illustrates the proposed transform for a model of submarine [Fig. 2(a)] and gives the corresponding foreground and background templates [Figs. 2(b) and 2(c), respectively). The matrix forms of the templates corresponding to Fig. 2 are given in Fig. 3 where each corner i has two templates denoted by $\mathbf{B}_i$ and $\mathbf{D}_i$ ($1 \le i \le 4$)—we explain the motivation behind their design in Sec. 3.

In practice, because of noise it is often too difficult to successfully match the pixel corresponding to the actual geometric corner; the corner pixel may be even missing from the image. Thus, we have made a further modification as follows: instead of matching a corner, we have, in practice, designed templates to match a pixel that lies close to the geometric corner under consideration and within the object itself. We call such a pixel a pseudo-corner. In our method, it is assumed that we have some idea about the group of objects we are trying to recognize and so the templates are designed using this *a priori* information. This approach is along the lines of the widely used model-based vision approach for object recognition[26] and we have used this principle in some of our previous works using the polynomial representation of images.[16,22]

3.1.1 *Design of templates*

In the foreground template matrix (i.e., the array representing the template) the idea is to capture the essential features of the corner pixel. This is attempted by mapping the corner pixel and its neighbors to the entries of the template matrix with corresponding values. For the background template matrix, we similarly try to capture the feature of complement of the corner pixel and its neighbors. Note that the two template matrices may not be of the same sizes. The templates used in the algorithm are generated manually for each class of images—in Sec. 4, we describe in detail the templates that we used to recognize two classes of images.

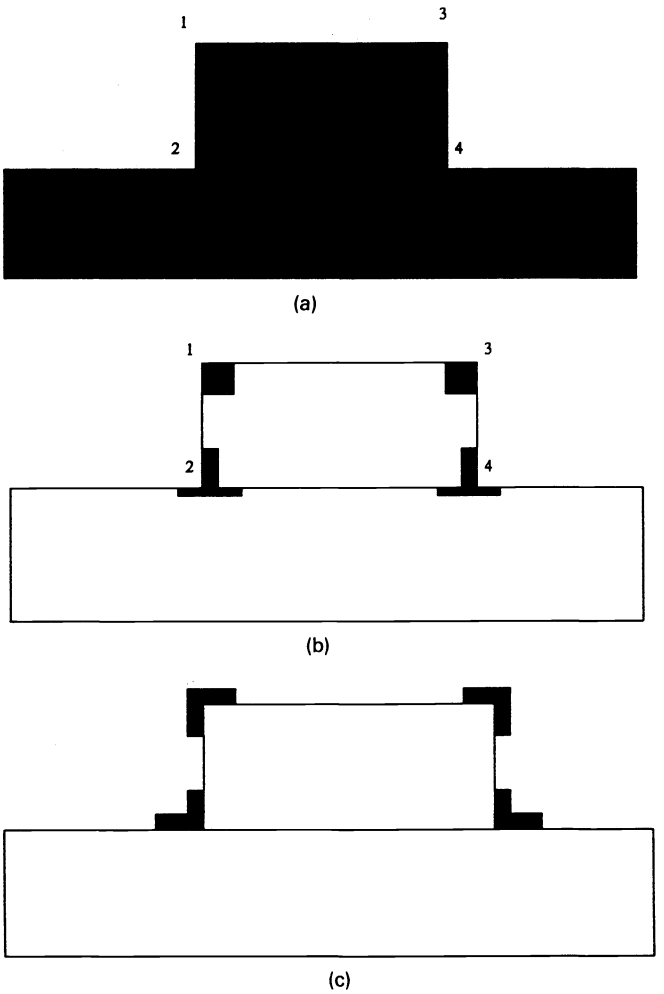


Fig. 2 Illustration of the proposed transform: (a) is the shape of a submarine and (b) and (c) are the foreground and background templates.

