COMPUTER VISION THROUGH LEARNING

by

R. S. Michalski
A. Rosenfeld
Y. Aloimonos
Z. Duric
M. Maloof
Q. Zhang

Reports of the Machine Learning and Inference Laboratory , MLI 97-12, George Mason University, Fairfax, VA, 1997.

Computer Vision through Learning

R.S. Michalski[1,3], A. Rosenfeld[2], Y. Aloimonos[2],
Z. Duric[1,2], M. Maloof[1], and Q. Zhang[1]

[1]Machine Learning and Inference Laboratory, George Mason University
Fairfax, VA 22030-4444

[2]Computer Vision Laboratory, University of Maryland
College Park, MD 20742-3275

[3]Also with the Institute of Computer Science, Polish Academy of Sciences

Abstract

This report concerns problems of learning patterns in images and image sequences, and using the obtained patterns to interpret new images. The study describes our approach to these problems, the developed methodology, called MIST, and our results in the following problem areas: (i) semantic interpretation of color images of outdoor scenes, (ii) detection of blasting caps in x-ray images of airport luggage, (iii) recognizing actions in video image sequences, (iv) recognizing targets in SAR images, (v) "robotic estimation", (vi) visual memories, and (vii) designing "Bisight control library". We discuss the image formation processes in these problem areas, and the choices of representation spaces used in our approaches to solving these problems. The results presented indicate the advantages and significant potential in applying machine learning to computer vision problems.

Contents

1 Introduction

The underlying motivation of this research is that vision systems need learning capabilities for handling problems for which algorithmic solutions are unknown or difficult to obtain. Learning capabilities can also make vision systems more easily adaptable to different vision problems, and more flexible and robust in handling variable perceptual conditions [MRA94].

Much of the current research on learning in vision systems has concentrated on neural network applications — for example, road navigation [Pom91], object detection, and object recognition in various types of images (visible, SAR, etc.) [FeB96, RBP96, RBK96]. Advantages of neural ne based methods include their generality and their ability to learn continuous transformations. Disadvantages include the difficulty of incorporating prior knowledge (especially relational knowledge), the difficulty of learning structural descriptions, slow learning rates, and lack of comprehensibility of the learned knowledge [MRA94].

While symbolic learning methods suffer much less from these problems, they have been applied mostly in areas other than computer vision. In computer vision, they may be particularly useful for discovering key object attributes, generating new high level attributes based on the original low level attributes, learning visual surface descriptions involving texture and color, learning complex shape descriptions, acquiring structural or relational models of objects, constructing and updating model databases, segmenting scenes to conceptual units, learning the "context" in which an algorithm can be successfully applied, and related problems [GrP96, MDMR96, MRADMZ96, StF95]. Applications of symbolic approaches to vision problems remain an insufficiently explored but potentially fruitful domain of research.

A significant importance for vision have multistrategy learning systems, that combine different representations and/or different learning algorithms. One particular multistrategy system combines neural network and symbolic learning. This method induces rules which are then used to structure a neural network architecture. A secondary learning step refines the network's weights. This method provides generality and very fast recognition rates [BMP94, MZMB96]. One can also use neural networks for lower-level vision processes and symbolic methods for higher-level visual processes. These methods are potentially very powerful and constitute promising research directions.

We have been studying the application of symbolic, neural net and multistrategy learning methods to such problems as interpreting outdoor scenes, recognizing objects in cluttered environments, recognizing actions in video image sequences, and detecting and recognizing targets in SAR images. The following sections summarize specific results obtained on a project on "Computer vision through learning" that was conducted jointly by George Mason University and the University of Maryland [MRADMZ96].

In Section 2 we review previous work on machine learning in computer vision. In Section 3 we address the problem of conceptually segmenting color images of outdoor scenes. For this purpose we use the Multi-level Image Sampling and Transformation (MIST) methodology; a detailed description of this methodology can be found in [MZMB96]. In Section 4 we address the problem of detecting blasting caps in x-ray images of luggage; the details can be found

in [MaM96, MDMR96]. In Section 5 we address the problem of recognizing a function of an object from its motion; the technical details can be found in [DFR96, DRR96]. In Section 6 we address the problem of recognizing targets in SAR images. In Section 7 we address the problem of a robot that needs to learn its environment. In Section 8 we address the problem of visual memories. Finally, in Section 9 we describe the "Bisight control library".

