SAR Target Recognition Based on Invariant Histograms and Deformable Template Matching

Katsushi Ikeuchi, Takeshi Shakunaga, Mark D. Wheeler and Taku Yamazaki

March 1996
CMU-CS-95-198

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

Abstract

Recognizing a target in synthetic-aperture radar (SAR) images is an important, yet challenging, application of the model-based vision technique. This paper describes a model-based SAR recognition system based on invariant histograms and deformable template matching techniques. An invariant histogram is a histogram of invariant values defined by geometric features such as points and lines in SAR images. Although a few invariants are sufficient to recognize a target, we use a histogram of all invariant values given by all possible target feature pairs. This redundant histogram enables robust recognition under severe occlusions typical in SAR recognition scenarios. Multi-step deformable template matching examines the existence of an object by superimposing templates over potential energy field generated from images or primitive features. It determines the template configuration which has the minimum deformation and the best alignment of the template with features. The deformability of the template absorbs the instability of SAR features. We have implemented the system and evaluated the system performance using hybrid SAR images, generated from synthesized model signatures and real SAR background signatures.

This research was sponsored in part by the Defense Advanced Research Project Agency (DoD) and monitored by Wright Research and Development Center, U.S. Air Force under Contract F33615-93-1-1282, in part by the Defense Advanced Research Project Agency and monitored by the Army Research Office, the Department of the Army under Grant DAAH04-94-G-0006, and in part by the Office of Naval Research under Grant N00014-93-1-1220.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Agency, Wright Research and Development Center, the Army Research Office, the Office of Naval Research or the U.S. Government.

Keywords: Synthetic-Aperture Rader (SAR), Automatic Target Recognition (ATR), Object Recognition, Geometric Invariance, Deformable Template

1. Introduction

Recognizing a target in synthetic-aperture radar (SAR) images [1]-[2] is a difficult problem for conventional computer vision systems. A SAR image is a collection of radar return signals, accumulated along the flight path of aircrafts or satellites. They are not tightly related with surface markers nor explicit object geometry such as edges. Rather, they are floating over a target surface like specular spots on a metal surface, which we referred to as non-attached features. With slight movements of the observer, features suddenly appear, disappear, and abruptly change their shapes. Secondly, in SAR image recognition, objects are often intentionally hidden from an observer. Thus, to achieve robust target recognition for SAR ATR scenarios, we must overcome unstable features, occlusion, and camouflage.

Historically, target recognition in SAR images is attacked using three different approaches: statistical pattern recognition [3], artificial neural networks [4], and model-based vision [5]-[6]. In essence, a model-based system analyzes each image in detail and identifies each part of a signature's 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 greater potential for robustness with respect to partial occlusion of the target and cluttered backgrounds. For these reasons, model-based approach [5]-[17] is the most promising.

Among several proposed model-based techniques, pose clustering is suitable for determining the object pose (and identifying the object) from sparse features typical of SAR images. 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], and 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 in 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 clue 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 for 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, the combinatorics becomes unwieldy when handling dense features typical of SAR images.

To avoid the difficult correspondence problem, this paper introduces an invariant histogram. Our invariant histogram is based on *weak invariants*, defined by pairs 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. 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 representation. To select the correct candidate, the verification module employs deformable template matching to precisely locate and test for the existence of each predicted feature. Deformations are necessary for fine-tuning each feature positions locally, since each SAR feature is non-attached and can vary its position.

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 experimental results, and in Section 6 we present our conclusions.

2. Invariant histogram

In order to achieve robust recognition under severe occlusion, camouflage, and unstable SAR features, we introduce 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.

The invariant histogram is used by our indexing module. The indexing module is designed to quickly reduce the number of the possible candidates before expensive candidate verification. It employs a dictionary lookup method. The dictionary consists of the invariant histograms, associated with various poses. By comparing the observed invariant histogram with model histograms in the dictionary, the module decides which candidate poses are the most likely ones. This process requires a measure of the similarity between an input and a model invariant histogram. In this section, we will discuss the similarity measure defined on the invariant histogram.

2.1. Concept of invariant histogram

Traditionally, invariants have been used as the key to identifying objects from a relatively small set of features. In practice, invariants are not truly invariant due to quantization effects and noise. Though only a few invariants are actually necessary for recognition, an invariant histogram stores many invariant values from a target model, Thus, this invariant histogram forms a redundant representation that is robust against variation in computed invariant values typical of SAR data.

This paper employs weak invariants, such as distances between two points or the slope of the bisecting line between 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 strong invariants require too many primitives to be detected reliably (for example a cross ratio needs four points to be identified in an image). In general the invariant nature does not hold if features are unstable. 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 found using a line detector based on Canny's edge detecting technique.

When a target rotates in 3D space, the appearance of the target in SAR images can drastically change. On the other hand, as a target translates along the ground plane, the appearance is not significantly altered. Thus, we use translation invariants to construct invariant histograms. Figure 1 shows six translation invariants which are used for constructing our invariant histograms.[1] Each invariant is between two features: two points (PP), two lines (LL), or a point and a line (PL).

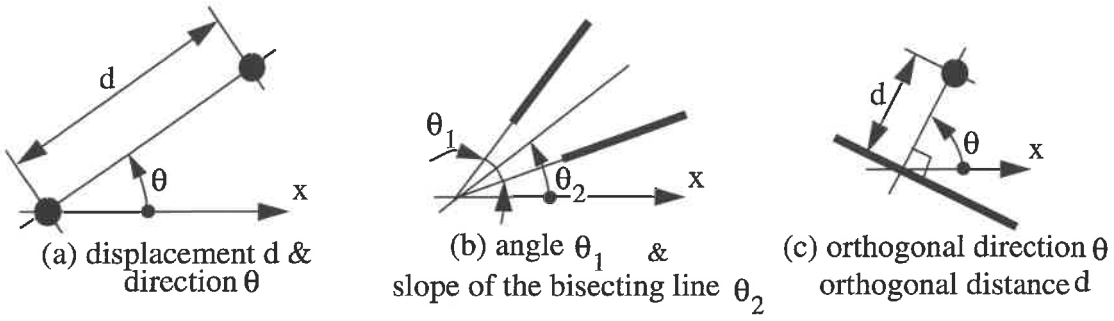
The first two invariants we use are the distance and angular direction of the segment connectiong a pair of points. These translation invariants are depicted in Figure 1(a) and represented in the 2D Point-Point (PP) histogram space[2].

The second pair of invariants are computed using two line segments. The angle between two line segments and the slope of their bisecting line are invariant to translation (Figure 1(b)). These invariants are represented in the 2D Line-Line (LL) histogram space[3].

The last two invariants that we use are between a point and a line segment. The orthogonal distance from a point to a line is invariant to translation and rotation. The orthogonal direction from a point to a line segment is also invariant to translation (Figure 1(c)). These are in the 2D Point-

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.

Line (PL) histogram space[4].



