WL-TR-97-1016

ADAPTIVE MODEL BASED ATR SYSTEM

KATSUSHI IKEUCHI
ROBERT COLLINS
TAKESHI SHAKUNAGA
KHOTARA OHBA
DEAN POMELEAU
MARK WHEELER
TAKU YAMAZAKI

CARNEGIE MELLON UNIVERSITY
THE ROBOTICS INSTITUTE
SMITH HALL
PITTSBURGH PA 15213-3896

SEPTEMBER 1996

FINAL REPORT FOR PERIOD 06/30/93 - 09/30/96

Approved for public release; distribution unlimited

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7623

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS REPORT IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

Raymond L Withman
RAYMOND L. WITHMAN
Project Engineer

JAMES C. RACHAL, CHIEF
ATR Development Branch

JERRY L. COVERT, CHIEF
COMBAT INFORMATION TECHNOLOGY DIVISION

IF YOUR ADDRESS HAS CHANGED, IF YOU WISH TO BE REMOVED FROM OUR MAILING LIST, OR IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR ORGANIZATION PLEASE NOTIFY WL/AACR WRIGHT-PATTERSON AFB OH 45433 7334 TO HELP MAINTAIN A CURRENT MAILING LIST.

Do not return copies of this report unless contractual obligations or notice on a specific document requires its return.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Sep 96	Final 30 Jun 93 - 30 Sep 96

4. TITLE AND SUBTITLE

Adaptive Model Based ATR System

5. FUNDING NUMBERS

F33616-93-1-1282
PE 63226
PR A368
TA R1
WU A1

6. AUTHOR(S)

Katsushi Ikeuchi, Dean Pomeleau, Robert Collins, Mark Wheeler, Takeshi Shakunaga, Taku Yamazaki, Khotaro Ohba

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Robotics Institute
Smith Hall
Carnegie Mellon University
Pittsburgh PA 15213-3896

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Project Agency
3701 N. Fairfax Dr
Arlington VA 22203-1714

Avionics Directorate
Wright Laboratory
Air Force Materiel Command
Wright-Patterson AFB OH 45433-7623
POC: R. Withman, SL/AACR, 937-1105 x 3428

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

WL-TR-97-1016

11. SUPPLEMENTARY NOTES

N/A

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for public release, distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This project develops a SAR-ATR system that uses invariant histograms and deformable templates. An invariant histogram is a histogram of geometric invariants given by primitive feature sets. Deformable template matching examines the existence of an object by superimposing templates over potential energy fields derived from the image so that it generates the minimum deformation (deformation energy) and the best alignment of the template with features (potential energy).

This system has two modes: off-line and on-line. In off-line mode, it generates a library for indexing and deformable templates for verification. In on-line mode, by calculating an invariant histogram from an input image, it performs the deformable templates, it determines the most likely pose and class of the target. We have demonstrated the effectiveness of these two techniques for robust SAR recognition using occluded and camouflaged target images.

By anlalyzinanalyzing the evaluation results, we have proposed three extensions of the system: dense sampling for robust recognition, partial view windows for robust indexing under occlusion, and photometric invariants for robust verification under camouflage. Some of these ideas have been evaluated; they are quite promising.

14. SUBJECT TERMS		15. NUMBER OF PAGES
Automatic Target Recognition	Eigenspace	49
Synthetic Aperture Radar		16. PRICE CODE
Geometric Invariant Histograms		

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

Table of Contents

List of Figures

List of Tables

1. Introduction

Recognizing a target in synthetic-aperture radar (SAR) image [1]-[2] is a difficult problem for conventional computer vision systems. First, all SAR features are non-attached; they are not tightly related with surface markers nor explicit object geometry such as edges. Rather, they are floating over a target surface. Thus, they suddenly appear, disappear, and abruptly change their shapes due to tiny movements by an observer. Secondly, in SAR image recognition, objects are of10 in10tionally hidden from an observer. For example, enemy tanks are often10 hidden under trees. A whole tank may be camouflaged completely with a camouflage net.

Historically, target recognition in SAR images is attacked using three different approaches: statistical pattern recognition [3], model-based [5]-[6], and artificial neural network [4]. Among these three approaches, model-based approach [5]-[17] is the most promising, because of its potential for the robust recognition. In essence, a model-based system analyzes each image in detail and identifies each part of a signature contribution toward recognition, while pattern recognition and artificial neural network based recognition system handle a target signature as a whole. This capability of part analysis in the model-based vision approach provides the potential for the robustness with respect to partial occlusion of target, and cluttered background.

Promising features from SAR appearances are isolated peaks. Among several proposed model-based techniques, pose clustering is suitable for determining the object pose (and identifying the object) from such sparse features. Representative pose clustering techniques include: Hough transform and geometric hashing. Ballard [18] generalized the Hough transform to detect arbitrary patterns. Recently, several alternative techniques have been proposed by Lamdan and Wolfson [19], Dhome et al. [20], Stockman [21]). Grimson [8] reported that searching pose space with Hough transforms is very effective for 2D object recognition. Wolfson and Lamdan [22] also reported an effective recognition system using geometric hashing. These pose clustering techniques are highly optimized so that each relation among a pair of features can reduce the possible interpretations as much as possible. However, pose clustering becomes unstable when relative relationships among features vary, as is the case of non-attached SAR features.

Recently, several model-based recognition systems based on geometric invariants have been proposed [27], [28]. Geometric invariants, such as the cross-ratio, provide very efficient clues for identifying 3D objects. In this paper, we will denote geometric invariants, such as the cross-ratio of four points, as *strong invariants*. The utilization of strong invariants to object recognition requires the correspondence problem to be solved prior to applying such invariants. This may be an easy problem when an object contains a few feature points; however, combinatorial explosion occurs when handling cluttered images typical of SAR images.

This paper introduces an invariant histogram based on *weak invariants*, defined by a pair of features, to avoid the difficult correspondence problem. Though each invariant is weak, we demonstrate that a histogram of observable weak invariant values can be used to identify the object uniquely. Moreover, by utilizing all of the weak invariants in an image in a highly redundant manner, our system can achieve robust recognition under severe occlusion with unstable SAR features.

We have built a recognition system that consists of indexing and verification. The indexing module quickly reduces the number of candidates using the invariant histogram technique. To select the correct candidate, the verification module employs deformable template matching to test for the existence of each feature. Here, each SAR feature is nonattached and can vary its position. Deformations are necessary for fine-tuning each feature positions locally.

The system is designed under the vision algorithm compilation paradigm [16]. The system has two modes: off-line and on-line. In off-line mode, model invariant histograms and deformable templates are generated from target models using a sensor simulator. In on-line mode, an image invariant and potential fields are computed from an input image and our indexing and verification algorithms are applied.

Section 2 will introduce the concept of our invariant histogram technique, and Section 3 describes how to use the technique for designing the indexing module. Our deformable template matching method will be discussed in Section 4. Section 5 presents our implementation and Section 6 shows the performance evaluation. In Section 8, we discuss our observation of the evalua-

tion. Sections 8, 9 and 10 show the extension of the system based on the observation. Section 11 summarizes this report.

2. Invariant Histogram

In order to achieve robust recognition under severe occlusion or camouflage with unstable SAR feature, our system introduces an invariant histogram based on weak invariants, defined by a pair of features. Though each invariant is weak, we demonstrate that a histogram of observable weak invariant values can be used to identify the object uniquely. We utilize all of the weak invariants in an image in a highly redundant manner.

We apply this invariant histogram to the indexing module. The indexing quickly reduces the number of the possible candidates before expensive candidate verification. It looks up an indexing library. The indexing library consists of the invariant histograms, distributions of invariant values of an object. By comparing the observed invariant histogram with model histograms in the indexing library, the module decides which candidates are the most likely ones. This process requires measurement similarity between an input and a model invariant histogram. The section will also discuss the similarity measure defined on the invariant histogram.

2.1. Concept of Invariant Histogram

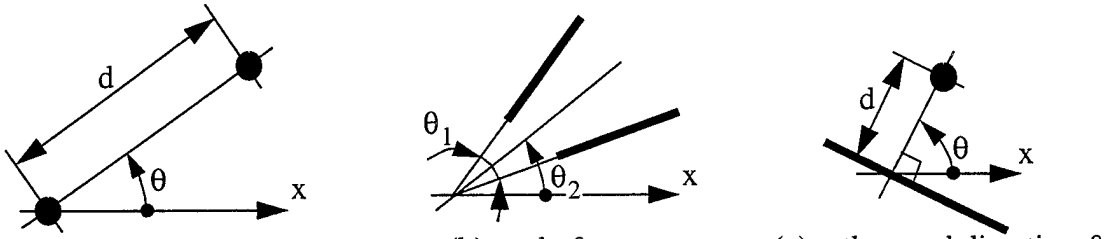
An invariant histogram, a histogram of invariant values, stores many invariant values from a target model, though only a few invariants are actually necessary for recognition. Thus, this invariant histogram is a redundant space robust against variation in computed invariant value typical of SAR data.

This paper employs weak invariants, such as distance of two points or the slope of the bisecting line of two lines. Strong invariants, such as a cross ratio of four points on a line, are convenient for object recognition, yet difficult to reliably extract from real data. We, instead, rely on weak invariants defined using only pairs of primitive features in this system. This is because known strong invariants require too many primitive features to be extracted reliably; for example, a cross ratio needs four points to be identified in an image. Detecting a group of features is rarely possible in the SAR domain, where most features are quite unstable.

In our SAR recognition system, two kinds of primitive features are extracted from an image:

points and line segments. All feature points are detected by applying an interest operator. All line segments are detected by a line detector based on the Canny edge detecting technique.

When a target rotates in 3D space, the appearance of the target in SAR images drastically changes. On the other hand, even though a target translates along the ground plane, the appearance is not significantly altered. Thus, we decide to use translation invariants to construct invariant histograms. Figure 1 shows six translation invariants which are used for constructing our invariant histograms.[1]



(a) displacement & direction

(b) angle & slope of the bisecting line

(c) orthogonal direction & orthogonal distance

Figure 1 Six Invariants Used in Our Implementation

2.1.1. Point-Point (PP) Histogram (Figure 1(a))

Distance and direction between a pair of points is a translation invariant. This invariant is calculated for a pair of primitive feature points, and histograms are made in the 2D displacement space[2].

2.1.2. Line-Line (LL) Histogram (Figure 1(b))

An angle between two line segments and a slope of their bisecting line are invariant to translation. We use these two invariants for characterizing a line pair[3].

2.1.3. Point-Line (PL) Histogram (Figure 1(c))

An orthogonal distance from a point to a line is invariant to translation and rotation. An orthogo-

---

1. In order to increase the robustness of the system against camouflage and surrounding noise, we do not use properties of peaks or edges (such as brightness values of a peak or area size of a peak); we only use spatial relations among peaks and edges.
2. The 2D space is composed of a 2D array, of which each cell has widths of 2 or 4 pixels along x- and y-axes in our implementation.
3. We do not use all the line segment pairs to make an LL histogram. A nearby subset is first generated from all the line segment pairs, so that the minimum distance between two line segments is less than a threshold. The coupled invariant is then calculated over the line pair subset. The resulting invariant histogram is in 2D space whose dimensions correspond to the angle and the slope of bisecting line. Both the angle and the slope are quantized to 10 degree intervals in our implementation.

nal direction from a point to a line segment is also invariant to translation. We use this orthogonal distance and its direction for characterizing a point-line pair.[1]

2.2. Implementation of Invariant Histogram

A two-dimensional invariant histogram is implemented as a collection of tessellated bins. Each pair of geometric features provides a pair of invariant values. Those values are then histogrammed to the corresponding bins in the corresponding 2D invariant histogram. At the same time, the bin maintains pointers to keep track of the original primitive pairs that vote for it. Since several pairs may lie in the same bin, each bin may contain multiple pointers. These pointers will be utilized later for establishing initial correspondences for verification between image and model features.

Figure 2 shows a procedure for generating an invariant histogram from an image. First, primitive features, point features in this example, are extracted from an image. From point pairs, invariant values are obtained: distance and direction. When making point pairs, the system considers only local feature pairs, those within a certain threshold, indicated as a circle in the figure. The horizontal axes in the indicated values of distance and direction, and the vertical axis denotes the number of votes for each bin.