2 Previous Work on Machine Learning in Computer Vision

Michalski [Mic72, Mic73] examined how symbolic AQ rule learning could be used for discrimination between textures or between simple structures. These seminal papers presented the Multi-Level Logical Template (MLT) methodology in which windowing operators scanned an image and extracted local features. These features were used to learn rules describing textures (or simple structures); the rules were then used for texture (or simple structure) recognition.

Shepherd [She83], encoding examples as feature vectors, learned decision trees for an industrial inspection task — specifically, classification of the shapes of chocolates. Comparisons of classification accuracy were made between decision tree, k-nearest neighbor (k-nn), and minimum distance classifiers. Experimental results for these classifiers were similar, with the minimum distance classifier producing the highest accuracy, 82%.

Channic [Cha89] extended the MLT methodology [Mic72, Mic73] by using convolution operators in conjunction with the original set of windowing operators for feature extraction. Using the AQ learning system, Channic investigated incremental learning and iterative learning from sequences of images using ultrasound images of laminated objects.

Instead of representing examples using feature vectors, Connell and Brady [CoB87] learned generalized semantic networks from images of classes of hammers and of overhead views of commercial aircraft. Training examples were generated by a vision system that took gray scale images as input and produced semantic networks for the objects. A learning system, which was a modified version of Winston's [Win84] ANALOGY program, learned by generalizing the training examples. The learning system was extended to learn disjunctive concepts and to learn from only positive examples. These generalized representations were used to classify unknown objects.

Cromwell and Kak [CrK91] proceeded as Shepherd did, using feature vectors to characterize shapes. Electrical component shapes were learned using a symbolic induction methodology based on that developed by Michalski [Mic80]. They reported that their method achieved 72% on testing data, but no comparisons were made to other learning methods.

Pachowicz and Bala [PaB91] also used the MLT methodology, following Michalski [Mic72, Mic73] and Channic [Cha89], but added a modified set of Laws' masks for texture feature extraction. They also applied techniques for handling noise in symbolic data. These tech-

niques included optimizing learned symbolic descriptions by weak rule truncation, which were developed for symbolic data by Michalski et al. [MMHL86], as well as removing training examples covered by weak rules and re-learning. The PRAX method for learning a large number of classes was introduced by Bala, Michalski, and Wnek [BMW92, BMW93].

Segen [Seg94] used a hybrid shape representation consisting of a hierarchical graph that takes into account local features of high curvature, and the angles and distances between these local features. This representation is invariant to both planar rotation and translation. Shapes were silhouettes of hand gestures. Segen's system runs in real time and has been applied to airplane simulator control as well as to control of a graphics editor program. Error rates were between 5% and 10%, but most errors were unknowns rather than misclassifications.

Cho and Dunn [ChD94] described a new learning algorithm for learning shape. This algorithm memorizes property lists and updates associated weights as training proceeds. Forgetting mechanisms remove useless property lists. Shapes are modeled by a series of line segments. Using the orientations of these segments, local spatial measures are computed and form a property list for a shape. The system was used to classify tools and hand gestures and achieved predictive accuracies of 92% and 96% on these problems.

Dutta and Bhanu [DuB94] presented a 3D CAD-based recognition system in which genetic algorithms are used to optimize segmentation parameters. Qualitative experimental results were presented for indoor and outdoor motion sequences in which the system recognized images of wedges (traffic cones) and cans from gray scale and depth map images.

Sung and Poggio [SuP94] worked on automatic human face detection. An example-based learning approach was tested for locating unoccluded frontal views of human faces in complex scenes. The space of human faces was represented by a few "face" and "non-face" pattern prototypes. At each image location, a two-valued distance measure was computed between the local image pattern and each prototype. A trained classifier was used to determine whether a human face is present. The authors showed that their distance metric is critical for the success of their system.