(a) displacement d & direction θ

(b) angle θ_1 & slope of the bisecting line θ_2

(c) orthogonal direction θ orthogonal distance d

Figure 1 Six invariants used in our implementation

2.2. Implementation

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 into 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 using PP as an example. First, primitive features, point features in this example, are extracted from an image. From point pairs, two invariant values are obtained: distance and direction. When making point pairs, the system considers only local feature pairs, those within a certain distance of each other, indicated by the circle in Figure 2. The horizontal axes in the figure indicate values of distance

4. 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

and direction, and the vertical axis denotes the number of votes in each bin.

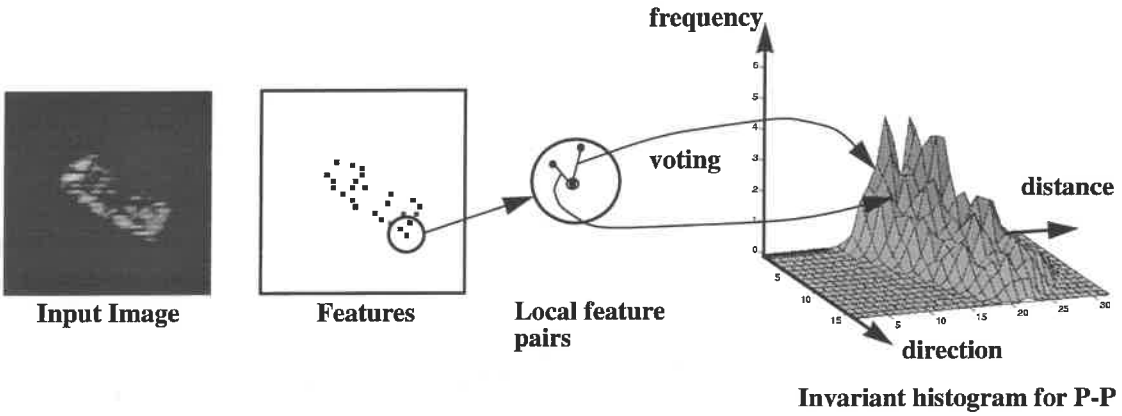


Figure 2 Making invariant histograms

To minimize problems due to quantization, we smooth the histogram over small neighborhoods. 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 each feature pair are copied into these four bins.

3. Indexing by Dictionary Lookup

We have described the details of our invariant histogram representation. Now we will describe how we utilize these histograms to screen or eliminate candidate hypotheses in our indexing module. We can build invariants for each representative view, and use these histograms to compute distance measures between an invariant histogram 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 a dictionary which represents how a target object appears, and thus, invariant values change depending on pose parameters. This dictionary is constructed from model appearances off-line.

3.1. Structure of a dictionary

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. We assume that the target is on the ground plane and the image is taken from a known depression angle. Thus, we have three degrees of freedom (DOF), two translation DOF and one rotation DOF. 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 a dictionary, a collection of invariant histograms, to cover all of the variant parameter space. The rotation parameter space is evenly sampled, and invariant histograms are constructed at these sampled rotation values. Here, each sampled rotation value is denoted as a representative view.

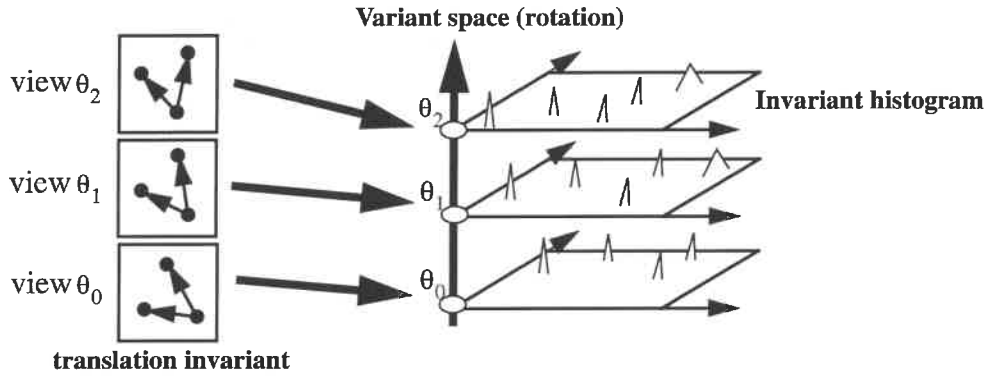


Figure 3 Invariant and variant spaces

We compute the average and variance histograms over an interval surrounding each representative view. 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 dictionary entry.

3.2. Similarity measure for invariant histograms

This section will describe a similarity measure for comparison between image and model histograms in a dictionary. 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 the variance histogram.

One simple similarity measure is the L1 norm as follows:

$$L_1 = \sum_{i,j} \left| x_{i,j} - m_{i,j} \right|, \tag{1}$$

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:

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

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. Namely, if the difference between the observed and model values are larger than this saturation factor, the penalty imposed is the saturation factor instead of the real distance:

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

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

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

The two constants, σ_o and k are obtained empirically.

3.3. Implementation of Indexing Algorithm

Using $L_{1,\,sat}$ 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 all the histograms.

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 $L_{1,\,sat}$ of 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

$$a_{max} = max_m \{L_{1,\,sat}(m)\} \,. \tag{5}$$

where m is over the set of dictionary histograms. 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 the weighted sum:

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

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 function of indexing is not to determine one particular view but to select multiple possible candidate views, we select those with distance less than a certain thresh-

old value below the best candidate.

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 dictionary 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. Using the 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 comparison using the mid-points of the line segments. If these two translation values are similar, 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 an image histogram bin, and m model feature pairs are referenced from the corresponding model histogram bin. We will consider mn possible correspondences for this bin. Note 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's invariants randomly occur in the input image is very low compared to actual occurrences due to the object's 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.