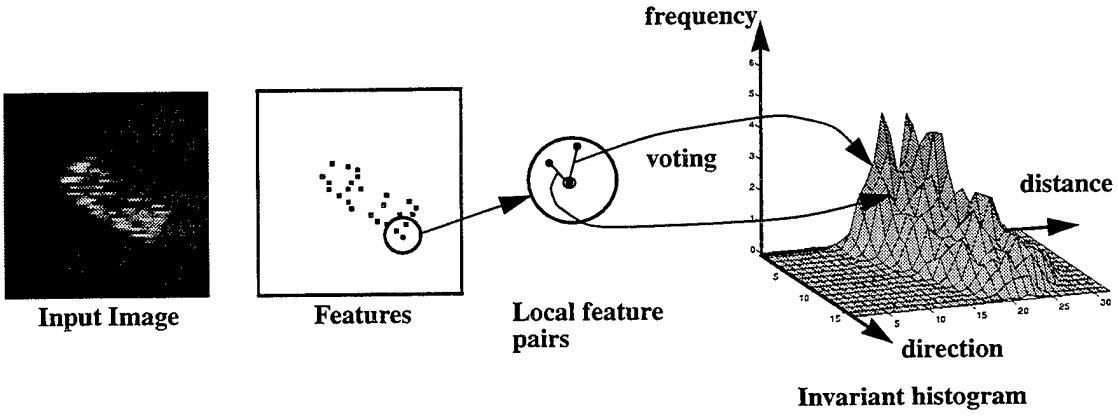


Figure 2 Invariant Histogram Generation

1. To construct a PL histogram, a subset of point-line pairs is first made up from all the pairs, so that the foot of the perpendicular is included in the line segment. Then the coupled invariant is calculated over the point-line pair subset. PL histograms are made in the 2D space of which two axes correspond to the distance and the direction. The distance is quantized to intervals of 2 or 4 pixels, and the direction is quantized to 10 degree intervals in our implementation.

To minimize problems due to quantization, we smooth the histogram over a small neighborhood. We use weighted voting for the four nearest neighbor bins of the real point in the invariant space. The weights are calculated so as to be inversely proportional to the distance from the centers of the bins to the real point. The sum of four weights is normalized for each occurrence. At the same time, the pointers to the feature pair also copied to the four bins

3. Indexing by Library Lookup

We have described the details of the invariant histogram representation. Now, we will describe how we utilize these histograms to screen or eliminate candidate hypotheses in our recognition algorithm in the indexing module. We can build invariant maps for each representative view, and use these maps to compute distance measures between an invariant map computed from the image and each candidate. The candidates can be ranked by this distance measure and then pruned accordingly. In this process, we refer to the collection of invariant histograms as an indexing library which represents how a target object appears and, thus, invariant values change depending on pose parameters. This indexing library is constructed from model appearances at off-line.

3.1. Structure of a Library

Pose parameters can be decomposed into two categories: invariant and variant pose parameters with respect to a defined weak invariant. Invariant pose parameters do not alter the invariant value; variant pose parameters do. In our current implementation, translation of a target does not change our invariant values, while rotation does change their values. Thus, translation and rotation parameters are invariant and variant pose parameters, respectively, with respect to our weak invariants.

We will construct an indexing library, a collection of invariant histograms, to cover all of the variant parameter space. Rotation parameter spaces are evenly sampled, and invariant histograms are constructed at these sampled rotation values. Here, each sampled rotation value is denoted as a representative view.

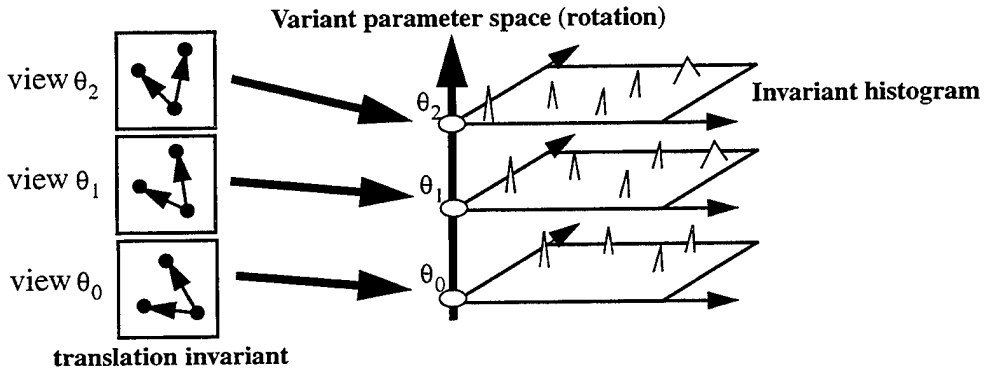


Figure 3 Invariant and Variant Space

For each interval, we compute the average and variance histograms over the interval. Some features appear and disappear abruptly, while other features may be observable from a wide range of viewing directions. Histogram values voted by such abrupt pairs are unstable and unreliable for indexing, while others are stable and reliable. A variance histogram conveys this reliability measure. We take a large number of neighboring images around each representative view and generate histograms for each. Then the average and variance histograms are computed from these surrounding histograms; this cumulative invariant histogram is used as a histogram of the particular indexing library entry.

3.2. Similarity Measure for Invariant Histograms

This section will describe a similarity measure for comparison between image and model histograms in an indexing library. A model histogram comprises average and variance histograms. Basically, average values in a model histogram are compared with those from input image and, then, the difference will be weighted using variance values.

One simple similarity measure is L1 norm as follows:

$$L = \sum_{i,j} |x_{i,j} - m_{i,j}| , \tag{1}$$

where $x_{i,j}$ is an image value in bin (i, j), while $m_{i,j}$ is a model value. This difference will be calculated over the all of the histogram.

Some values in bins are less reliable than other values depending on the reliability of values at bins; we will adjust the difference using a variance value$\sigma_{i,j}$ at each bin:

$$L_1 = \sum_{i,j} \frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j}}. \tag{2}$$

The L_1 norm provides severe penalty, when some features are occluded and values disappear from a histogram. In order to avoid such effect from occlusion, we further modify the measure by introducing the saturation factor, $k\sigma_{i,j}$. Namely, if the difference between the observed and model values is larger than this saturation factor, the penalty imposed is the saturation factor

instead of the real distance:

$$L_{1,sat} = \sum_{i,j} min\left(\frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j}}, k\sigma_{i,j}\right) \quad . \tag{3}$$

When a histogram does not have a value in one bin, the variance, $\sigma_{i,j}$ is zero; we cannot evaluate the value. Thus, we will add a constant variance σ_o:

$$L_{1,sat} = \sum_{i,j} min\left(\frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j} + \sigma_o}, k\sigma_{i,j}\right) \quad . \tag{4}$$

Here, two constants, σ_o and k are obtained empirically.

3.3. Implementation of Indexing Algorithm

Using this similarity measure, we will design the following four step indexing algorithm. Since there are three different histograms, PP, LL, and PL, their relative weights are adjusted using normalization factors given by the maximum distance values over the bins of the histogram.

Step 1: Absolute distance

For each of the PP, LL and PL histograms, the absolute distance is calculated between an image and each model histograms. The absolute distance is given by the L1 norm with saturation given by the equation (4).

Step 2: Relative distance

For obtaining relative distance, the maximum distance between corresponding bins of the image and model histograms is determined for each of the PP, LL and PL histograms using

$$a_{max} = max_{i,j}\left\{min\left(\frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j} + \sigma_o}, k\sigma_{i,j}\right)\right\}. \tag{5}$$

For example, we will use $\frac{L^{PP}_{1,sat}}{a^{PP}_{max}}$ as the relative distance between two PP histograms.

Step 3: Total distance

The total distance is, thus, defined by:

$$L^{total}{}_{1,sat} = \frac{L^{PP}{}_{1,sat}}{a^{PP}{}_{max}} + \frac{L^{LL}{}_{1,sat}}{a^{LL}{}_{max}} + \frac{L^{PL}{}_{1,sat}}{a^{PL}{}_{max}} \quad (6)$$

Step 4: Candidate screening by total distance

The most likely representative view is determined by the total distance between the input and model histograms. Since the indexing is not to determine one particular view but to select multiple possible candidate views, we select those with distance less than a certain threshold value.

4. Pose Clustering Using Invariant Histogram

After obtaining variant pose parameters (rotation parameters), we will determine the invariant pose parameters (translation parameters) using the correspondences between image and library features through an invariant histogram. First, we will explain how to establish these correspondences using invariant histograms. Then, we will describe our method for obtaining invariant pose parameters by pose clustering.

4.1. Sampling Correspondences

Each bin of an invariant histogram has pointers to the primitive features that vote for this bin. By retrieving the pointers of corresponding bins of the input and model histograms, we can establish correspondences between image and model primitive features.

Let us consider a case of the LL histogram as an example, as shown in Figure 4. By tracking pointers, three line pairs are retrieved in an image as candidates of one line pair of a model. The two translation values are computed for each candidate. These translation values are computed by comparing between the middle points of lines. If these two translation values are near to each other, the correspondence can be established. Then, their average is combined, yielding the average translation parameters, and the transformation is given to the pose clustering algorithm. Otherwise, the line pair correspondence is removed as a false correspondence.

Assume that n image feature pairs are referenced from a bin, and m model feature pairs are referenced from the corresponding bin. We will consider mn possible correspondences in this bin. It is noted that no more than $min(m, n)$ ones are correct among the mn correspondences, if the one-to-one mapping holds between the input and the model features. These correct correspondences generate the correct pose candidates in the invariant parameter space, while the other correspondences generate false pose candidates. When the number of possible correspondences is too large, we do not have to consider all of them. The possibility that the values of the model invariants randomly occur in the input image is very low compared to actual occurrences due to the model presence in the image. Thus, random sampling can achieve a large reduction of the computation time

with little or no loss in the detection rate.

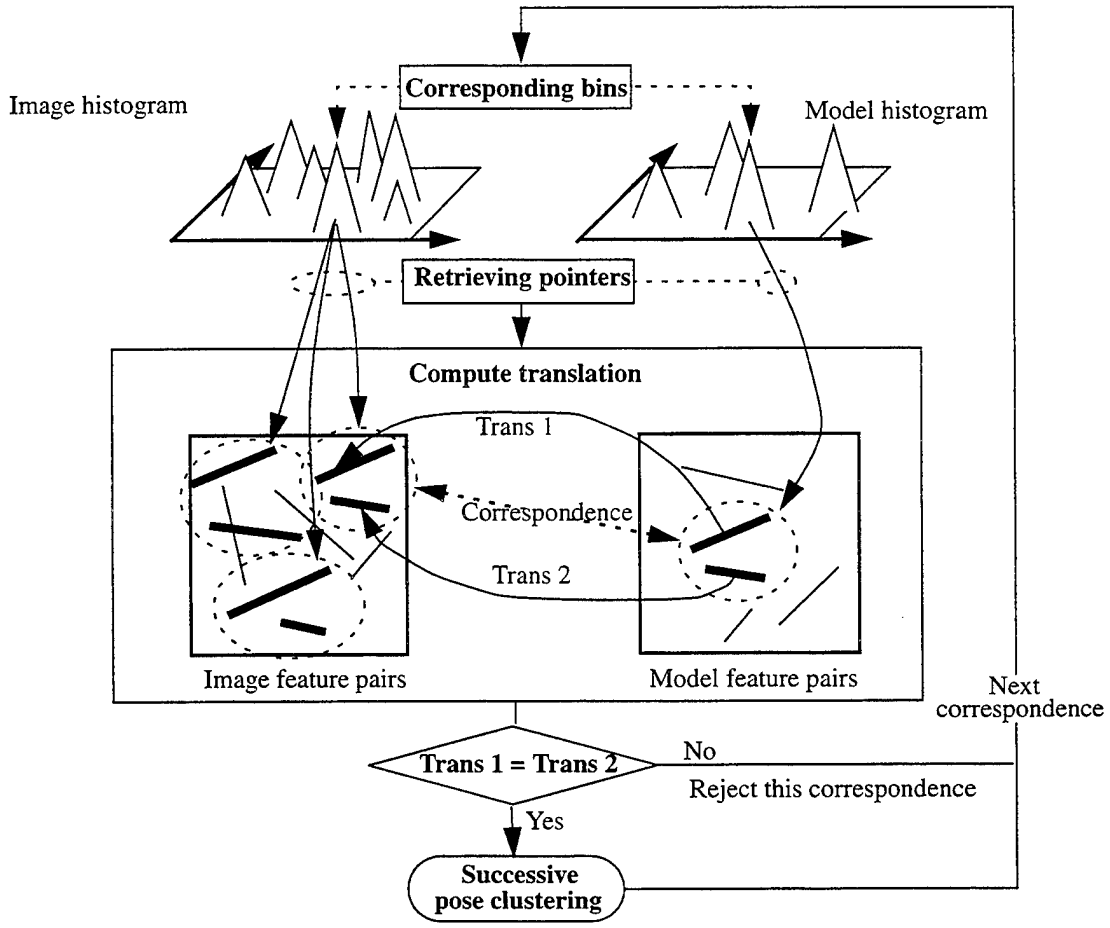


Figure 4 Pose Clustering Through an Invariant

4.2. Successive Pose Clustering

Several kinds of techniques are used for pose clustering ([18]-[25]). The most popular technique is the generalized Hough transform in which voting is applied to the quantized pose space. Although these voting methods are easy to implement, we have to determine the size of each cell in the quantized space before execution. The quantization is closely related to uncertainty which is difficult to estimate.