Zheng and Bhanu [ZhB96] examined how Hebbian learning mechanisms could be used to improve the performance of an image thresholding algorithm for automatic target detection and recognition. Qualitative results were presented in which the adaptive thresholding algorithm was shown to be superior to the classical thresholding algorithm for both SAR and FLIR images.

Rowley et al. [RBK96] built a neural network-based face detection system by using a retinally connected neural network to examine small windows of an image and decide on the existence of a face. A bootstrap algorithm was implemented during training so as to add false detection into the training set and as a consequence, eliminate the difficult task of manually selecting non-face training examples. Experimental results showed better performance in terms of detection and false-positive rates.

Romano et al. [RBP96] built a real-time system for face verification. Experiments showed

that simple correlation strategies on template-based models are sufficient for many applications in which the identity of a face in a novel image must be verified quickly and reliably from a single reference image. The authors suggested that this automatic real-time face verification technique could be put to use in such human-machine interface applications as automated security systems. The technique has been integrated into a screen locking application which permits access to workstations by performing face verification in lieu of password authentication.

The MLT methodology [Mic72, Mic73] has recently been extended into the Multi-Level Image Sampling and Transformation (MIST) methodology. MIST has been applied to a variety of problems including natural scene segmentation [MZMB96] and identification of blasting caps in x-ray images [MDMR96]. For classifying natural scenes, three learning techniques were compared: AQ15c [WKBM95], a backpropagation neural network [Zur92], and AQ-NN [BMP94].

AQ-NN is a multistrategy learning technique in that it uses two different representations and two different learning strategies. Specifically, the AQ learning algorithm is used to learn attributional decision rules from training examples. These decision rules are then used to structure a neural network architecture. A backpropagation algorithm is then used as a learning step to further optimize the AQ induced descriptions. In such a system, learning times and recognition rates are often significantly decreased, while predictive accuracy is improved, with respect to conventional neural network learning. To learn classes such as ground, grass, trees and sky, hue, intensity, and convolution operators are used to extract features from a user-designated training area. These examples are then presented to the learning system, which induces a class description. AQ15c, used alone, achieved a predictive accuracy of 94%, while AQ-NN and a standard neural network achieved predictive accuracies near 100%. The training time of AQ-NN was approximately two orders of magnitude shorter than the training time of the standard NN.

3 Semantic Interpretation of Color Images of Outdoor Scenes

The MIST methodology (Multi-level Image Sampling and Transformation) provides an environment for applying diverse machine learning methods to problems of computer vision. The methodology is illustrated here in connection with a problem of learning how to semantically interpret natural scenes. In the experiments described here, three learning programs were used: AQ15c for learning decision rules from examples; NN, neural network learning; and AQ-NN, multistrategy learning combining symbolic and neural network methods.

The results presented below illustrate the performance of the learning programs for the chosen problem of natural scene interpretation in terms of predictive accuracy, training time, recognition time, and complexity of the induced descriptions. The MIST methodology has proven to be very useful for this application. Overall, the experiments indicate that the

multistrategy learning program AQ-NN appears to be the most promising approach.

This section briefly describes the MIST methodology and illustrates it by an application to natural scene interpretation. As pointed out in [FiS88, StF91], the semantic interpretion of natural scenes and recognition of natural objects is one of the most challenging open vision problems. The MIST methodology offers a new approach to these problems.

3.1 The MIST Methodology

The MIST methodology works in two basic modes: *Training mode* and *Interpretation mode.* In Training mode, the system builds or updates the Image Knowledge Base (IKB), which contains class descriptions and the background knowledge relevant to image interpretation. A description (or model) of a visual category is developed by inductive inference from examples specified by a trainer. Class descriptions are arranged into procedures defining sequences of image transformation operators.

In Interpretation mode, a learned (or predefined) image transformation procedure is applied to a given image to produce an Annotated Symbolic Image (ASI). In an ASI, areas that correspond to the locations of recognized classes in the original image are marked by symbols (e.g., colors) denoting these classes, and linked to annotations (text containing additional information about the classes, such as degree of certainty of recognition, properties of the class, relations to other classes, etc.). (Although developed independently, MIST's concept of an ASI is similar to the concept of a class map in the ALISA system [HoB94].) The following paragraphs describe the two modes in greater detail.