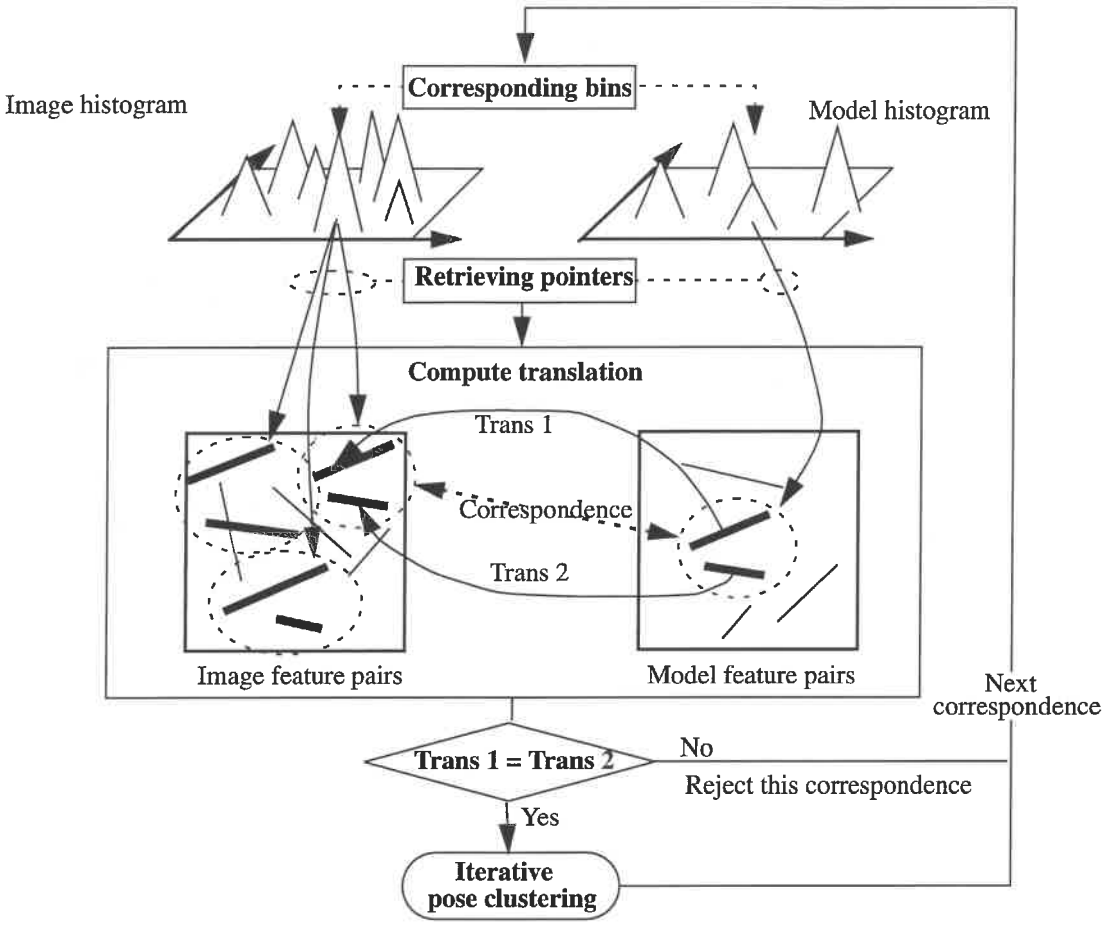


Figure 4 Pose clustering through an invariant histogram

4.2. Successive Pose Clustering

Several kinds of techniques have been used for pose clustering [18]-[25]. The most popular technique is the generalized Hough transform in which voting is applied to a 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 an iterative 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. We first look for a cluster within a certain distance of the pose candidate. If it is close to several clusters, the largest cluster will be selected to include the new candidate. The average pose and the size of the cluster is 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 the final result.

Pose clustering provides a rough estimate of translation parameters (invariant parameters), while a dictionary lookup gives a rough estimate of rotation parameters (variant parameters). Using these estimates of the pose parameters, the precise pose is found using deformable template matching.

5. Deformable Template Matching

Deformable template matching precisely examines the existence of a target by superimposing and aligning deformable templates over potential energy fields defined by image features. Deformation is necessary to robustly verify the existence of an object since most features in SAR images are non-attached. We employ multi-step 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 pose clustering. Multiple-level template and potential fields are progressively used to refine the template parameters at coarse, medium, and fine levels.

5.1. Template Generation

The model templates are generated off-line from model appearances using a SAR simulator. The first two levels of template matching, the coarse and medium level, share the same non-deformable template, while the fine level matching allows deformation of the template. These templates are generated at each representative view used for the indexing dictionaries.

5.1.1. Coarse/Medium level Template

Non-deformable 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 eleven times around a representative view and superimpose them taking the union. The resulting binarized point distribution, $T^{\theta_o}(x, y)$, is the template at the central viewing direction, θ_o (i.e., the representative view).

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

Combining the templates is necessary to absorb all unstable SAR non-attached 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

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

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

$P(x, y)$ 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 variation in the 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 non-deformable templates, eleven such point distributions around a representative view, θ_o, are again combined:

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

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

The total energy of this template is:

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

where

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

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

5.1.3. Difference template

For many objects, confusion often occurs between one pose and its counter pose (rotated 180 degrees from the original pose). In order to avoid this pitfall, our system employs a difference-template as the fourth step of matching. This template suppresses common parts and emphasizes conflicting parts between the pair. Difference templates are used only when it is necessary to dis-ambiguate a pair of candidates with close scores.

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 the 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:

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

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 on-line 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 binary image I_{th}:

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

Figure 6 shows the overview of this module.

5.2.1. Coarse level potential field

For the coarse level potential fields, to remove isolated bright pixels, 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}. Finally, we apply an exponential smoothing function with the width k_{coarse}, to this output:

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

The exponential function is preferred over the Gaussian function because it emphasizes the central value and suppresses peripheral areas.

5.2.2. Medium level potential field

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

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

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 an even narrower exponential function to the binary distribution:

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

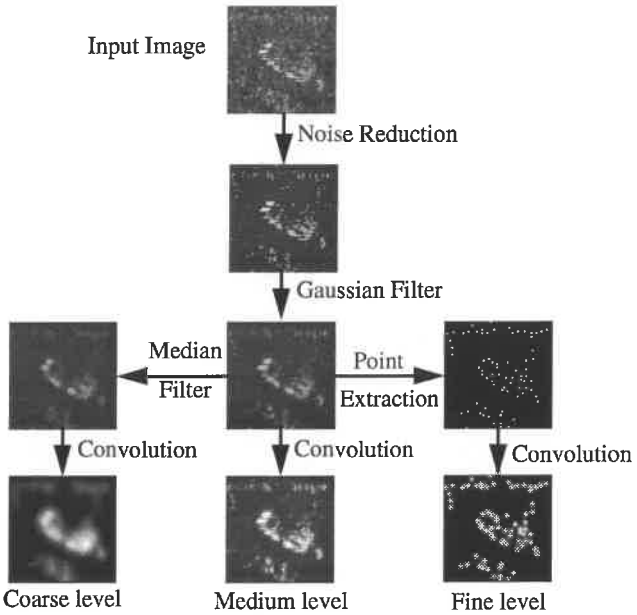


Figure 5 Potential field generation

6. SAR ATR system

Figure 6 shows the overview of the SAR recognition system. It has two modes: off-line and on-line. In off-line mode, the system generates dictionaries for targets using target models and XPATCH SAR simulator, developed by Wright Research and Development Center, WPAFB, [30]. 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.