We implement a successive clustering algorithm to avoid the difficulty of quantization. This method successively generates clusters without voting. By tracking pointers in bins, some feature

correspondences are established and a candidate pose can be obtained. This pose candidate is examined whether or not it is within a certain distance to one of the existing clusters. If it is within distance from several clusters, the largest cluster will be selected to include the new candidate. The average pose and the size of the cluster will be updated at each iteration.

The clustering process terminates either when the size of the largest cluster is large enough or when the total number of generated pose candidates reaches a threshold. In both cases, the largest cluster is selected as giving the final results.

Pose clustering provides a rough estimate of translation parameters (invariant parameters), while a library lookup gives a rough estimate of rotation parameters (variant parameters). Using these estimates of the pose parameters, the pose is refined using deformable template matching before final verification.

5. Verification through Deformable Template Matching

Deformable template matching examines the exis10ce of a target by superimposing deformable templates over potential energy fields given by features by obtaining the template configuration of the minimum deformation (deformation energy) and the best alignment of the template with features (potential energy). A SAR image of10 contains a large number of nonattached features. We employ multistep deformable template matching to avoid local minima given by these erroneous features. We start to examine the existence of a target object in the position given by the translation parameter from the previous pose clustering. Then, our recognition system uses multiple-level template and potential fields progressively from the coarse to the fine level.

5.1. Template Generation

Templates are generated from model appearances generated by XPATCH simulator in off-line mode. The first two levels of template matching, the coarse and medium level, share the same nondeformable template, while the fine level matching employs a deformable template. These templates are generated at each representative view over the evenly sampled rotation space as used to sample the rotation parameter for the indexing dictionaries.

5.1.1. Coarse/Medium Level Template

Nondeformable templates are generated from binarized model images. First, we threshold noise-free model images and binarize the output, $I(x, y)$. Then, we repeat this process 11 times around a representative view and superimpose them taking the union. The resulting superimposed binarized point distribution, $T^{\theta_o}(x, y)$ is the template at the central viewing direction, θ_o (representative view).

$$T^{\theta_0}(x, y) = I^{\theta o - 5\Delta}(x, y) \cup \ldots I^{\theta o}(x, y) \ldots \cup I^{\theta o + 5\Delta}(x, y) \quad . \tag{7}$$

Combining the templates is necessary to absorb all unstable SAR nonattached features in one template.

For the coarse and medium templates, only translation (x_t, y_t) is allowed; there is no relative

movement of each point. The total energy is provided as the sum of potential energy values at each point position and the translation energy of the entire template:

$$E_{total} = E_{potential} + E_{trans}, \tag{8}$$

where

$$E_{potential} = \iint P(x, y) T^{\theta_o}(x + x_t, y + y_t) dx dy , \tag{9}$$

$$E_{trans} = k_t (x_t^2 + y_t^2)^{\frac{1}{2}} . \tag{10}$$

$P(x, y)$ is the potential field function given from an input image and k_t is a spring constant. Both the coarse and medium level matching uses the same value for this spring constant. This translation term is introduced to give the priority to positions close to the one given by feature correspondences.

5.1.2. Fine Level Template

The fine level matching employs deformable templates: each feature point moves freely relative to other points. At this level, there is no translation of the entire template. The deformations are necessary to account for typical perturbations in position of SAR features.

These deformable templates are generated using a point feature extractor. First, a noise-free model image of a target object, $I(x, y)$, is convolved with a Gaussian filter. From this smoothed image, $I_{gauss}(x, y)$, we extract isolated brightness peaks, $I_{point}(x, y)$ using our regular point feature extractor. This is a binary distribution; $I_{point} = 1$ at a peak and 0 otherwise. In the same way as the nondeformable templates, 11 such point distributions around a representative view, θ_o, are superimposed.

$$D^{\theta_o}(x, y) = \sum_{i=1}^{p} \delta(x - x_i, y - y_i), \tag{11}$$

where p is the total number of points over 11 point distributions.

The total energy of this template is:

$$E_{total} = E_{potential} + E_{deform} \tag{12}$$

where

$$E_{potential} = \sum_{i=1}^{p} \iint P(x, y)\delta(x - x_i - \Delta x_i, y - y_i - \Delta y_i)dxdy \,, \tag{13}$$

$$E_{deform} = \sum_{i=1}^{p} k_d \left\{ (\Delta x_i)^2 + (\Delta y_i)^2 \right\}^{\frac{1}{2}} \,. \tag{14}$$

5.1.3. Difference Template

Confusion often occurs between one pose and its counter pose (rotated 180 degrees from the original pose). In order to avoid confusion between a pair of poses, our system employs a difference template as the fourth step of matching. This template suppresses common parts and emphasizes conflicting parts between the pair. This difference template is used only when it is necessary to disambiguate a pair of candidates of close score.

In order to make a difference template prior to execution, first, the best possible alignment of a pair of coarse-level templates is obtained. Let us denote a pair of poses as A and B . A coarse-level potential field, P_B is generated from a template T_B. Then, the template T_A will be applied to the potential field, P_B to obtain necessary translation $(\Delta x_A, \Delta y_A)$ for optimally superimposing template, T_A over the template, T_B. See Figure 5.

Using this translation value, we superimpose a pair of fine-level templates to extract the common points in fine-level template, D_A. Then, the common points are suppressed in the template and the difference template for pose A is:

$$S_A(x + \Delta x_A, y + \Delta y_A) = D_A(x + \Delta x_A, y + \Delta y_A) - D_B(x, y) \otimes D_A(x + \Delta x_A, y + \Delta y_A) \,. \tag{15}$$

By exchanging A and B, we will also obtain the difference template of pose B.

5.2. Generating the Potential Fields

Three potential fields, coarse, medium, and fine, are generated at on-line mode from an input image. For all these three potential fields, a threshold operation is applied to the original intensity distribution of the input image and then, a Gaussian filter is applied to the threshold image I_{th}. Figure 6 shows the overview of this module.

$$I_{gauss}(x, y) = \iint I_{th}(x-u, y-v) e^{-\frac{1}{2}\frac{u^2+v^2}{\sigma^2}} dudv \tag{16}$$

5.2.1. Coarse Level Potential Field

To generate the coarse level potential fields, we first apply the median filter; the median value is obtained among nine neighboring pixels, and then, is assigned to the central pixel I_{median}. This process removes isolated bright pixels. Finally, we apply an exponential function with the width k_{coarse}, to this output:

$$I_{coarse}(x, y) = \iint I_{median}(x-u, y-v) e^{-k_{coarse}(u^2+v^2)^{\frac{1}{2}}} dudv \quad . \tag{17}$$

We prefer to the exponential function than the Gaussian function, for emphasizing the central value and suppressing peripheral areas.

5.2.2. Medium Level Potential Field

For this level, we directly apply the exponential function to the output of the Gaussian filter. k_{medium} is selected to make this exponential function narrower:

$$I_{medium}(x, y) = \iint I_{gauss}(x-u, y-v) e^{-k_{medium}(u^2+v^2)^{\frac{1}{2}}} dudv \quad . \tag{18}$$

5.2.3. Fine Level Potential field

The third step is deformable template matching. This step allows each point to move to further reduce the potential energy. For this step, we extract isolated brightness peaks, $I_{point}(x, y)$

using our regular point feature extractor. Then, we apply the exponential function to the binary distribution:

$$P_{fine}(x, y) = \iint I_{point}(x-u, y-v)e^{-k_{fine}(u^2+v^2)^{\frac{1}{2}}}dudv \tag{19}$$

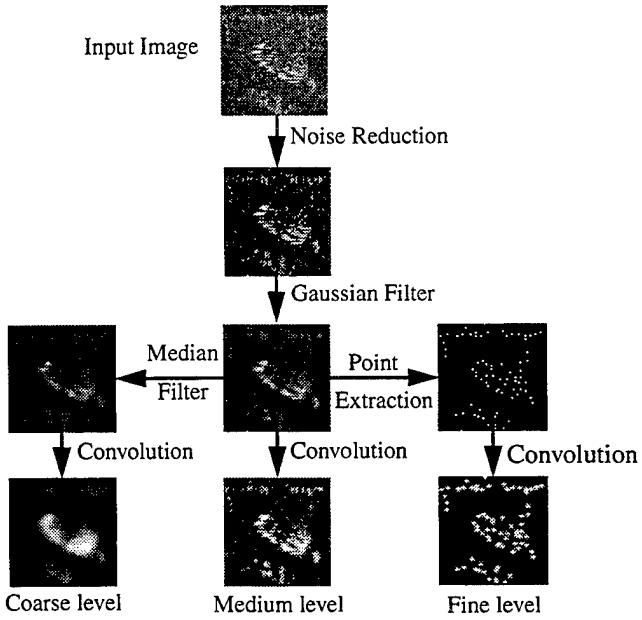


Figure 5 Potential Field Generation

6. Implementation

6.1. Outline of the System

Figure 6 shows the overview of the SAR recognition system. It has two modes: off-line and on-line mode. In off-line mode, the system generates dictionaries for targets using XPATCH SAR simulator and target models. It also generates templates for verification module. In on-line mode, the system generates an invariant histogram and potential fields from the input image. By using the invariant histogram, the indexing module selects possible candidates. Then, the final decision is made by the verification module using the potential fields and templates.