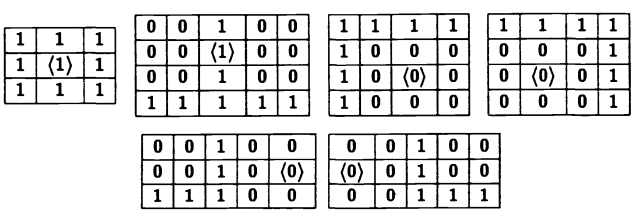


Fig. 3 Matrix form of the templates for the image in Fig. 2(a): matrices $\mathbf{B}_1 = \mathbf{B}_3$, $\mathbf{B}_2 = \mathbf{B}_4$, $\mathbf{D}_1$, $\mathbf{D}_3$, $\mathbf{D}_2$, and $\mathbf{D}_4$, respectively.

3.1.2 *Advantages of our method*

The traditional hit-or-miss transform requires two large templates with sizes $n \times m$ and $n+i \times m+i$, where $n \times m$ is the size of the foreground template and $n+i \times m+i$ is the size of the background template with i pixel wide outlay. Thus, if the object has large size, the process requires relatively large memory space to compute. On the other hand, our proposed method only requires much smaller constant-size templates. Because our algorithm matches corners instead of the whole object, and each corner is independent of each other corner, the method can be directly adapted to run on a single-instruction multiple-data (SIMD) parallel computer. Because we are matching the corners, the size of the unknown object is irrelevant. Thus, we can actually calculate the size of the object after we match the corners.

3.2 *Recognition Process*

Suppose we have an object (assumed to be box shaped) with n corners. For each corner we choose a pair of templates $\mathbf{B}_i$ and $\mathbf{D}_i$, where $\mathbf{B}_i$ denotes the foreground template and $\mathbf{D}_i$ denotes the background template ($1 \le i \le n$). Using these templates the hit-or-miss transform given by Eq. (3) can be expressed as

$$X(\mathbf{B}_i, \mathbf{D}_i) = [(X \ominus \mathbf{B}_i) \cap (X^c \ominus \mathbf{D}_i)] \quad (1 \le i \le n) \ . \tag{7}$$

The polynomial approach for image representation gives the following formulation of Eq. (7): after implementing the hit-miss transform algebraically in the manner described in Sec. 2.2, the hit-or-miss transform given by Eq. (7) for each corner, we calculate the polynomial

$$X_{\text{final}} \equiv \sum_{i=1}^{n} X(\mathbf{B}_i, \mathbf{D}_i) \ , \tag{8}$$

where the summation is the usual addition of polynomials. In the polynomial X_{final}, we note the terms corresponding to the first n highest coefficients. Clearly, these terms would correspond to the n corners of the unknown image. We can also obtain algebraically the specific positions (i.e., coordinates in the give rectangular grid system) of the corners of the unknown object from the polynomial X_{final}. If $a_{ij} X^i Y^j$ is a term such that a_{ij} is one of the n highest coefficients, then there exists a corner at the (i,j) position. Knowing the positions of all the corners, we can estimate the size of the unknown object.

We now give the pseudo-code of the proposed method.

```
ALGORITHM: CORNER DETECTION
int col, row
Pixel DetectedCorners[NUMBER_CORNER];
  for (col = 0; col<NUMBER_COL; col + + )
    for (row = 0; row<NUMBER_ROW; row + + )
      for    (corner = 0;    corner<NUMBER_CORNER;
      corner + + )
 Polynomial[col,row] =
       HIT(image(col, row), ForegroundTemplate[corner])
 MISS(backgroundImage(col, row), BackgroundTemplate
 [corner]);
 DetectedCorners = Top(Polynomial, NUMBER_COL);
```

3.3 *Additional Steps*

After recognizing the corners, a few clean-up steps can be taken as follows: We can use a standard filter to reduce the noise of the image. Further, to determine if there is a solid object bounded by the corners, we can check the coefficients of those terms in the polynomial representation of the image that correspond to the pixels inside the boundary. If the corners are given, then it is easy to draw the boundary between the corners. We can improve the proposed method by using a good thresholding procedure,[27] but in the implementation described in Sec. 4 we used only arbitrary thresholding as

was done in Ref. 9. Further, we have assumed in our design of the templates that the unknown object appears in a frontal view (i.e., has the aspect angle zero). This is not a stringent assumption because we are eventually selecting the best match by choosing the one with the highest percentage. If we wish, we could estimate the aspect angle of the object.[22]