Training Mode

This mode (see Figure 1) is executed in four phases: **LP1**—description space generation and background knowledge formulation; **LP2**—event generation; **LP3**—learning or refinement; and **LP4**—image interpretation and evaluation. These four phases can be repeated iteratively, creating images at different levels.

LP1: *Description space generation and background knowledge formulation*

A trainer assigns class names to areas in the image(s) that contain objects to be learned. These areas are divided into training and testing areas. Objects to be learned are presented in different poses and with different appearances (by changing perceptual conditions) so the system can learn a description that is invariant to class-preserving transformations. The trainer also defines the initial description space, i.e., initial attributes and/or terms to be measured on image samples, and specifies their value sets (measurement scale) and their types. This phase also involves an optimization of the image volume, that is, a reduction of the image resolution and intensity levels (hue and saturation, in color images) according to the needs of the given problem. The trainer may also define constraints on the description space, initial recognition rules, and possibly forms for expressing the descriptions (e.g.,

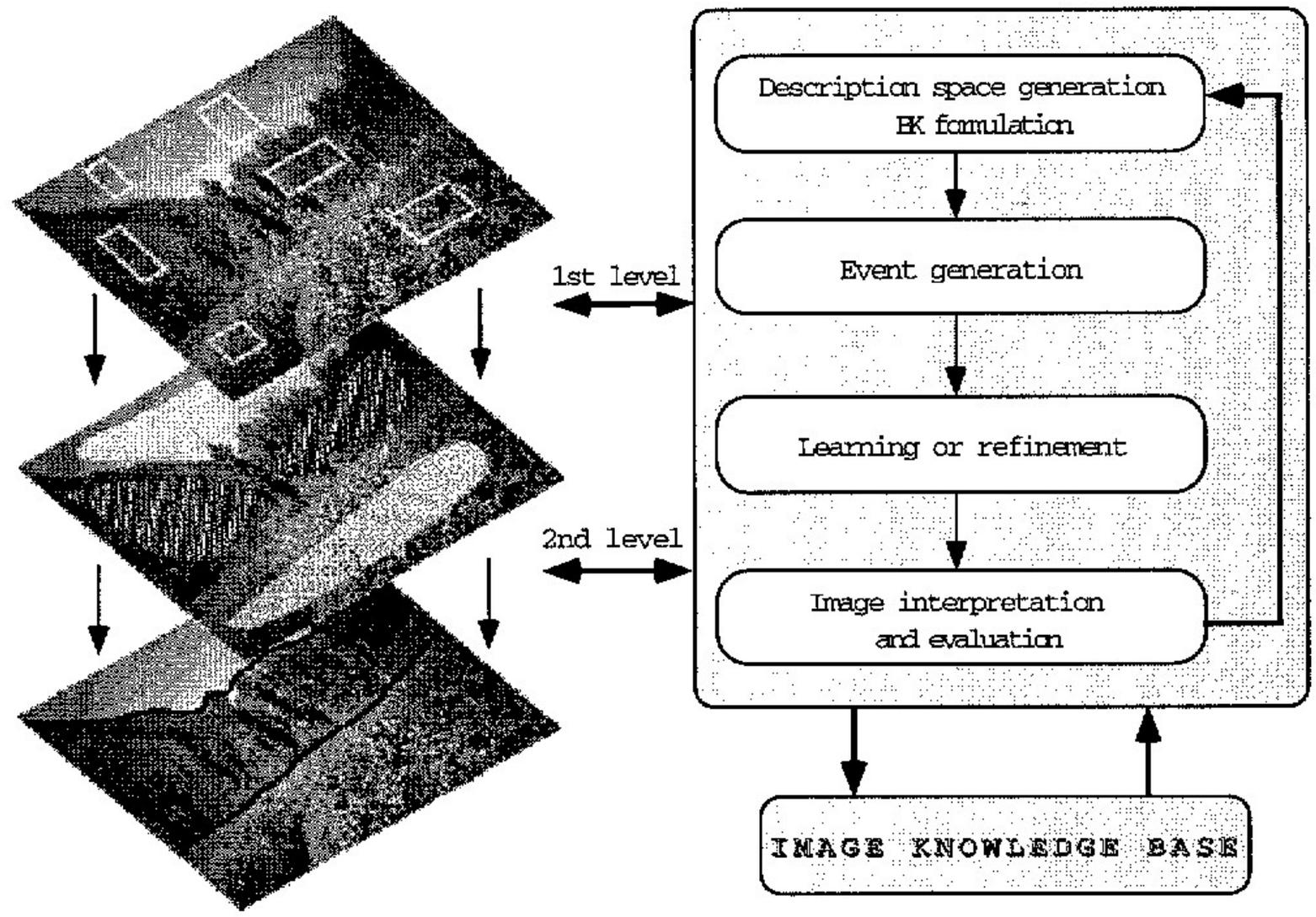


Figure 1: The MIST training mode.

conjunctive rules, DNF, the structure of the neural network, etc.). Procedures for the measurement of attributes/terms are selected from a predefined collection.

LP2: *Event generation*

Using the chosen procedures, the system generates initial training examples ("training events") from each area. The areas are sampled exhaustively or selectively.

LP3: *Learning or refinement*

The system applies a selected machine learning program to the training examples to generate a class description. Currently, we have the following programs available: AQ15c for learning general symbolic rules from examples; NN, neural network learning with back-propagation; and AQ-NN, a system that integrates AQ rule learning with neural network learning.

LP4: *Image interpretation and evaluation*

The developed descriptions are applied to the testing areas to generate an *Annotated Symbolic Image* (ASI). In an ASI, the areas corresponding to given classes are marked by symbols representing these classes (numbers, colors, etc.). These areas are also linked to text that includes additional information about the class descriptions. The quality of the generated descriptions is determined by comparing the ASI with testing areas in the original image. Depending on the results, the system may stop, or may execute a new learning process (iteration), in which the ASI is the input (hence the term "multilevel" in the name of the methodology). If the generated descriptions need no further improvement, the process