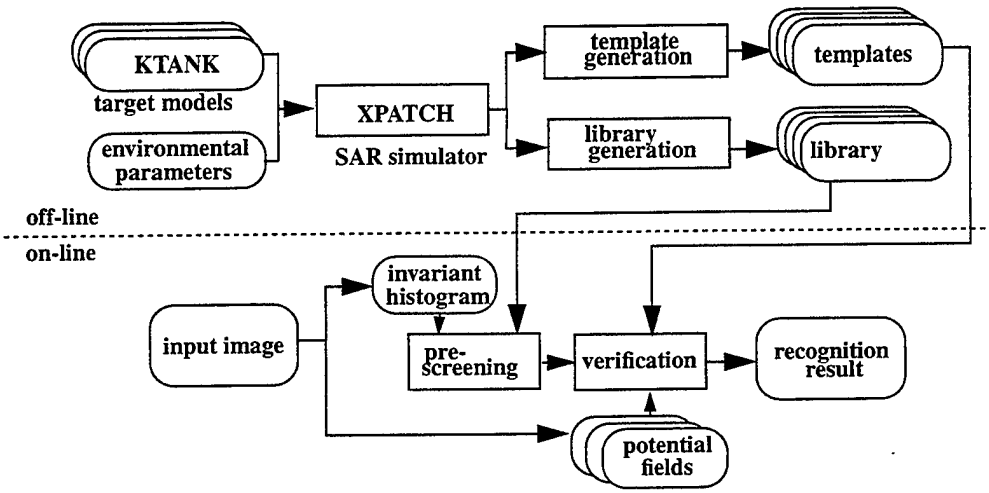
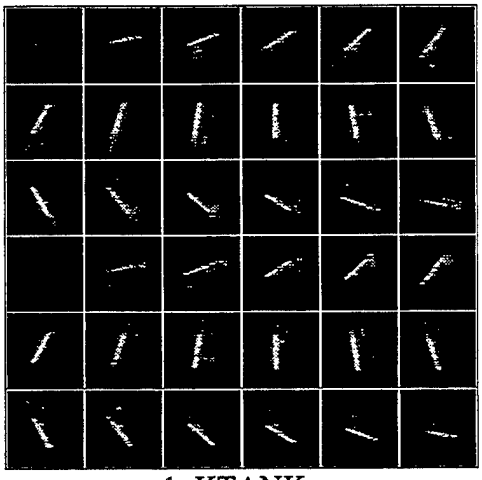


Figure 6 System Overview

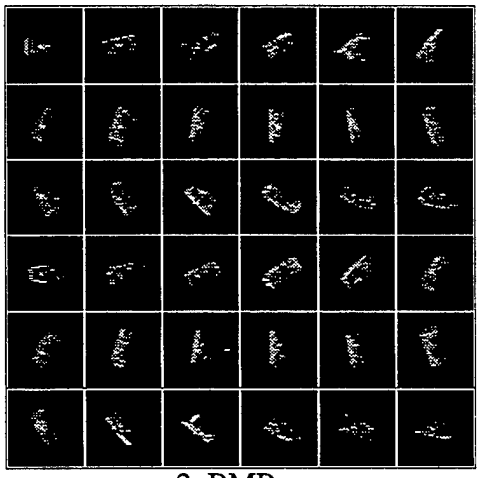
6.2. Off-line Mode

In typical SAR image scenarios, the depression angle and resolution are fixed during image acquisition. In this paper, we use 22.5 degrees as the depression angle and 30cm/pixel as a scale. We employ the XPATCH SAR simulator, developed at WPAFB[1], to generate simulated SAR images for the indexing library generation. Figure 7 shows the five series of 36 images: KTANK, BMP, BTR60, M60, and F15AR generated. An indexing library is constructed for 36 views rotated around the axis perpendicular to the ground plane, sampled every 10 degrees.

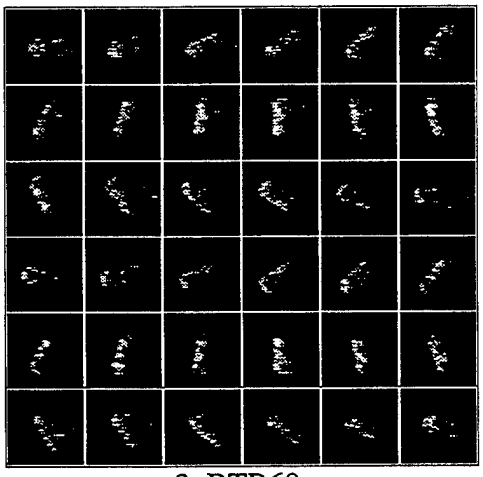
1. WRIGHT-PATTERSON AIR FORCE BASE, OHIO



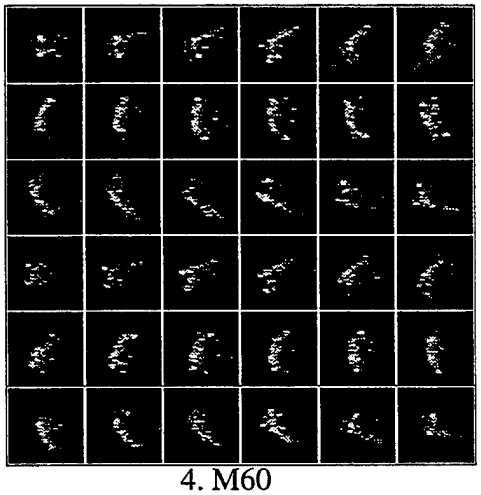
1. KTANK



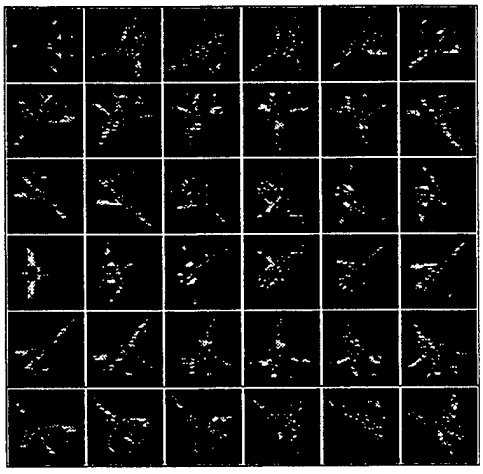
2. BMP



3. BTR60



4. M60



5. F15AR

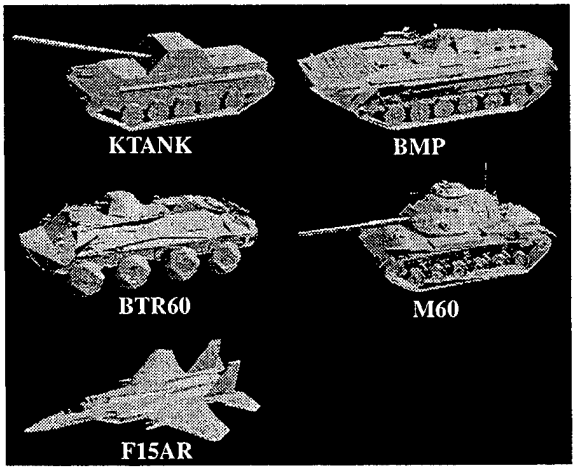


Figure 7 Model SAR Images for Library Generation

In order to obtain an estimate of the variance of each invariant value around a representative view,

19 images around each representative view are generated within 1 degree (0.1 degree intervals). Each representative view of an indexing library consists of three invariant histograms: point-point, line-line, and point-line as shown in Figure 8.

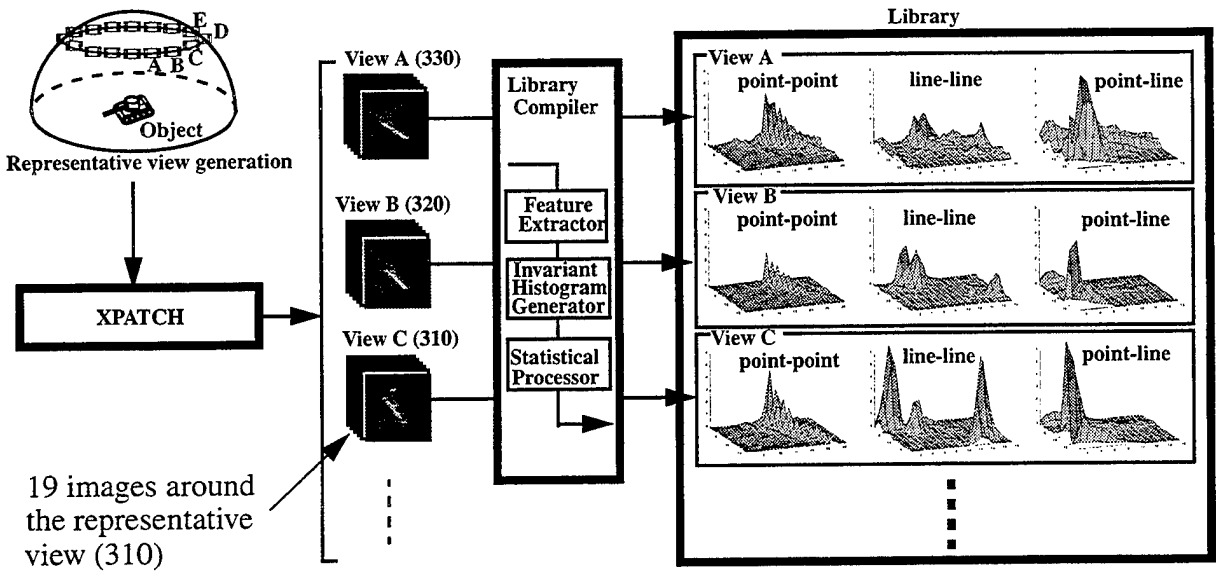


Figure 8 Generating a Library

In off-line mode, coarse, medium and fine templates are also generate at each representative view. For each template, 11 images around a representative view are utilized.

6.3. On-line Model

For the recognition experiments, hybrid SAR images, synthesized SAR simulated signatures of target objects on real SAR background signatures (Lincoln Stockbridge Data), are used. The ratio of signal level between simulated and real signatures are determined using a car parked in a parking lot observed in the Stockbridge Data (M90P5F8HH). Figure 9 shows the KTANK model at a

rotation of 312 degrees superimposed in the lawn area in the Stockbridge Data [31].

Figure 9 Hybrid SAR Image

In on-line mode, the feature detector and invariant generator are used to create the invariant histograms from an input image. The indexing module eliminates impossible candidates by measuring the distance between an input and library images and prunes the number of possible objects for recognition using the similarity measure.

Our similarity measure function, $L_{1,\,saturation} = \sum_{i,j} min\left(\frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j} + \sigma_o}, k\sigma_{i,j}\right)$, has two parameters: k and σ_o. We use $k = 10$ and $\sigma_o = 0.5$.

Figure 10 shows the flow of the indexing. From an input hybrid image (rotation of 312 degrees), the module generates three invariant histograms and then compares them with the library. In this example, the module selects five candidates, including 110, 250 and 310 (the correct pose) of KTANK as the possible candidates for verification.

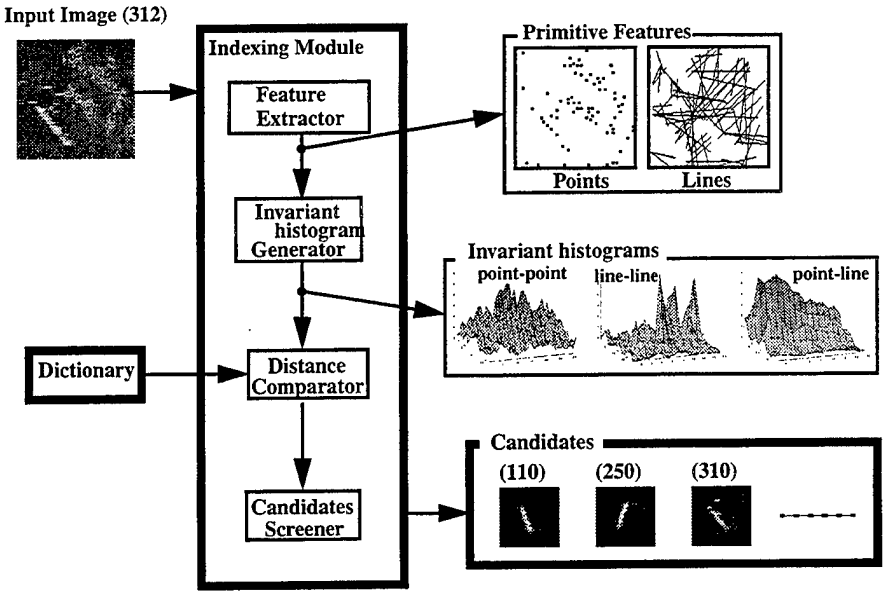


Figure 10 Indexing Module

Figure 11 shows the flow of the verification module. It determines the initial template position using the feature correspondences. Then, the module evaluates each candidate pose through a three-step matching over potential energy fields given by images/features. It determines the template configuration that has the minimum deformation (deformation energy) and the best alignment of the template with features (potential energy). In this example, the module correctly

identifies the template of 310 degrees, as the minium energy template (most likely pose).