4 Implementation

First, we describe the implementation for a group of images of submarines. Figure 2(a) shows a model of a submarine. Note that a distinctive feature of a submarine is the shape of its "sail," which is of rectangular shape and connected to the base of the submarine.* In Fig. 2, we have numbered the four corners of the sail 1 to 4. Figure 2(b) shows the foreground templates, and Fig. 2(c) gives the background templates. We define for each corner a pair of arrays templates $\mathbf{B}_i$ and $\mathbf{D}_i$ ($1 \le i \le 4$), where $\mathbf{B}_i$ (the foreground template matrix) is used to match the foreground, and $\mathbf{D}_i$ (the background template matrix) is used to match the background. In the design of these templates, we try to match not the actual corner pixel but actually a pixel close to it that lies on the "sail." We call this pixel a pseudo-corner. The idea behind this is to compensate for the noise in the immediate neighborhood of the corner, because if we match only the corner point, then there may be a possible error caused by the noise (the corner pixel may even be missing because of some blur). In Fig. 2, we give the templates we used; the motivation is given next for each template we have marked the entry corresponding to the pseudo-corner with the symbol $\langle\ \rangle$.

Motivation for the templates. Notice that in Fig. 2, the pairs of corners (1,3) and (2,4) have the same profiles for the foreground. Thus we assign the same foreground templates for each of these pairs, in other words, $\mathbf{B}_1 = \mathbf{B}_3$ and $\mathbf{B}_2 = \mathbf{B}_4$. For the array $\mathbf{B}_1$, however, the (1,1) entry corresponds to the actual corner, whereas for $\mathbf{B}_3$ the (1,3) entry corresponds to the actual corner. In $\mathbf{B}_2$ or $\mathbf{B}_4$, the last row corresponds to the pixels in the base of the submarine, and each entry of these rows is assigned the value 1. In $\mathbf{B}_2$ the (3,2) entry corresponds to the actual corner corresponding to the corner numbered 2 in Fig. 2. In $\mathbf{B}_4$, the (3,4) entry corresponds to the actual corner corresponding to the corner numbered 4 in Fig. 1. In $\mathbf{D}_1$, the first row and first column represent the pixels in the background, which are adjacent to the contour of the tower at the corner numbered 1; similar motivation applies to $\mathbf{D}_2$, $\mathbf{D}_3$, and $\mathbf{D}_4$.

Figure 4 shows a simplistic model of a tank gun and turret where the corners of the gun are numbered 1 through 4; the corners numbered 3 and 4 are the ones that are joined to the turret. Figure 5 gives the templates we used, where for each template the entry corresponding to the pseudo-corner is indicated by the symbol $\langle\ \rangle$. The motivation behind the design of these templates can be seen as follows. In Fig. 4, notice that the pairs of corners (1,2) and (3,4) have the same profiles for the foreground, so we assign the same foreground templates for each of these pairs, in other words, $\mathbf{B}_1 = \mathbf{B}_2$ and $\mathbf{B}_3 = \mathbf{B}_4$. For the array $\mathbf{B}_1$, however, the (1,3) entry corresponds to the actual corner, whereas for $\mathbf{B}_2$ the (3,3) entry corresponds to the actual corner. In $\mathbf{B}_3$ or $\mathbf{B}_4$, notice that the first column corresponds to the pixels on the turret adjacent to the gun, and these are all assigned the value 1. The (4,2) entry on $\mathbf{B}_3$ or $\mathbf{B}_4$ corresponds to the actual corners (3,4) (see Fig. 4). In $\mathbf{D}_1$, the first row and last column represent the pixels in the background, which are adjacent to the contour of the gun at the corner 1; similar motivation applies to $\mathbf{D}_2$, $\mathbf{D}_3$, and $\mathbf{D}_4$. These templates have been designed here under the assumption that the tank gun is on the horizontal position—this hypothesis may be too simplistic in practice. Later in the next subsection, we consider the situation where the tank gun has a nonzero elevation above the ground level. The polynomial representations of the various templates can be obtained easily.

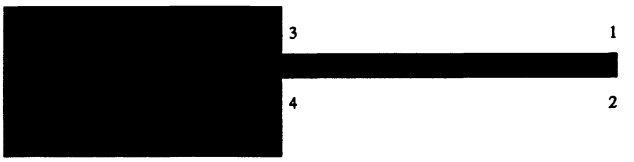


Fig. 4 Shape of a tank gun and turret.