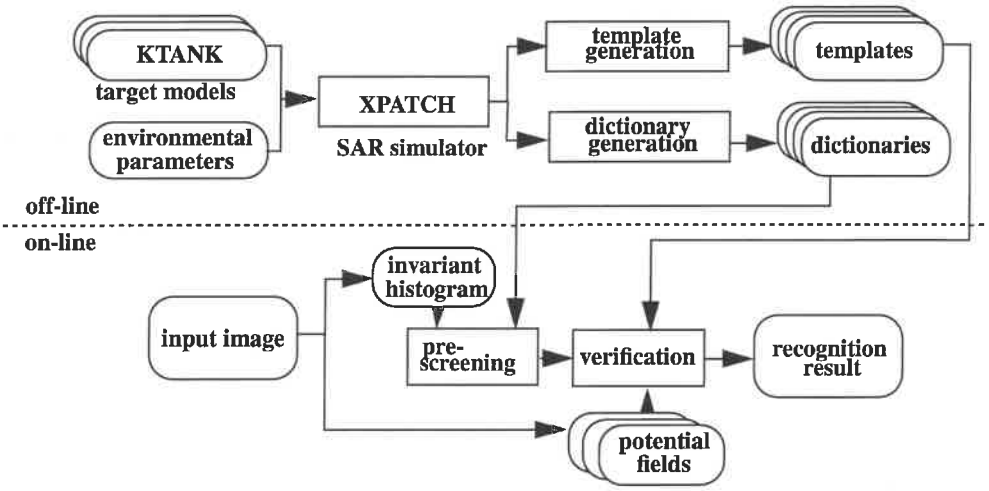


Figure 6 System overview

6.1. 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 30 cm/pixel as the scale. We employ the XPATCH SAR simulator to generate simulated SAR images for the dictionary generation. Figure 7 shows three series of 36 generated images: KTANK, BMP, and BTR60. A dictionary is constructed for 36 views rotated around the axis perpendicular to the ground plane,

sampled every 10 degrees.

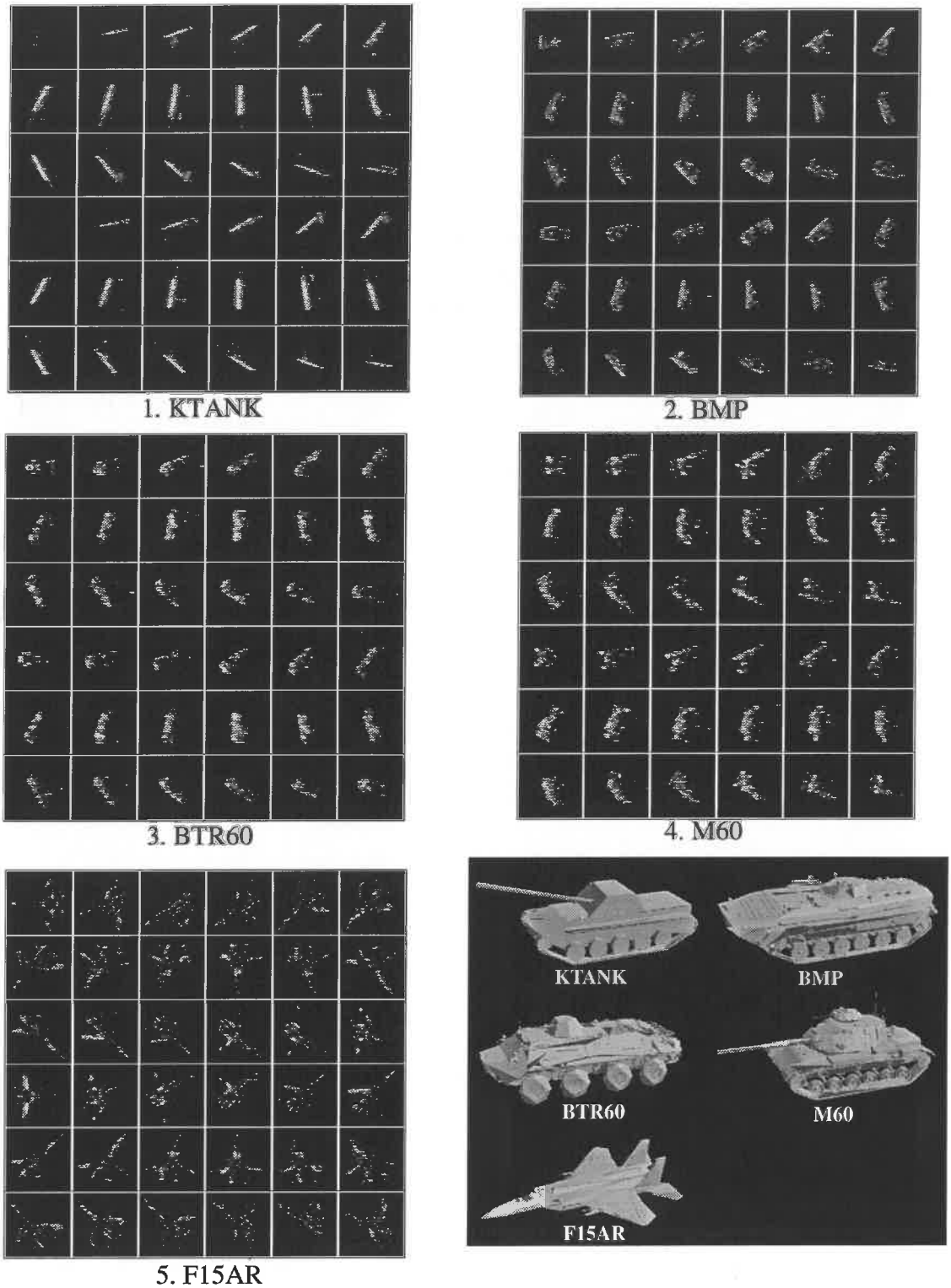


Figure 7 Model SAR images for dictionary generation

In order to obtain an estimate of the variance of each invariant value around a representative view, 19 images around each representative views are generated within 1 degree (0.1 degree intervals). Each representative view of a dictionary consists of three invariant histograms: point-point, line-line, and point-line as shown in Figure 8.