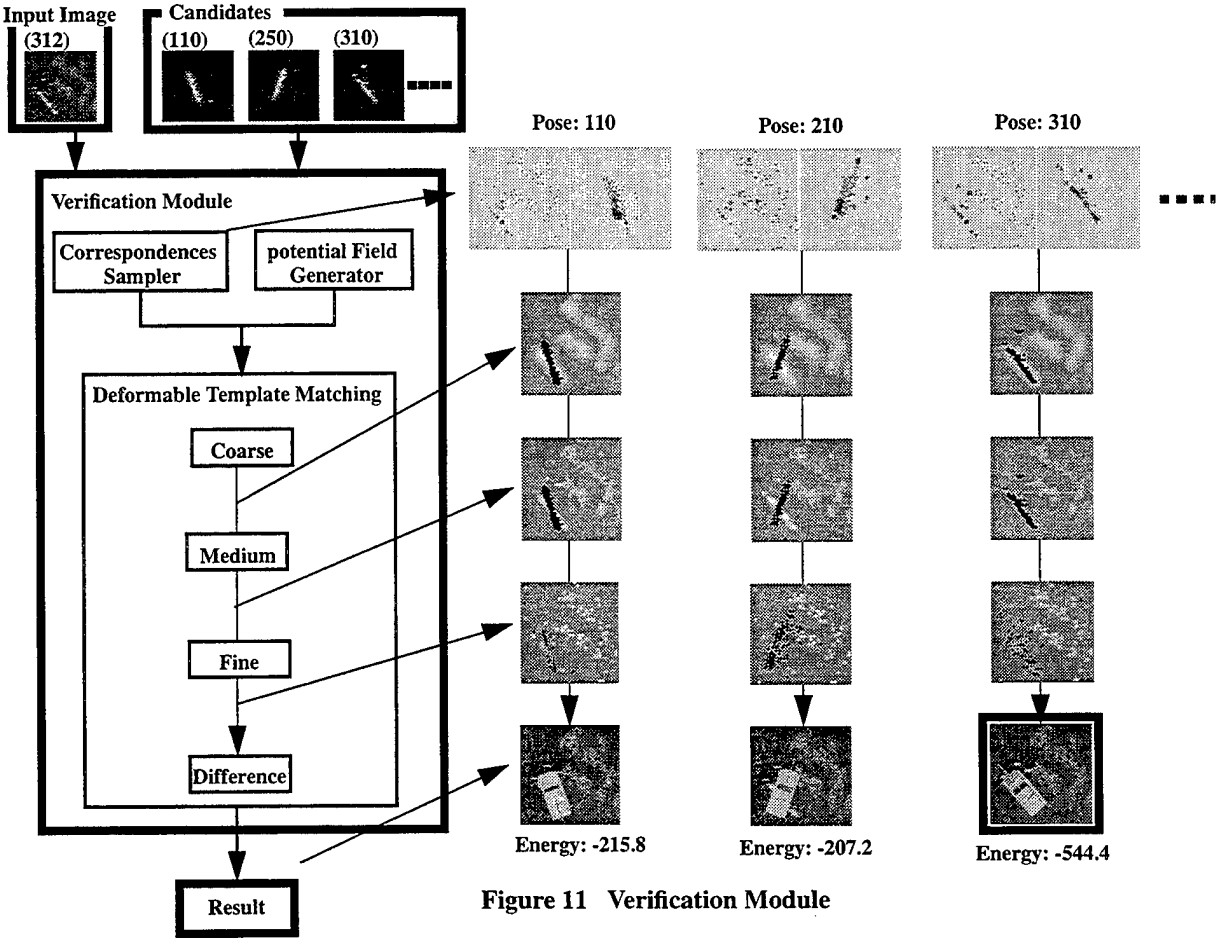


Figure 11 Verification Module

Figure 12 shows the KTANK model at 310 degree aspect angle superimposed on the hybrid SAR images containing the signature from KTANK rotated 312 degree.

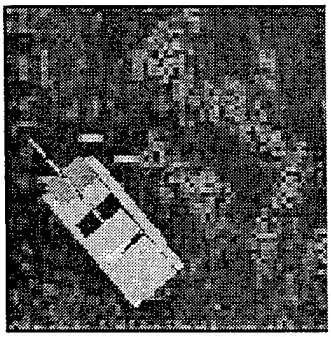


Figure 12 Recognition Result

7. Performance Evaluation

In order to examine the performance of our recognition system, we have generated 180 hybrid SAR images, from viewing directions sampled every 2 degrees around 360 degrees using the following models: KTANK, BMP, BTR60, M60, and F15AR.

We have tested the indexing module, which determines the possible candidates for the verification. For each test, 180 images of the object are given to the system with the single object's library being used by the indexing. At each viewing direction, 5 to 9 candidates are selected on average. Table 1 shows the results of the testing. The first row in the table denotes the results for the object. When the candidate set contains the direction nearest to the input direction, the indexing is considered a success. The second column in the table represents the correct indexing ratio.

Then, the verification module is executed using the templates of the candidate poses selected by the indexing module. Here candidate templates are sampled evenly 10 degrees. When the template nearest to the input direction has the least energy, we consider that the correct recognition (in the fourth column) as well as the correct verification (in the third column) is achieved.

In case that the candidate set given by the indexing does not contain the correct direction, this is the failure of the recognition (in the fourth column), However, for the calculation of the correct verification ratio (in the third column), we discard this case.

Table 1: Recognition Results

Vehicle	Indexing	Verification	System
KTANK	97.8%	90.3%	88.3%
BMP	92.8%	88.0%	81.7%
BTR60	99.4%	97.2%	96.7%
M60	98.3%	98.3%	96.7%
F15AR	80.6%	99.3%	80.0%

7.1. Occlusion

In order to evaluate the effect of occlusion, we generated five series of 180 images (64×64 pixels) with: 0 pixel shift, 8 pixel shift, 16 pixel shift, and 20 pixel shift. Each image is shifted a cer-

tain amount from the center position; if pixels move outside of the window (64×64), we consider them to be occluded. Thus, for example, for a 20 pixel shift, roughly one half of the original pixels are lost in the worst case. We evaluate the effect using five targets, BMP, BTR60, F15AR, KTANK and M60. Tables 2-6 show the results, while Figures 13 and 14 summarize these results in a graphical display.

7.1.1. KTANK

Table 2: KTANK

Occlusion(shift)	Indexing	Verification	System
0 pixel	97.8%	90.3%	88.3%
8 pixel	96.7%	90.8%	87.8%
12 pixel	96.7%	92.0%	88.9%
16pixel	92.8%	86.7%	80.0%
20 pixel	84.4%	74.3%	62.8%
24 pixel	36.7%	62.1%	22.8%

7.1.2. BMP

Table 3: BMP

Occlusion(shift)	Indexing	Verification	System
0 pixel	92.8%	88.0%	81.7%
8 pixel	95.0%	90.6%	86.1%
12 pixel	97.8%	88.6%	86.7%
16pixel	97.8%	90.3%	88.3%
20 pixel	92.8%	86.8%	80.6%
24 pixel	54.4%	83.7%	45.6%

7.1.3. BRT60

Table 4: BTR60

Occlusion(shift)	Indexing	Verification	System
0 pixel	99.4%	97.2%	96.7%
8 pixel	98.9%	97.8%	96.7%
12 pixel	98.3%	95.5%	93.9%
16pixel	98.3%	94.4%	92.8%
20 pixel	85.6%	90.9%	77.8%
24 pixel	35.0%	85.7%	30.6%

7.1.4. M60

Table 5: M60

Occlusion(shift)	indexing	Verification	System
0 pixel	98.3%	98.3%	96.7%
8 pixel	97.8%	99.4%	97.2%
12 pixel	98.9%	98.9%	97.8%
16pixel	96.7%	97.1%	93.9%
20 pixel	73.9%	88.7%	65.6%
24 pixel	21.1%	89.5%	18.9%

7.1.5. F15AR

Table 6: F15AR

Occlusion(shift)	Indexing	Verification	System
0 pixel	80.6%	99.3%	80.0%
8 pixel	82.8%	99.3%	82.2%
12 pixel	80.0%	97.2%	77.8%
16pixel	81.7%	97.3%	79.4%
20 pixel	67.2%	90.1%	60.6%
24 pixel	34.4%	93.5%	32.2%

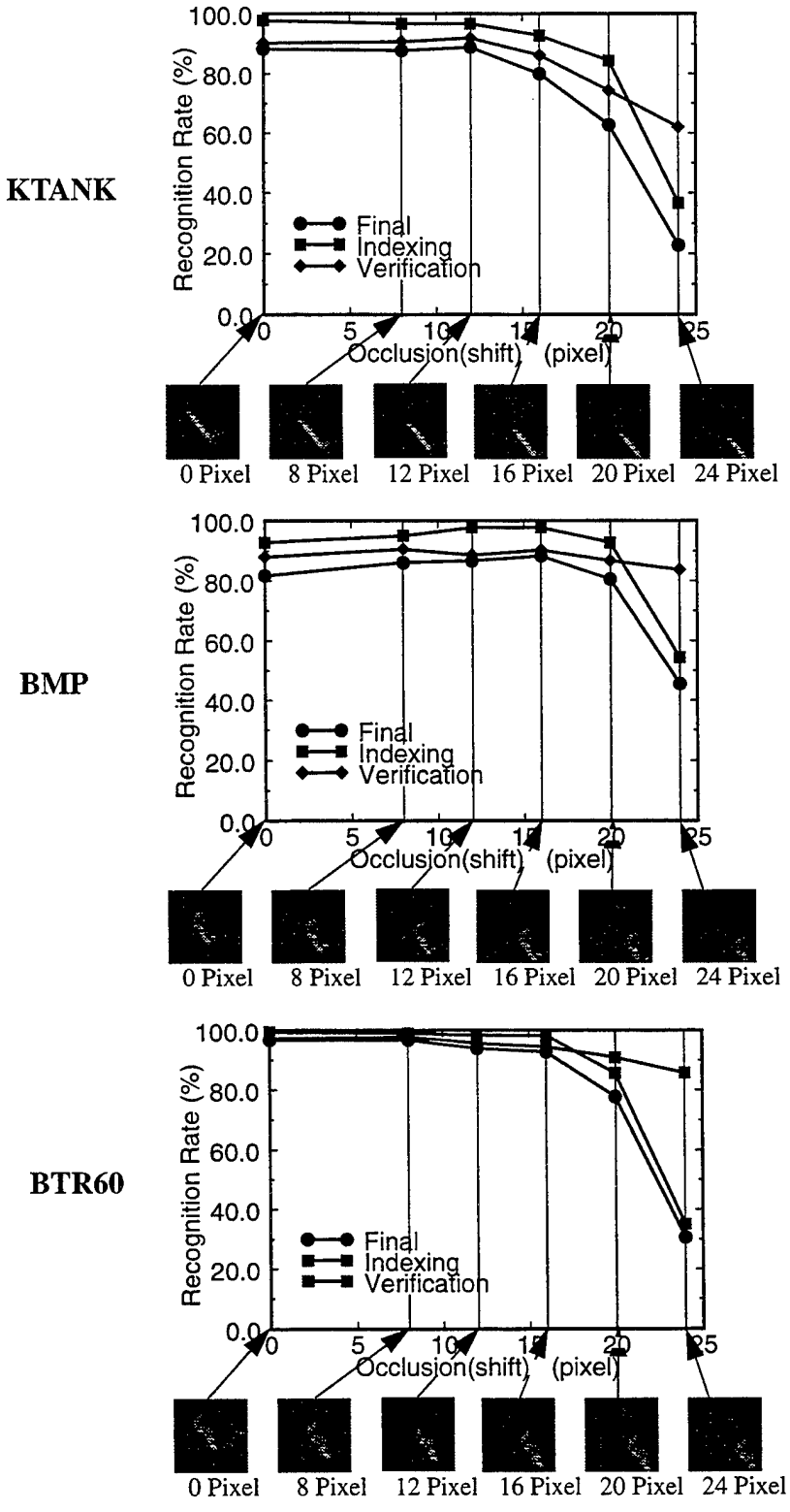
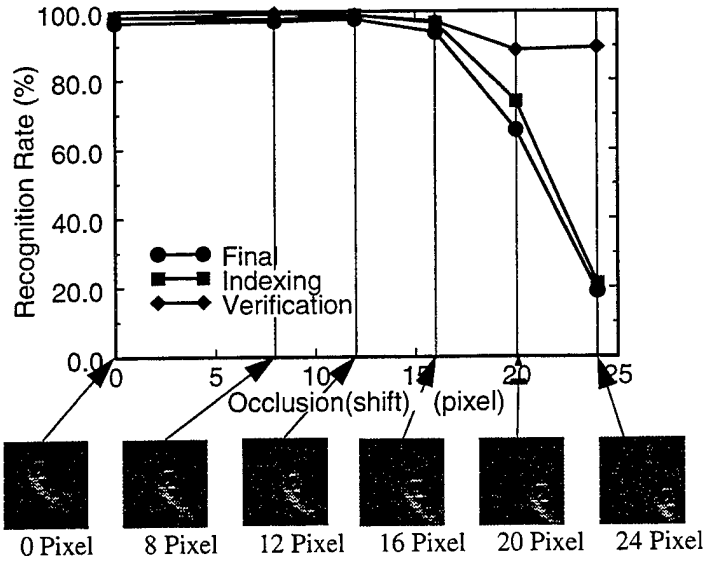