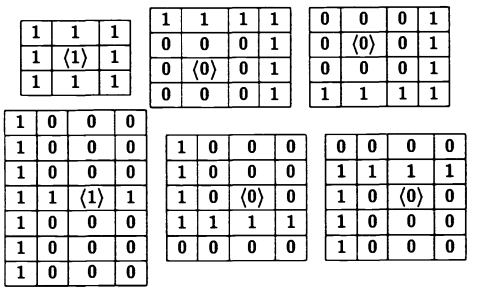


Fig. 5 For tank images in Figure 4: templates for $\mathbf{B}_1 = \mathbf{B}_2$, $\mathbf{D}_1$ and $\mathbf{D}_2$, $\mathbf{B}_3 = \mathbf{B}_4$, and $\mathbf{D}_3$ and $\mathbf{D}_4$, respectively.

The implementation of the proposed method was carried out using the standard image processing package [28] *Khoros* developed by the University of New Mexico that was run on a SUN4 workstation. We tested the algorithm on a set of submarine images and a set of tank images, using the appropriate templates. Figures 6(a) through 6(e) give five images of submarines of various types—we name these images sub1 through sub5, respectively. Figures 7(a) through 7(f) show six tank images, which we number as tank 1 through tank 6. We thresholded the images in a manner similar to that in Casasent et al.[9] The idea is to use a suitable threshold to highlight the objects. In Table 1 (Table 2), the symbols t_i ($1 \le i \le 4$) denote the four corners of a submarine tower (tank gun) corresponding to the corners i ($1 \le i \le 4$) shown in Fig. 2 (Fig. 4).

Note that tank 6 [Fig. 7(f)] is a disconnected model from Casasent et al.[9] Because the tank gun is disconnected from the turret, after thresholding according to their paper, we made slight changes in our definition of templates in this special case, we omit the details. Notice that the sizes of the various images in Figs. 6 and 7 are different. Also, some images in Fig. 7 have orientations different from the rest—in this case, the templates have been modified slightly to

*In the submarines of some countries there are also two small fin-like elements protruding from the top-part of the "sail," but for simplification we have not considered such designs here.