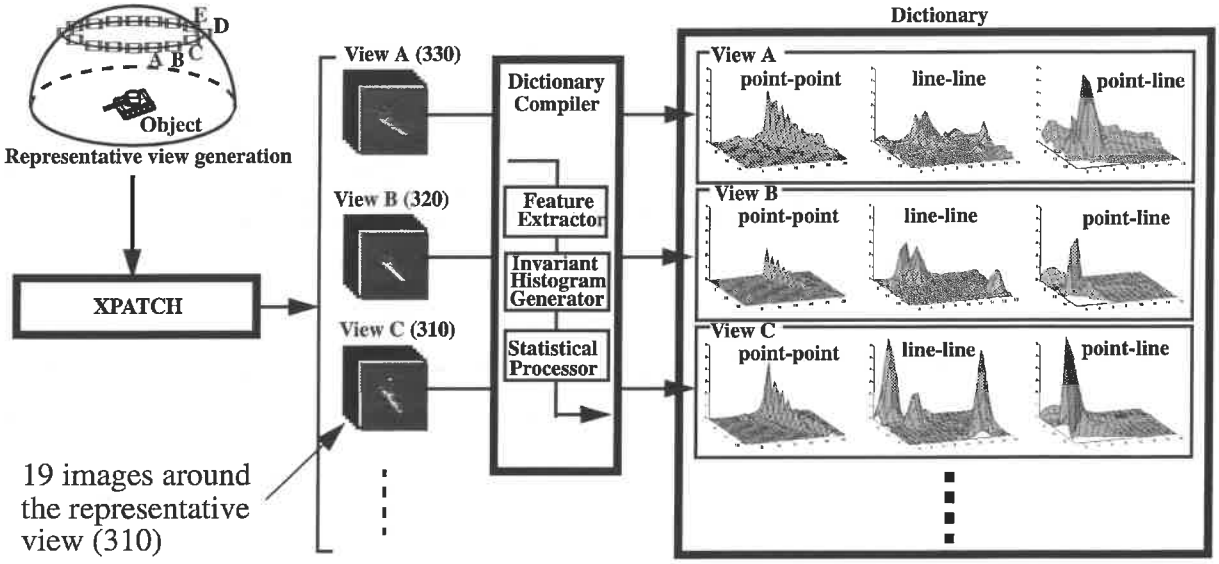


Figure 8 Generating a dictionary

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

6.2. On-line mode

6.2.1. Hybrid SAR image

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

6.2.2. Indexing module

Figure 10 shows the flow of the indexing. From an input hybrid image (rotation of 312 degrees), the module detects point and line features, and then generate three invariant histograms. The indexing module eliminates unlikely candidates by measuring the distance between the input histogram and those in the dictionary and pruning the number of possible objects for recognition using the similarity measure. Our similarity measure function, $L_{1,\,sat} = \sum_{i,j} min\left(\frac{|x_{i,j} - m_{i,j}|}{\sigma_{i,j} + \sigma_o}, k\right)$, has two parameters: k and σ_o. We use $k = 10$ and $\sigma_o = 0.5$. In this example, the module selects five candidates, including 110, 250 and 310 (the correct pose) of KTANK as the possible candidates for verification.