Figure 13 Occlusion(1)

M60



F15AR

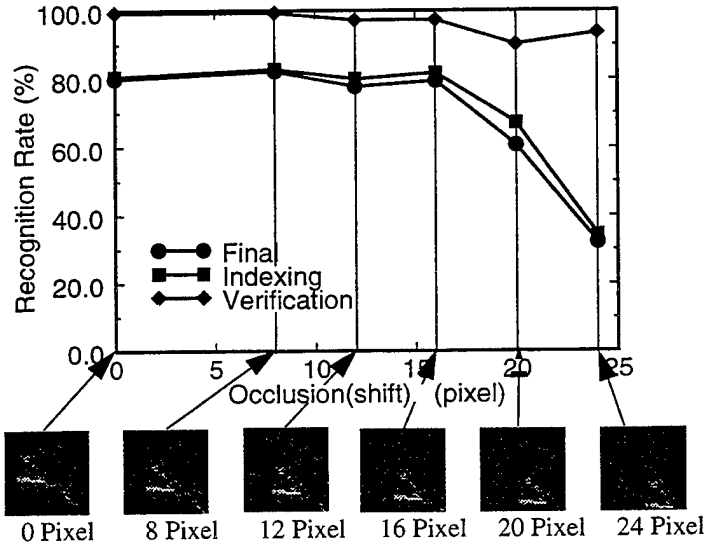


Figure 14 Occlusion(2)

7.2. Camouflage

In order to simulate the effect of camouflage, we reduce the intensity of a target while maintaining the same background intensity. In this experiment in Tables 7-11, the effect of the camouflage is represented by the ratio of the number of features extracted from the reduced and original intensity images. See attached Figures 15 and 16 for the graphical display of these results.

7.2.1. KTANK

Table 7: KTANK

Camouflage(features)	Indexing	Verification	System
100.0%	98.3%	90.4%	88.9%
94.8%	98.9%	86.0%	85.0%
86.8%	98.3%	83.6%	82.2%
77.7%	93.9%	78.1%	73.3%
67.0%	92.2%	74.7%	68.9%
53.4%	91.1%	68.9%	46.6%

7.2.2. BMP

Table 8: BMP

Camouflage(features)	Indexing	Verification	System
100.0%	95.0%	87.7%	83.3%
96.2%	96.1%	86.7%	83.3%
87.6%	96.1%	82.1%	78.9%
79.0%	90.6%	78.5%	71.1%
65.1%	82.2%	79.7%	65.6%
32.4%	57.8%	52.9%	30.6%

7.2.3. BTR60

Table 9: BTR60

Camouflage(features)	Indexing	Verification	System
100.0%	100.0%	96.6%	96.6%
96.7%	100.0%	94.4%	94.4%
92.3%	100.0%	95.0%	95.0%
85.4%	97.8%	90.9%	88.9%
77.7%	97.2%	79.4%	77.2%
56.6%	80.6%	69.0%	55.6%

7.2.4. M60

Table 10: M60

Camouflage(features)	Indexing	Verification	System
100.0%	98.3%	97.2%	95.6%
96.6%	98.3%	97.7%	96.1%
91.9%	98.9%	96.6%	95.6%
86.1%	98.9%	92.7%	91.7%
75.9%	93.9%	91.1%	85.6%
55.5%	70.6%	84.3%	59.4%

7.2.5. F15AR

Table 11: F15AR

Camouflage(features)	Indexing	Verification	System
100.0%	76.7%	98.6%	75.6%
94.0%	86.7%	98.7%	85.6%
86.0%	87.8%	98.7%	86.7%
73.0%	78.3%	97.9%	76.7%
55.5%	51.1%	87.0%	44.4%
4.5%	26.7%	72.9%	19.4%

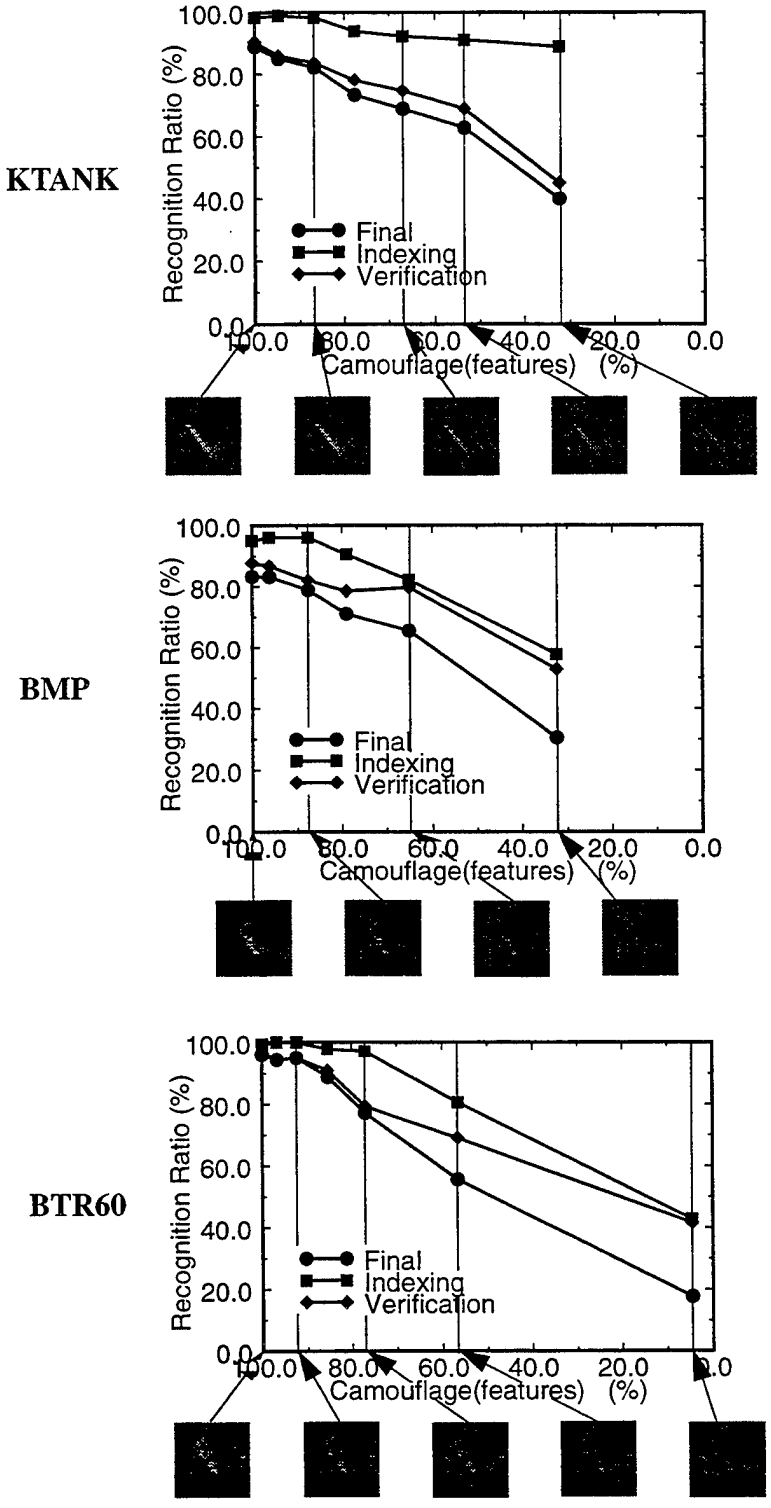


Figure 15 Camouflage(1)

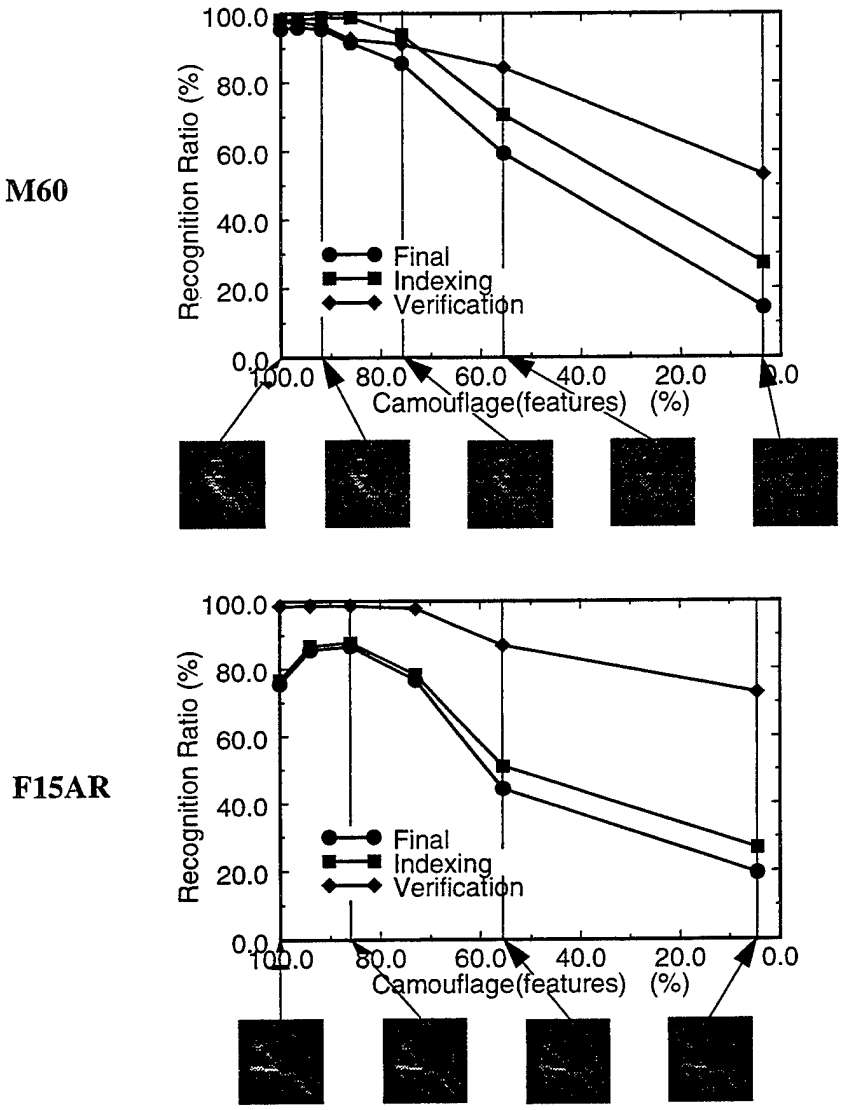


Figure 16 Camouflage(2)

7.3. Multiple Object Databases

We evaluated the effect of confusion among ground vehicles using BMP, BTR60 and KTANK. We generated 90 hybrid SAR images of these vehicles, at 90 viewing directions sampled every 4 degrees over 360 degrees. Here, we use a model library containing BMP, BTR60 and KTANK. In the table, each row and column represents input image and system response, respectively. The

number indicates the classification ratio by the system for each vehicle, while the numbers in parentheses represent the ratio of the correct pose as well as correct vehicle class. For example, the BMP was correctly identified in 90% of the tests, but only 75.6% found the correct pose.