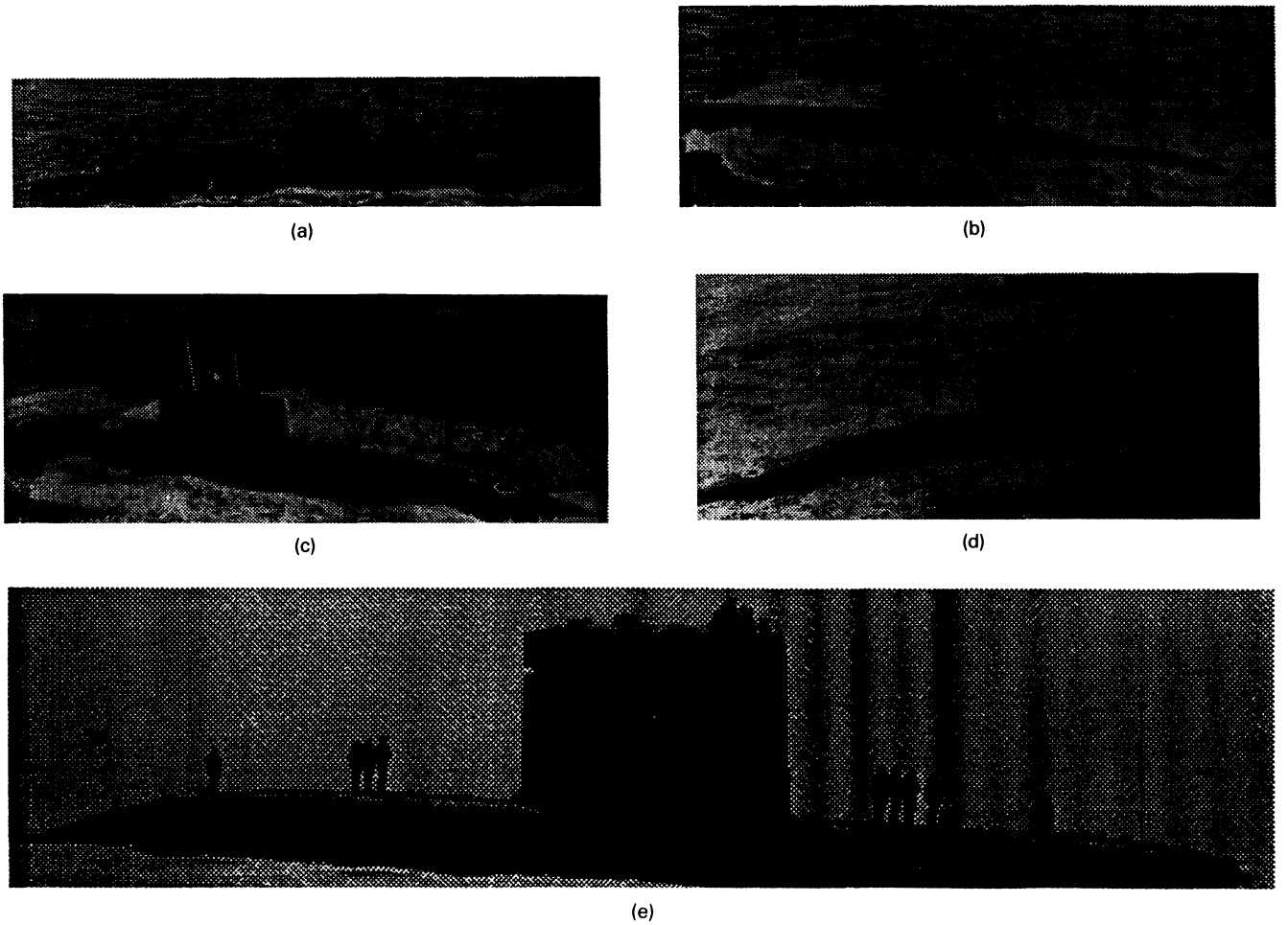


Fig. 6 (a) to (e) Five different submarine images, sub1 to sub5.

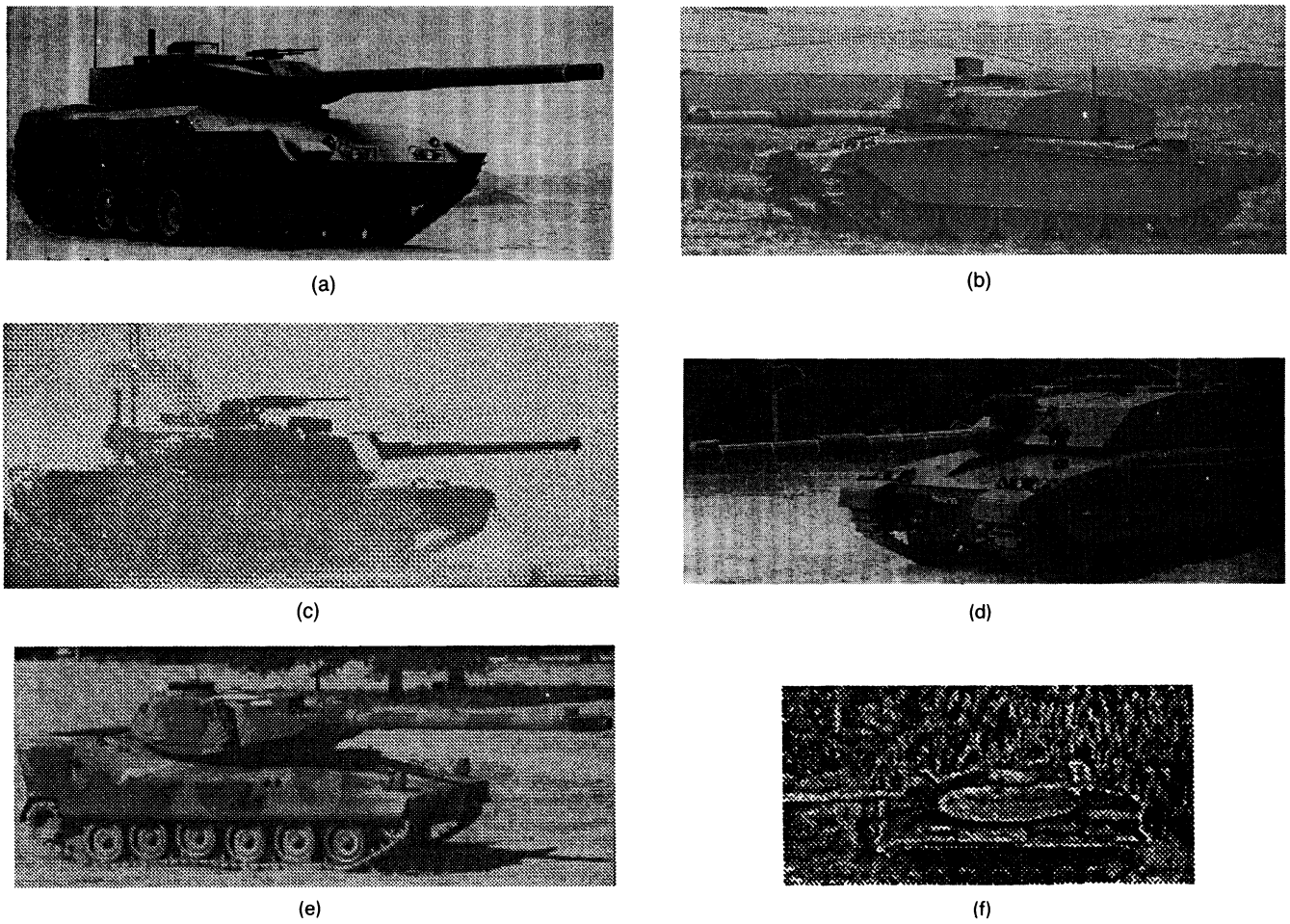
accommodate the different orientation. Thus, the implementation indicates clearly that the proposed method is capable of handling dynamically images of various sizes and different orientations. It is easy to design templates for the recognition of other classes of objects based on the motivation described here, and then apply our algorithm.