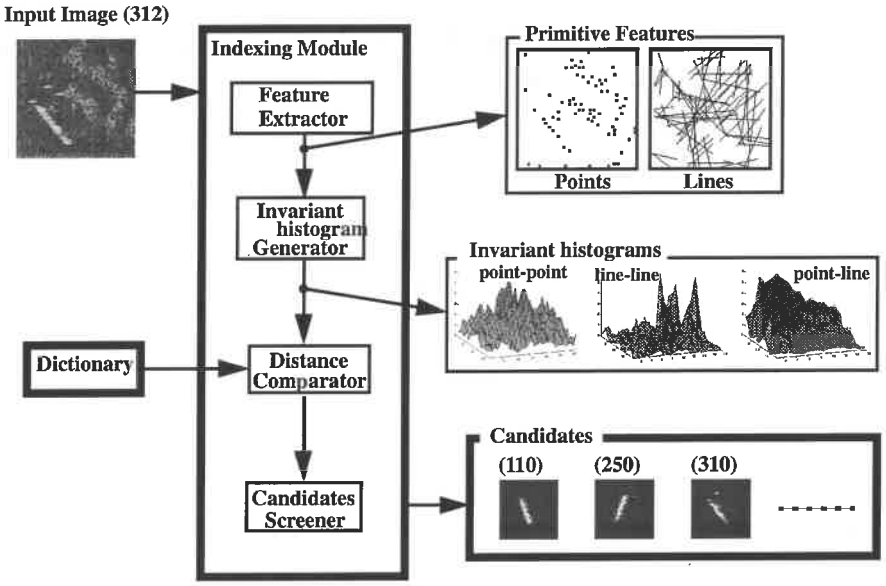


Figure 10 Indexing module

6.2.3. Verificatoin 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).

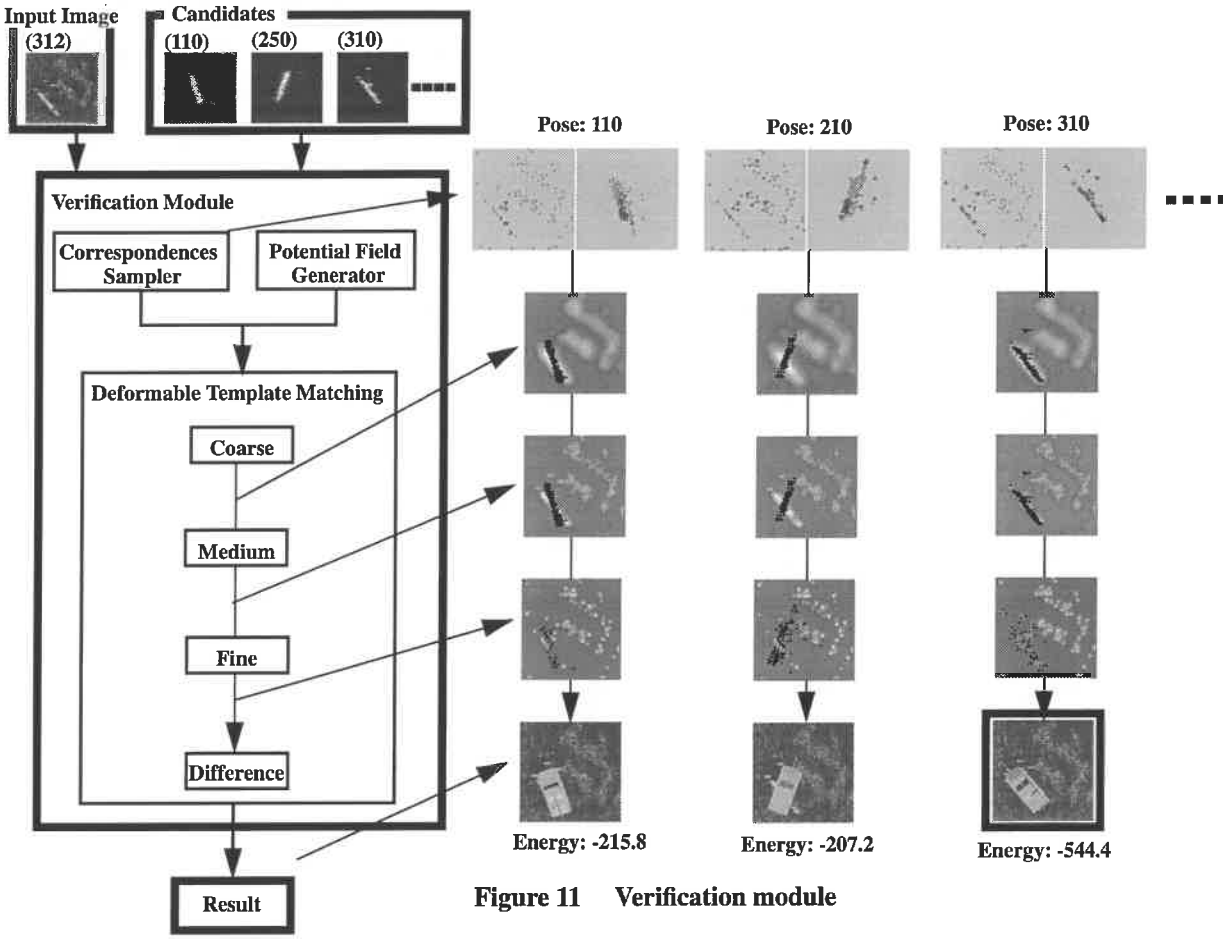


Figure 11 Verification module

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

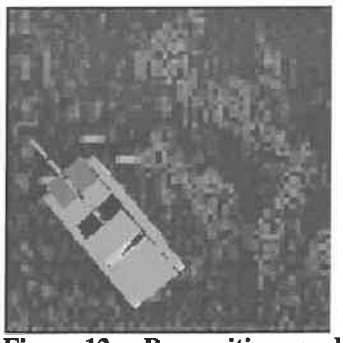


Figure 12 Recognition result

7. Recognition experiments

7.1. Single object database

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

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 dictionary being used for the indexing. For these test images, 5 to 9 candidates were selected on average. The rows in table 1 denotes the results for each object. When the candidate set contains the direction nearest to the input direction, the indexing is considered as 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 every ten 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%

7.2. 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 certain 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 shi roughly one half of the original pixels are lost in the worst case. We evaluate the effect using five targets, KTANK, BMP, and BTR60, M60, and F15AR. Tables 2-6 show the results, while Figure 13 and 14 summarize these results

Table 2: KTANK

Occlusion(shift)	Prescreening	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%

Table 3: BMP

Occlusion(shift)	Prescreening	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%
16 pixel	97.8%	90.3%	88.3%
20 pixel	92.8%	86.8%	80.6%
24 pixel	54.4%	83.7%	45.6%

Table 4: BTR60

Occlusion(shift)	Prescreening	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%
16 pixel	98.3%	94.4%	92.8%
20 pixel	85.6%	90.9%	77.8%
24 pixel	35.0%	85.7%	30.6%

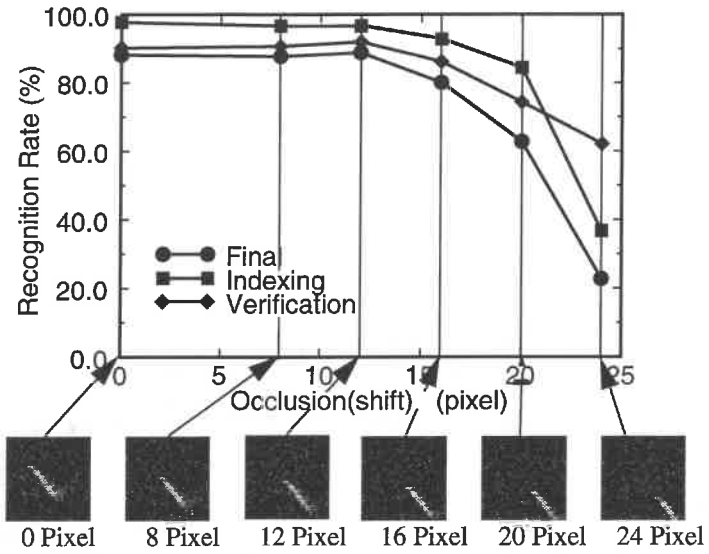
Table 5: M60

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%
16 pixel	96.7%	97.1%	93.9%
20 pixel	73.9%	88.7%	65.6%
24 pixel	21.1%	89.5%	18.9%

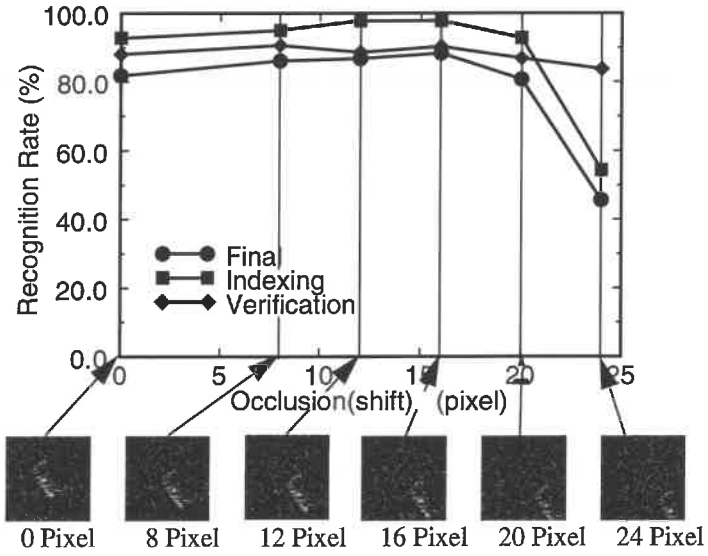
Table 6: F15AR

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%
16 pixel	81.7%	97.3%	79.4%
20 pixel	67.2%	90.1%	60.6%
24 pixel	34.4%	93.5%	32.2%