Table 12: Observations

Input\Response	BMP	BTR60	KTANK
BMP	90%(75.6%)	10%	0%
BTR60	6.7%	93.3%(86.7%)	0%
KTANK	10%	11.1%	78.9%(68.9%)

By closely examining the results of the performance evaluation, we have made several observations:

- **More sampled images are necessary:** The reported system sampled training images every 10 degrees. However, for more robust recognition, it is desirable to sample training images at shorter intervals.

- **Partial views are necessary for reliable indexing:** The performance of the indexing module rapidly deteriorates under occlusion. Even though we have introduced redundancy in the invariant histograms by including all translation invariant values given by all feature pairs, those values are obtained from the entire image. When a part of a target is occluded, the histogram varies, often resulting in failure of indexing. It is necessary to design an indexing module based on partial views of a target.

- **Photometric invariants are necessary for reliable verification:** The performance of the verification module rapidly deteriorates under camouflage. This is because the deformable template runs on the potential field given from the brightness values. When the input brightness varies due to camouflage, the brightness distribution varies, and so does the potential field. As a result, verification errors increase. To avoid this effect, it is necessary to introduce a photometric invariant distribution for verification.

8. Dense Sampling of Views for Indexing Library

Dense sampling of views requires a large amount of memory space for an indexing library. We can reduce the memory size by using the eigenspace method. For each target, a set of densely sampled large histograms is obtained. This histogram set is compressed to obtain a low-dimensional subspace, referred to as the eigenspace, in which the histogram set is represented as a hypersurface. Given an unknown input image and, thus, an invariant histogram of the input image, the recognition system projects the input histogram onto the eigenspace. The position of the projection on the hypersurface determines the target pose in the image.

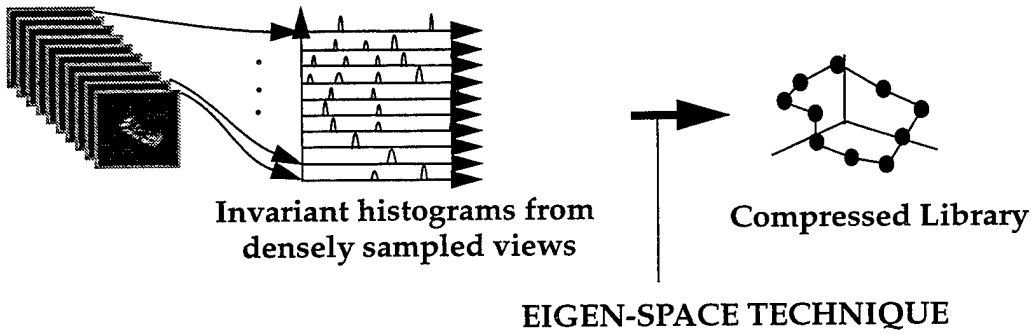


Figure 17 Compression of Indexing Library

In off-line mode, the eigenspace representation is generated from a set of invariant histograms, denoted as $H = [h_1, h_2, \ldots, h_n]$, where h_i denotes the i-th invariant histogram in the library. The training module obtains a product matrix of H and its transpose matrix, H^t,

$$Q = HH^t$$

and then calculates eigenvectors, $\hat{e}_i$ and eigenvalues, λ_i, of this matrix Q. By selecting $m (m < n)$ significant eigenvalues, it forms an eigenmatrix E from m corresponding eigenvectors. Using these eigenvalues and the eigenmatrix, all the histograms h_i are projected on a hyperplane in the (reduced-sized) eigenspace by $g_i = E^t h_i$. The eigenvalues, eigenvectors, and projected points are stored as models by the training module.

In on-line mode, from the input image, an invariant histogram, x, is generated. Then, this invariant histogram is projected in the eigenspace by $g = E^t x$ By searching for near-by points on the

hyperplane in the eigenspace, object indexing is achieved.

We have implemented this method and evaluated its performance using K-tank SAR appearances. We have generated five series of 180 K-tank images (64 X 64) with the following pixel shifts: 0 pixel shift, 8 pixel shift, 16 pixel shift, and 20 pixel shift. Each image is shifted a certain amount from the center position; if pixels move outside of the window (64 X 64), we consider them to be occluded. Figure 18 shows the comparison of the new and old indexing module. In both cases, for the sake of clear comparison, we use the single-choice strategy that considers the indexing to be a success only when the highest scored pose is the correct pose. In both cases, the new indexing module outperformed the old module.

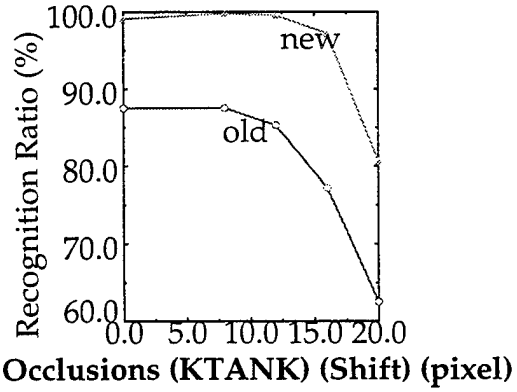


Figure 18 The Indexing Results of the New/Old Modules

9. Partial View Windows in Indexing Library

The main problem of the invariant-histogram under occlusion derives from its use of the entire image. Our approach to resolving these limitations is to divide an image into small windows. In order to reduce the memory size, we can apply the eigenspace method to these windows, mapping each one separately into the eigenspace. We will refer to this method as the"eigenwindow method." The small windows have two benefits. First, when some parts of a target are occluded, remaining windows defined on visible parts can identify the target. Second, to detect a target with articulated components, we can define separate windows on articulated parts and the body. The recognition can proceed separately on the articulated parts and the body.

In off-line mode, the training module divides training images into a collection of windows. Then, it calculates eigenvalues and eigenvectors from these windows, determines significant eigenvalues, spans the eigenspace using the significant eigenvectors, and maps those windows into eigenpoints. In on-line mode, a matching module divides an input image into windows and projects them in the eigenspace. By matching them with the nearest training eigenpoint, each window is assigned to one part under one particular target pose. The voting module prepares voting spaces and accumulates possible poses in voting space. The pruning module prunes unpopular candidate poses and determines the most likely pose.

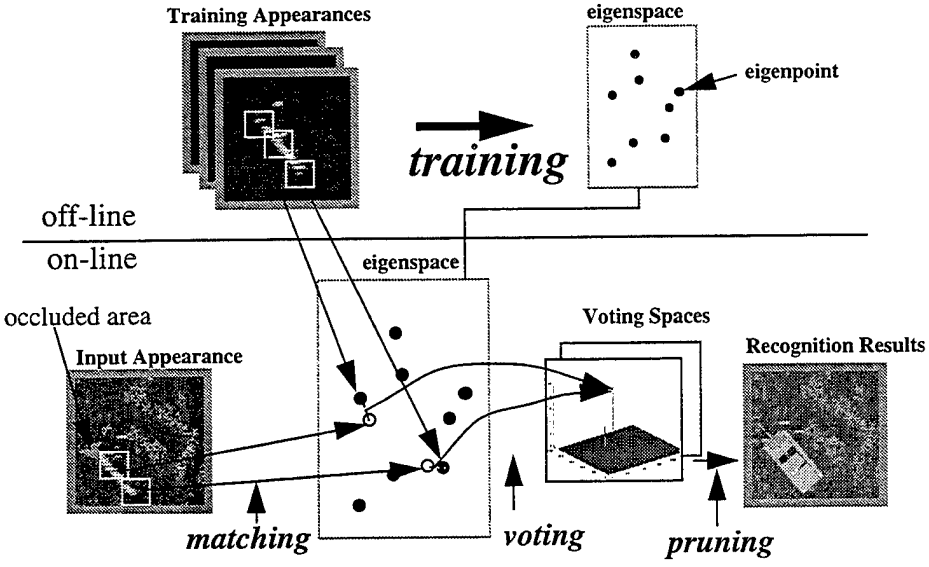


Figure 19 Eigenwindow-Based ATR System

9.1. Off-line Mode

The training set of eigenwindows is given from the training images as

$$W = [W_1, W_2, \ldots, W_n]$$
$$= [[w_{11}, w_{12}, \ldots, w_{1l}], [w_{21}, w_{22}, \ldots, w_{2l}], \ldots, [w_{n1}, w_{n2}, \ldots, w_{nl}]]$$

Here, W_i denotes the collection of eigenwindows from the i-th training image, for which w_{ij} denotes the j-th eigenwindow.

The eigenwindow method considers all of the windows as a new training set, and generates eigenpoints from them. It considers the W matrix as a collection of independent small windows $W = (x_1, x_2, \ldots, x_k)$, where $x_k = w_{ij}$.

From this and its transpose matrix, the system calculates eigenvectors, and eigenvalues. Then, by selecting significant eigenvalues, it forms an eigenmatrix. Using these eigenvalues and the eigenmatrix, all the training images are projected to points in the (reduced-sized) eigenspace. The resulting representation forms a cloud of points in the eigenspace.

The eigenvalues, eigenvectors, and projected points are stored as models in the training module. For the purpose of pose determination, the relative position of the window with respect to the image center is also calculated and stored at each eigenpoint.

9.2. On-line Mode

9.2.1. Matching Module

The matching module divides an input image into windows, projects them in the eigenspace, searches for model eigenpoints near the input eigenpoints, and establishes pairs among them.

The training module provides a cloud of model points in the eigenspace. These points do not necessary form a continuous hypersurface in the space. Rather, they are a collection of independent random points. This is because the eigenwindow method divides a set of training images into windows and there is no continuous parameterization among windows. Brute force search for pairings requires prohibitive computational time.

9.2.2. Voting Module

Each window in an input image corresponds to one part of one target under one pose. Over an input image, many interpretations are generated-- some are consistent, while others are contradictory. We are developing a method to reach a consensus among these hypotheses through voting.

First, interpretations will be classified into groups so that each group represents one common interpretation (the same target at the same pose). For each group, a voting space will be created to accumulate votes for the position of the target in the image. In off-line mode, each eigenwindow was associated with its relative coordinates with respect to the center of the model. Retrieving the coordinates, each interpretation votes for a center position of the input image in the voting spaces. In order to absorb the digitization error, a single interpretation actually votes for a 5X5 region of cells around the calculated center. We repeat this operation using all the interpretations in the group.

9.2.3. Pruning Module

Some small peaks are due to noise; other prominent peaks are due to actual targets in the input image. By thresholding these peaks, the pruning module eliminates noise peaks and extracts the prominent peaks in the voting spaces.

9.3. Performance Evaluation

We have finished the preliminary implementation of this module and evaluated it. We have generated 9 series of 360 K-tank images (64 x 64) with from 0 pixel shift through 36 pixel shift, as given in Table 13. Each image is shifted a certain amount from the center position; if pixels move outside of the window (64 x 64), we consider them to be occluded. For example, at a 20 pixel shift, roughly one half of the original pixels are lost in the worst case.

Table 13 contains the evaluation results for K-tank under the single-choice strategy that considered the recognition correct only when the top scored one is the correct candidate. Since the new indexing module utilizes training images sampled at every 1 degree, evaluation images were generated at the fractional angles *.1, *.3, *.7 and *.9 for using different images between training and

evaluation. Also, for comparison with the previous indexing module, determined poses were counted as correct matches if they were +/- 5 degrees from the true pose. This comparison concluded that the new method is more robust against occlusion, in particular, after 16 pixels shift.

Table 13: Comparison Under Single-Choice Strategy

shift (pixel)	0	8	12	16	20	24	28	32	36
old	87.5	87.5	85.3	77.2	62.5	---	---	---	---
new	89.5	89.5	89.5	89.1	85.3	84.5	79.0	51.7	34.7

We have extensively evaluated this module under multiple-choice strategy. Since, in an operational scenario, an indexing module is supposed to select a small number of multiple candidates quickly, later the verification module will determine the correct candidate after the precise examination of part existence. Here, the old and new indexing modules employ slightly different selection strategies. The old module selects the candidates with 80% or more of the best score, and is considered as successful if the candidate set contains the correct pose. From the previous evaluation, this selection strategy usually generated more than 5 candidates. The new module simply chooses five of the most likely candidates and is considered a success if that set contains the correct pose. As a matter of fact, this comparison favors the old method because it generates more candidates -- sometimes as many as 10 -- under severe occlusion. Yet, the results, shown in Tables 14, 15, and 16 indicate that the new module outperformed the old module, in particular, under severe occlusion. Figure 20 shows the graphical representation of this result.