4.1 *Further Modifications*

The given results show that our algorithm works nicely. In many situations, however, tanks do not have guns in the horizontal position. To recognize the tank guns with elevated angles, we now add some further steps to make the algorithm work more realistically. First, we need to decide on the trade-off between accuracy and efficiency. If the objective is to obtain very accurate elevating angles, it will cost extra memory and speed for the computing. In our example, we decided on a cutoff angle of 10 deg. Thus, we rotate the image in 10-deg steps, using a rotation procedure in the algebraic formulation.[16,22] We cannot just rotate the templates because after rotation, the dimensions of each template will change, and more importantly the rotated array will have extra blank pixels within the templates that will make it difficult to run the hit-miss transformation. Thus we could rotate the whole image instead to avoid this problem (but this, of course, would be a very slow process, but see the preceding remark concerning the trade-off between accuracy and efficiency). We then apply the proposed method to match each rotated image. During each match, we need to check the false alarm and we eliminate the match that is not of approximately rectangular shape. Table 3 gives the false alarm rates for the tank images 1 to 3, where as before t_i denotes the corner that is numbered i in Fig. 4 ($1 \leq i \leq 4$). After computing all the matches, we select the best match. The angle of this match also corresponds to the degree by which the image has been rotated.

5 Summary and Conclusions

Our paper is a follow-up of a recent work by Casasent et al.[9] We have developed a modification of the hit-or-miss morphological transform to recognize the shapes of objects where we match the corners of the object instead of the block structure pattern. The advantage of this approach is that matching the corners of the object makes the size of image relatively unimportant for the recognition. The proposed method works nicely for objects with polyhedral shapes. Implementation has been included for recognizing the "sail" of a submarine and also the muzzle of a tank gun. We have also computed the false alarm rates for some of the images. The results clearly indicate that the algorithm recognizes shapes correctly but that rotating the image may be a slow process. It can be easily adapted to recognize the features of other types of images by choosing different templates. The method works particularly well to recognize box-shaped objects. We have



(a) (b) (c) (d) (e) (f)

Fig. 7 (a) to (f) Six different tank images, tank 1 to tank 6.