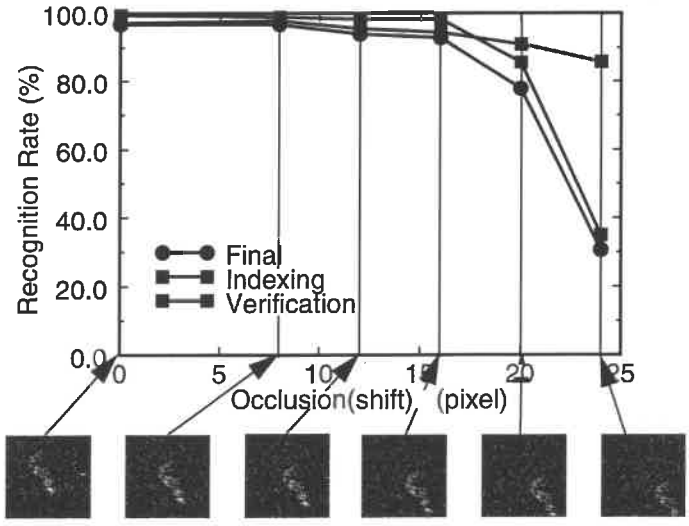
KTANK



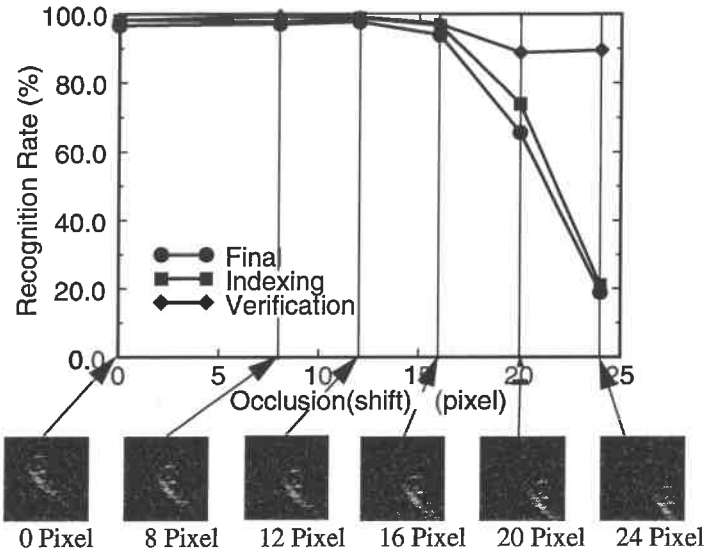
BMP



BTR60



M60



F15AR

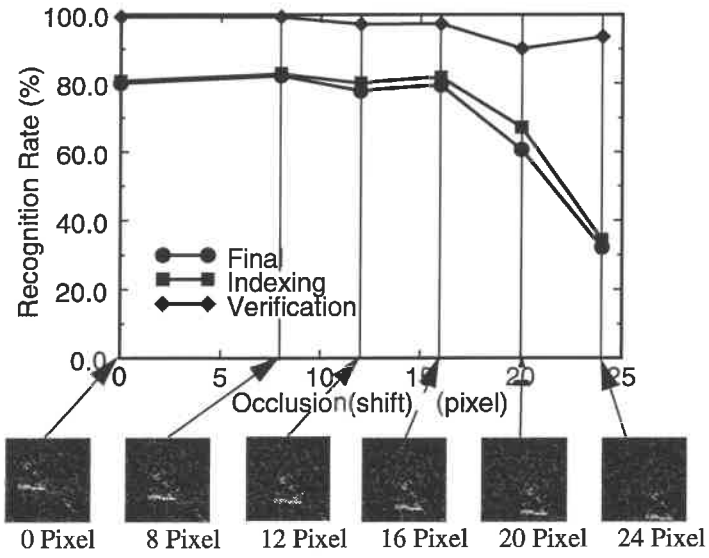


Figure 14 Occlusion(2)

7.3. 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 percentage of camouflaged features (extracted from the original intensity but not in the reduced image). See Figures 15 and 16 for the graphical display of these results.

Table 7: KTANK

Camouflaged features	Indexing	Verification	System
0.0%	98.3%	90.4%	88.9%
5.2%	98.9%	86.0%	85.0%
13.2%	98.3%	83.6%	82.2%
22.3%	93.9%	78.1%	73.3%
33.3%	92.2%	74.7%	68.9%
46.6%	91.1%	68.9%	46.6%

Table 8: BMP

Camouflaged features	Indexing	Verification	System
0.0%	95.0%	87.7%	83.3%
3.8%	96.1%	86.7%	83.3%
12.4%	96.1%	82.1%	78.9%
21.0%	90.6%	78.5%	71.1%
34.9%	82.2%	79.7%	65.6%
67.6%	57.8%	52.9%	30.6%

Table 9: BTR60

Camouflaged features	Indexing	Verification	System
0.0%	100.0%	96.6%	96.6%
3.3%	100.0%	94.4%	94.4%
7.7%	100.0%	95.0%	95.0%
14.6%	97.8%	90.9%	88.9%
22.3%	97.2%	79.4%	77.2%
43.4%	80.6%	69.0%	55.6%

Table 10: M60

Camouflaged features	Indexing	Verification	System
0.0%	98.3%	97.2%	95.6%

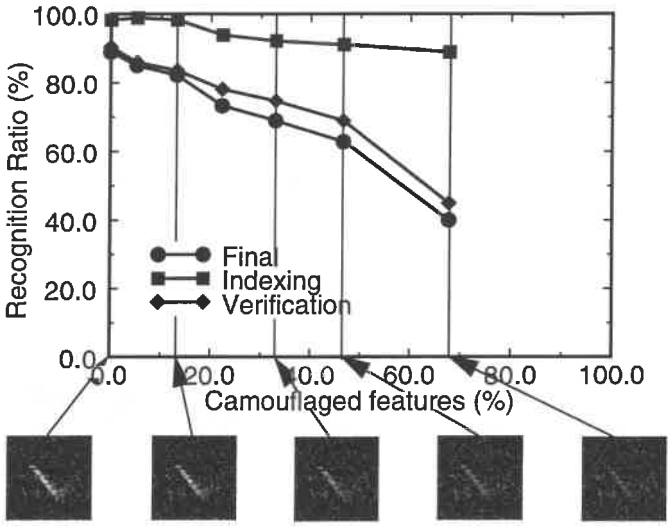
Table 10: M60

Camouflaged features	Indexing	Verification	System
3.4%	98.3%	97.7%	96.1%
8.1%	98.9%	96.6%	95.6%
13.0%	98.9%	92.7%	91.7%
24.1%	93.9%	91.1%	85.6%
54.5%	70.6%	84.3%	59.4%

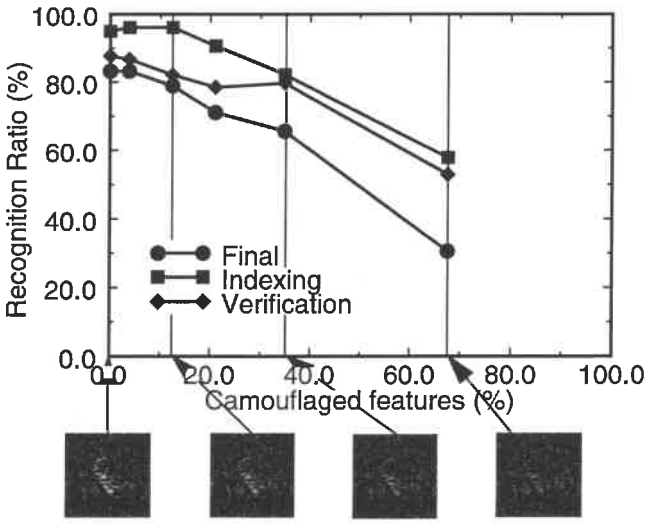
Table 11: F15AR

Camouflaged features	Indexing	Verification	System
0.0%	76.7%	98.6%	75.6%
6.0%	86.7%	98.7%	85.6%
14.0%	87.8%	98.7%	86.7%
27.0%	78.3%	97.9%	76.7%
54.5%	51.1%	87.0%	44.4%
95.5%	26.7%	72.9%	19.4%