Table 14: KTANK

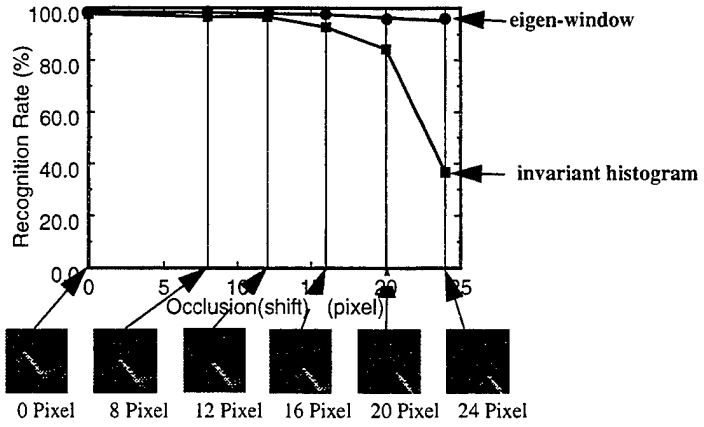
Occlusion (shift)	Invariant histogram	Eigen window
0 pixel	97.8%	97.1%
8 pixel	96.7%	97.1%
12 pixel	96.7%	97.1%
16 pixel	92.8%	97.1%
20 pixel	84.4%	95.1%
24 pixel	36.7%	94.3%

Table 15: BMP

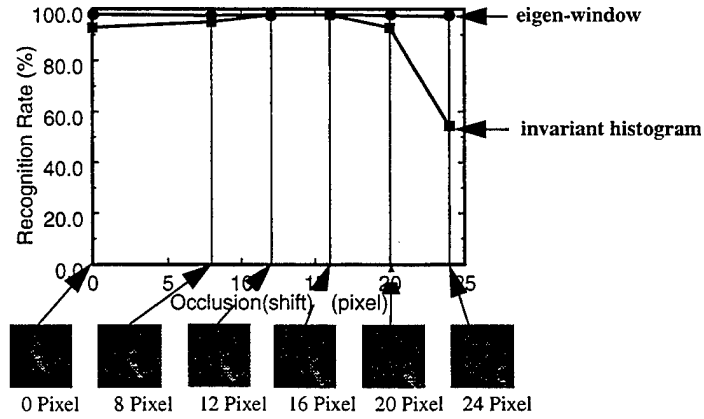
Occlusion (Shift)	Invariant histogram	Eigen window
0 pixel	92.8%	99.5%
8 pixel	95.0%	99.5%
12 pixel	97.8%	99.5%
16 pixel	97.8%	99.5%
20 pixel	92.8%	99.4%
24 pixel	54.4%	98.5%

Table 16: BTR60

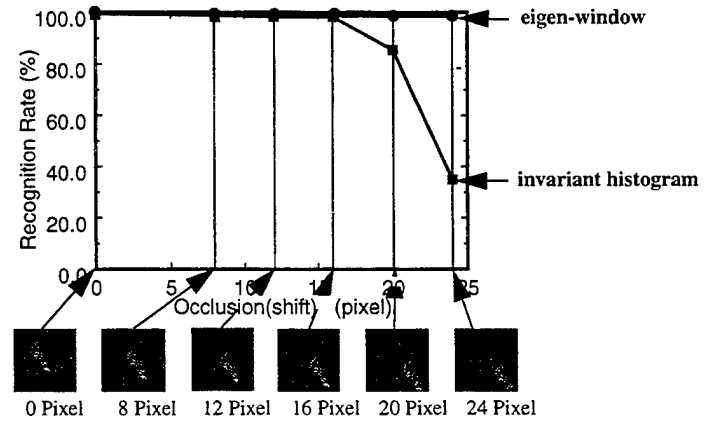
Occlusion (shift)	Invariant histogram	Eigen window
0 pixel	99.4%	99.3%
8 pixel	98.9%	99.3%
12 pixel	98.3%	99.2%
16 pixel	98.3%	99.0%
20 pixel	85.6%	98.5%
24 pixel	35.0%	97.7%



(a) K-TANK



(b) BMP



(c) BTR60

Figure 20 Performance Evaluation of Eeigen-Window Indexing

9.4. Photometric Invariants for Verification

The verification module employs potential fields defined by input brightness. Thus, it is susceptible to variance in brightness due to camouflage. In order to reduce this effect, we are developing methods based on photometric invariants whose distribution is independent of absolute brightness distributions.

For example, SAR signatures have three polarization components: HH, HV, and VV. When returned signals are reduced due to camouflage, those components are also reduced. However, their relative strength will not be affected. We plan to use the ratio of polarizations to compensate for the variance in the absolute brightness distribution: $I = (HV)/(HH)$.

10. Summary

This project develops a SAR ATR system that uses invariant histograms and deformable templates. An invariant histogram is a histogram of geometric invariants given by primitive feature sets. Deformable template matching examines the existence of an object by superimposing templates over potential energy fields derived from the image so that it generates the minimum deformation (deformation energy) and the best alignment of the template with features (potential energy).

We have demonstrated the effectiveness of these two techniques for robust SAR recognition through extensive evaluation of the system using occluded and camouflaged target images.

This system has two modes: off-line and on-line. In off-line mode, the system generates a library for indexing and deformable templates for verification. Currently, it takes a half hour for this compilation on a SPARC 20. In on-line mode, by calculating an invariant histogram from an input image, the system performs the indexing to reduce the number of possible candidates. Then, using the potential fields from the input image and the deformable templates, the system determines the most likely pose and class of the target. Indexing takes about 2 to 3 seconds, and verification takes a few seconds per candidate pose. The run times include time to build invariant histograms and compute potential fields.

By analyzing the evaluation results, we have proposed three extensions of the system: dense sampling for robust recognition, partial view windows for robust indexing under occlusion, and photometric invariants for robust verification under camouflage. Some of these ideas have been implemented and are being evaluated.

By analyzing the evaluation results, we have proposed three extensions of the system: dense sampling for robust recognition, partial view windows for robust indexing under occlusion, and photometric invariants for robust verification under camouflage. Some of these idea has been implemented and evaluated.

11. References

[1] Tomiyasu, K., "Tutorial review of synthetic-aperture radar (SAR) with applications to imaging of the ocean surface," *Proc. IEEE*, Vol. 66, No.5, pp. 563-583, 1978.

[2] Chellapa, R., E. Zelnio, and E. Rignot, "Understanding synthetic aperture radar images," Proc. Image Understanding Workshop, pp.229-245.1992.

[3] Novak, L.M., G.J. Owirka, and C.M. Netishen, "Performance of a high-resolution polarimetric SAR automatic target recognition system," Lincoln Lab Journal, vol.6, No. 1, pp.11-23, 1993.

[4] Maxman, A.M., M. Seibert, A.M. Bernardon, and D. A. Fay, "Neural systems for automatic target learning and recognition," Lincoln Lab Journal, vol. 6, no. 1, pp.77-116, 1993.

[5] Kuno, Y., K. Ikeuchi, and T. Kanade, "Model-based vision by cooperative processing of evidence and hypotheses using configuration spaces," *Proc. SPIE,* Vol. 938, pp.444-453, 1988.

[6] Sato, K., K. Ikeuchi, and T. Kanade, "Model based recognition of specular objects using sensor models," *CVGIP: Image Understanding*, Vol. 55, No.2, pp.155-169, 1992.

[7] Gremban, K.D. and K. Ikeuchi, "Appearance-based vision and the automatic generation of object recognition program," T*hree-dimensional object recognition systems*, A.K. Jain and P.J. Flynn (eds.), Elsevier Publishers, pp.229-258, 1993.

[8] Grimson, W. E. L., *Object recognition by computer*, The MIT Press, 1990.

[9] Lowe, D. G., *Perceptual organization and visual recognition*, Kluwer Academic Publishers, 1985.

[10] Bolles, R. C. and M. A. Fischler, "A RANSAC-based approach to model fitting and its application to finding cylinders in range data," *Proc. IJCAI*, pp.637-643, 1981.

[11] Brooks, R., "Symbolic reasoning among 3-dimensional models and 2-dimensional images," *Artif. Intell.*, vol.5, no.5, pp.493-505, 1983.

[12] Ayache, H. and O. D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Trans. PAMI*, vol.8, no.1, pp.44-54, 1986.

[13] Bolles, R. C. and R. A. Cain, "Recognizing and locating partially visible objects: The local feature focus method," *Int. J. Robotics Res.*, vol.1, no. 3, pp.56-82, 1982.

[14] Matsuyama, T., H. Arita and M. Nagao, "Structural matching of line drawings using the

geometric relationship between line segments," *Computer. Vision, Graphics, Image Proc.*,vol.27,pp.177-194,1984.

[15] Ikeuchi, K., "Generating an interpretation tree from a CAD model for 3D-object recognition in bin-picking tasks," *Int. J. Comp. Vision*, vol.1, no.2, pp.145-165, 1987.

[16] Ikeuchi, T. and T. Kanade, "Toward automatic generation of object recognition programs," *Proc. of IEEE,* vol. 76, no.8, pp.1016-1035, 1988.

[17] Wheeler, M. and K. Ikeuchi, "Sensor modeling, probabilistic hypothesis generation, and robust localization for object recognition," IEEE Trans. PAMI, vol. 17, no.3, pp.252-265, 1995.

[18] Ballard, D. H., "Generalizing the Hough Transform to detect arbitrary patterns," *Pattern Recognition*, vol. 13, no. 2, pp. 111-122, 1981.

[19] Lamdan, Y and H. J. Wolfson, "Geometric Hashing: A general and efficient model-based recognition scheme," *Proc. ICCV'88*, pp.238-249, 1988.

[20] Dhome, M., M. Richetin and G. Rives, "Model-based recognition and location of local patterns in polygonal contours via hypothesis accumulation," *Pattern Recognition in Practice II,* edited by E. S. Gelsema and L. N. Kanal, North-Holland, 1986.

[21] Stockman, G. S., "Object recognition and localization via pose clustering," *Computer. Vision, Graphics, Image Proc.*, vol. 40, pp.361-387, 1987.

[22] Wolfson, H. J. and Y. Lamdan, "Transformation invariant indexing," in *Geometric Invariance in Computer Vision* (J. Mundy and A.Zisserman edit, MIT Press), pp.335-353, 1992.

[23] Clemens, D. T. and D. W. Jacobs, "Model group indexing for recognition," *IEEE Trans. PAMI*, vol.13, no.10, pp.1007-1017, 1991.

[24] Kiryati, N., Y. Eldar and A. M. Bruckstein, "A probabilistic Hough transform," *Pattern Recognition*, vol. 24, pp.303-316, 1991.

[25] Xu, L. and E. Oja, "Randomized Hough transform (RHT): Basic mechanisms, algorithms and computational complexities," *CVGIP: Image Understanding*, vol. 57, pp.131-154, 1993.

[26] Hut10locher, D. P. and S. Ullman, "Object recognition using alignment," *Proc. ICCV'87*, pp. 102-111, 1987.

[27] Mundy, J. L. and A. Zisserman, edit, *Geometric Invariance in Computer Vision*, The MIT Press, 1992.

[28] Binford, T. O. and Levitt,T.S., "Quasi-invariants: theory and exploitation," *Proc. IUW93*, pp.819-829, 1993

[29] Kanatani, K., *Geometric computation for machine vision*, Oxford university press, 1993.

[30] Kass, M., A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," Int. J. Com. Vision, Vol. 2, No.1, pp.321-331,1988.

[31] Bessette, L. A., Stockbridge Mission 85 Pass 5 Data Package, MIT Licoln Laboratory Project Report TT-80, 8 May1991

[32] Turk, M. A. and Pentland, A. P., "Face recognition usin eigenfaces," *Proc. CVPR'91*, pp.586-591, 1991.

[33] Murase, H. and S. K. Nayar, "Illumination planning for object recognition in structured environment," *Int. J. Computer Vision*, vol. 1, no. 2, pp.145-165, 1987.