Table 1 Statistics for submarine images.

	t_1	t_2	t_3	t_4
sub 1	100%	100%	93.75%	100%
sub 2	93.75%	100%	87.5%	100%
sub 3	93.75%	100%	87.5%	76.92%
sub 4	100%	100%	87.5%	100%
sub 5	93.75%	100%	87.5%	100%

Table 2 Statistics for tank images.

	Size	t_1	t_2	t_3	t_4
tank 1	263x107	75%	93.5%	80.9%	71.4%
tank 2	340x140	93.5%	93.5%	85.7%	76.2%
tank 3	338x147	87.5%	87.5%	90.6%	80.9%
tank 4	240x102	87.5%	100%	90.6%	90.5%
tank 5	285x111	81.5%	93.5%	90.6%	90.5%
tank 6	320x120	75.0%	75.0%	56.0%	75.0%

used an algebraic approach to represent the images and to apply the proposed transform. This has the advantage that the polynomials representing images and templates can be operated easily and a considerable speed-up is possible if we use the well-known FFT to compute the multiplications of polynomials involved in the algorithm. The proposed algorithm has an advantage over the standard hit-or-miss transform for real-time computation because it uses smaller,

Table 3 False alarm rates for (a) to (c) tanks 1 to 3, respectively.

	0 deg	11.5 deg	22.5 deg	45 deg
t_1	87.5%	70.5%	False	False
t_2	100%	81.5%	False	False
t_3	90.6%	87.5%	False	False
t_4	90.5%	81.5%	False	False
result	select	drop	drop	drop

(a)

	0 deg	11.5 deg	22.5 deg	45 deg
t_1	91.5%	False	False	False
t_2	81.5%	False	False	False
t_3	100%	False	False	False
t_4	90.5%	False	False	False
result	Select	drop	drop	drop

(b)

	0 deg	11.5 deg	22.5 deg	45 deg
t_1	78.5%	False	56.6%	False
t_2	91.5%	False	76.8%	False
t_3	87.5%	False	81.5%	False
t_4	90.5%	False	45.5%	False
result	Select	drop	drop	drop

(c)

constant-sized templates. In addition, it can be implemented directly on a SIMD parallel computer.

Acknowledgments

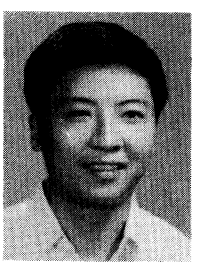
This research has been supported by the U.S. Air Force Office of Scientific Research under grant No. AFOSR F49620-94-1-0029 and also by matching support by the Center for Communication and Information Sciences, University of Nebraska-Lincoln.