KTANK



BMP



BTR60

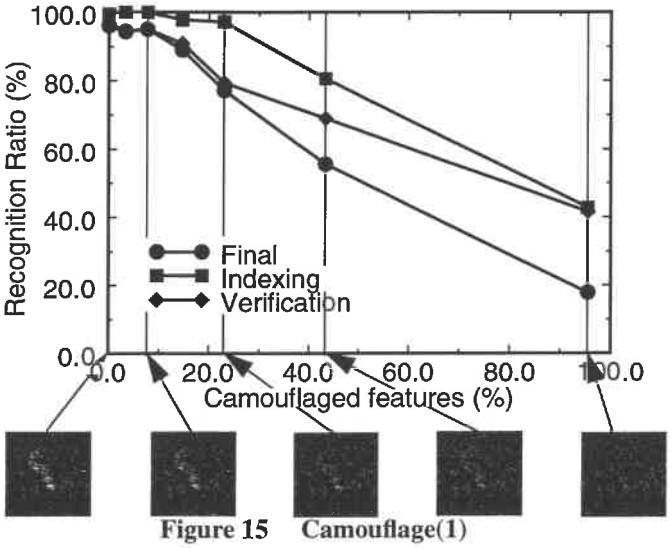
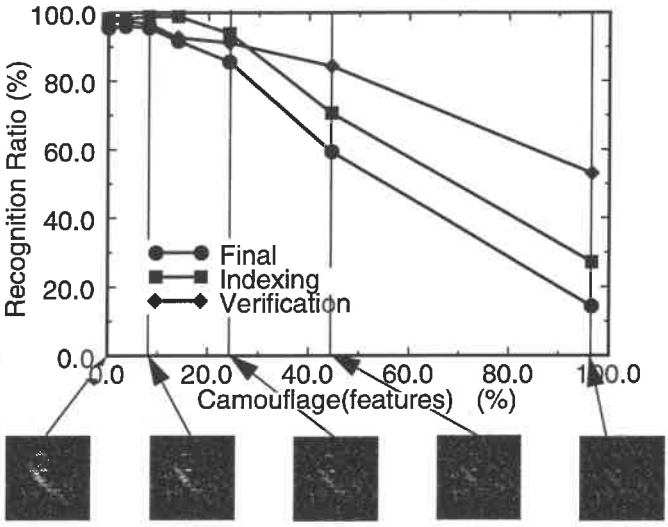


Figure 15 Camouflage(1)

M60



F15AR

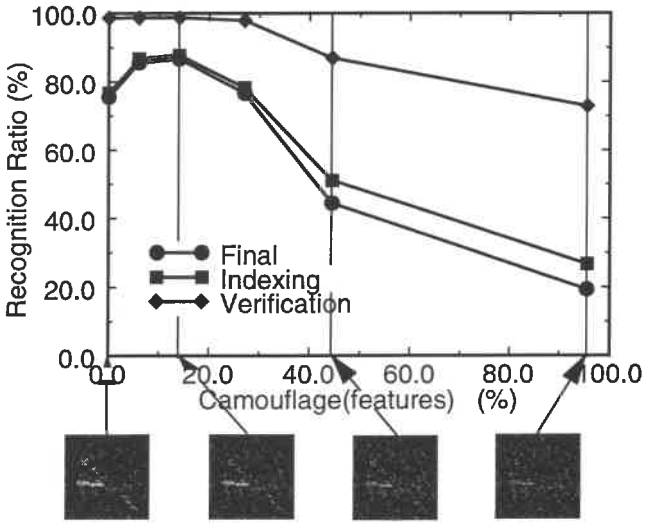


Figure 16 Camouflage(2)

7.4. 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 dictionary containing BMP, BTR60 and KTANK. In Table 12, each row and column represents the input image and system response, respectively. The number indicates the classification ratio by the system for each vehicle, while the numbers in parentheses presents 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.

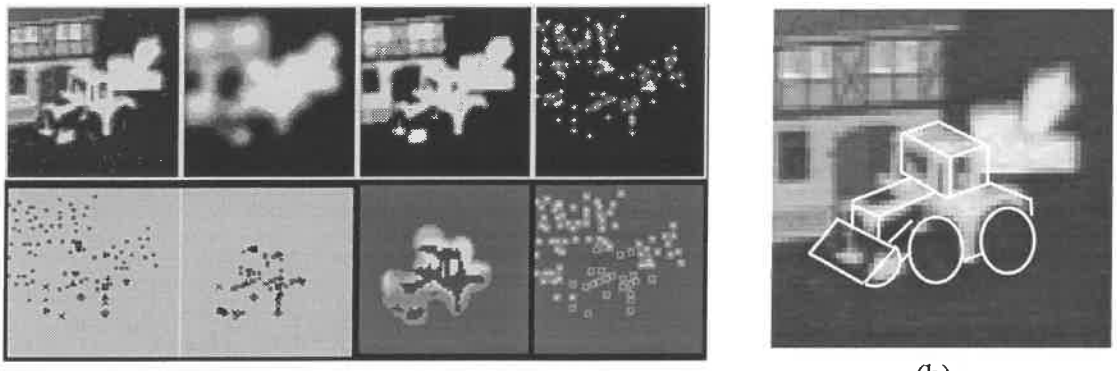
Table 12:

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%)

7.5. Optical Image Recognition

Our recognition system can be applied to other sensors by simply replacing the sensor simulator. We have applied our recognition system to real specular images given by a specular sensor, a CCD camera with a point light source mounted on it. In this example, the dictionary is obtained directly from a sequence of real images instead of simulated image: an object is rotated using an rotation table, and a sequence of images are captured.

The top row in Figure 17 (a) depicts input image and three potential fields. The second row shows the point correspondences and the final position of the deformable template. Figure 17 (b) shows the recognition result superimposed on the original image.



(a) (b)

Figure 17 Specular sensor image recognition

8. Conclusion

This paper introduces the use of invariant histograms and deformable templates for SAR recognition. An invariant histogram is a histogram of geometric invariants given by many primitive feature sets. Deformable template matching precisely examines the existence of an object by superimposing and aligning templates over potential energy field given by image features. The deformation allows for precise localization of non-attached features typical of SAR data.

We have develop a SAR recognition system using these two techniques, and 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 dictionary for indexing and deformable templates for verification. Currently, it takes a half hour for this compilation on 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, from the potential fields from an 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 from the input image and compute potential fields.

Recently, several researchers have begun to develop appearance-based recognition systems. From a large number of images, they effectively extract compress essential features, eigen-values in an orthogonal eigen space, and use those eigen-values for object recognition. Turk and Pentland [32] recognized human faces using eigen-vectors and Murase and Nayar [33] applied an eigen-space analysis for illumination planning. The main focus of these techniques are how to effectively reduce the size of necessary features for recognition.

In contrast to these compression-oriented approaches, this paper proposes a redundancy oriented approach; by using a redundant representation of image features, this work shows it is possible to build a robust recognition system, in particular for SAR recognition. These characteristics are particularly important when handling occluded target images consisting of unstable non-attached features typical of SAR data.

9. Reference

[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] Waxman, 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, K. 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] Huttenlocher, 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. IUW-93*, 819-829

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

[30] Andersh, D, *XPATCH Manual*, Wright Research and Development Center, WPAFB, 1993

[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, 1994.