References

1. J. Serra, *Image Analysis and Mathematical Morphology,* Academic Press, London (1982).
2. J. Serra, Ed., *Image Analysis and Mathematical Morphology II: Theoretical Advances,* Academic Press, London (1988).
3. P. Maragos, "Tutorial on advances in morphological image processing and analysis," *Opt. Eng.,* **26,** 623–632 (1987).
4. M. Schmitt and L. Vincent, *Morphological Image Analysis: A Practical Handbook,* Cambridge University Press, New York (1994).
5. R. M. Haralick, Ed., *Mathematical Morphology: Theory and Hardware,* Oxford University Press, New York (1994).
6. R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-9,** 532–550 (1987).
7. R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision,* Vol. 1, Chaps. 5–6, Addison-Wesley, Reading, MA (1992).
8. D. Bloomberg and P. Maragos, "Generalized hit-miss operations," *Proc. SPIE* **1350,** 116–128 (1990).
9. D. Casasent, R. Schaefer, and R. Sturgill, "Optical hit-or-miss morphological transform," *Appl. Opt.* **31**(29), 6255–6263 (1992).
10. D. Casasent and R. Schaefer, "Multilevel hit-miss transform for object detection," *Proc. SPIE* **2353,** 2–9 (1994).
11. E. R. Dougherty and R. P. Loce, "Optimal mean-absolute-error hit-or-miss filters: morphological representation and estimation of the binary conditional expectation," *Opt. Eng.* **32,** 815–827 (1993).
12. R. Mehrotra, S. Nichani, and N. Ranganathan, "Corner detection," *Pattern Recog.* **23**(11) 1223–1233 (1990).
13. J. A. Noble, "Finding corners," *Image Vis. Comput.* **6**(2), 121–128 (1988).
14. K. Rangarajan, M. Shah, and D. V. Brackle, "Optimal corner detector," in *Proc. IEEE Intl. Conf. Computer Vision,* pp. 95–102 (1982).
15. L. Kitchen and A. Rosenfeld, "Gray level corner detection," *Pattern Recog. Lett.* **1,** 95–102 (1982).
16. K. Qian, X. Lu, and P. Bhattacharya, "A 3D object recognition system using voxel representation," *Pattern Recog. Lett.* **13,** 725–733 (1992).
17. P. Bhattacharya, K. Qian, and X. Lu, "An algebraic approach to morphological operations on 2D and 3D images," *Pattern Recog.* **26,** 1785–1796 (1993).
18. K. Qian and P. Bhattacharya, "Binary image processing by polynomial approach," *Pattern Recog. Lett.* **11,** 395–403 (1990).
19. K. Qian and P. Bhattacharya, "An algebraic method for processing colored images," *Pattern Recog. Lett.* **12,** 805–811 (1991).
20. K. Qian and P. Bhattacharya, "A template polynomial approach for image processing and visual recognition," *Pattern Recog.* **25,** 1505–1515 (1992).
21. K. Qian and P. Bhattacharya, "Determining holes and connectivity in binary images," *Comput Graph.* **16,** 283–288 (1992).
22. P. Bhattacharya, "Estimation of aspect angles of targets in forward-looking infrared images," *Opt. Eng.* **13**(10), 3334–3341 (1994).
23. P. Bhattacharya, "Parallel optical imge processing by the image-logic algebra," *Appl. Opt.* **33**(26), 6142–6145 (1994).
24. B. T. Lerner and H. Thomas, "Extensions of algebraic operations: an approach to model based vision," *Comput. Electr. Eng.* **17,** 181–190 (1991).
25. T. Agui, M. Nakajima, and Y. Arai, "An algebraic approach to the generation and description of binary pictures," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-4,** 635–641 (1982).
26. W. Grimson, *Object Recognition by Computer,* MIT Press, Cambridge, MA (1991).
27. P. K. Sahoo, S. Soltani, A. K. C. Wong, and Y. C. Chen, "A survey of thresholding techniques," *Comput. Vis. Graph. Image Process.* **41,** 233–260 (1988).
28. K. Konstantinides and J. Rasure, "The Khoros software development for image and signal processing," *IEEE Trans. Image Process.* **3**(3), 243–252 (1994).

Prabir Bhattacharya received BA (Honors) in 1967 and MA in 1970, both in mathematics, from the University of Delhi, India, and a PhD in 1979 from the University of Oxford, UK, specializing in group theory. He is currently a professor at the University of Nebraska–Lincoln, Department of Computer Science and Engineering, which he joined in 1986 as an associate professor. His past assignments included extended visits to the Center for Automation Research, University of Maryland, College Park; Wright Patterson Air Force Base, Dayton, Ohio; European Molecular Biology Laboratory, Heidelberg, Germany; and the Tata Institute for Fundamental Research, Bombay, India. His current research interests include computer vision, image processing, multidimensional transforms, and parallel computing. He has published 53 refereed journal publications and coedited one monograph on vision geometry. He is on the editorial boards of *Pattern Recognition* and the *Journal of Computing and Information.* He is a fellow of the Institute of Mathematics and Its Applications (UK), a chartered mathematician (UK), and a senior member of the IEEE.

Weibin Zhu received a BS (Honors) in computer science in 1992 from Georgia Southeastern College, Americus, Georgia. He completed an MS in computer science at the University of Nebraska–Lincoln in December 1994. Since January 1995 he has been a systems network analyst at Paranet, Inc., Oakbrook, Illinois.



Kai Qian received a BS in electrical engineering from the Harbin Institute, China, an MS in computer science from East China Normal University, and his PhD in computer science from the University of Nebraska–Lincoln in 1990. He joined the Department of Computer Science, Georgia Southwestern College, Americus, Georgia, in 1990 where he is now an associate professor. His current research interests include computer vision, design and analysis of experiments, and database theory.