is terminated. This occurs when the obtained symbolic image is "sufficiently close" to the target image labeling (indicating the "correct" labeling of the image). Complete object descriptions are sequences of image transformations (defined by descriptions obtained at each iteration) that produce the final ASI. Learning errors are computed by comparing the target labeling (made by the trainer) with the learned labeling (produced by the system).

Interpretation Mode

In this mode, the system applies descriptions from the Image Knowledge Base to semantically interpret a new image. To do so, the system executes a sequence of operators (defined by the description) that transform the given image into an ASI. A given "pixel" in the ASI is assigned a class on the basis of applying operators to a single event, or to a sample of events, and applying majority voting (typically within a 3×3 window). In ASI, different classes are denoted by different colors and/or textures. The simplest form of annotation used in the ASI is to associate degrees of confidence with the ASI pixels denoting a given class.

3.2 Implementation and Experimental Results

The current MIST methodology has been implemented with the following learning systems:

- Symbolic rule learning program AQ15c [MMHL86, WKBM95].
- Multistrategy learning system AQ-NN combining decision rule learning with neural network learning [BMP94].
- Multistrategy learning system AQ-GA that combines decision rule learning with a genetic algorithm [MBP93].
- Class similarity-based learning for building descriptions of large numbers of classes (PRAX) [BMW92, BMW93].

An earlier version of MIST has been applied to learning descriptions of classes of surfaces [MBP93]. The core part of the descriptions was in the form of decision rules, which were determined by the inductive learning program AQ15 [MMHL86] and represented in the VL_1 logic-style language (Variable-Valued Logic System 1) [Mic73]. Such decision rules can be applied to an image in parallel or sequentially.

A simple version of the MIST methodology was applied to problems of semantically interpreting outdoor scenes using several learning methods. In the experiments, we used a collection of images representing selected mountain scenes around Aspen, Colorado (see Figure 2).

The input to the learning process was a training image in which selected examples of the visual classes to be learned had been labeled by a trainer — for example, trees, sky, ground, road, and grass. We experimented with different sets of attributes defining the description space, images obtained under different perceptual conditions, different sizes of training areas,

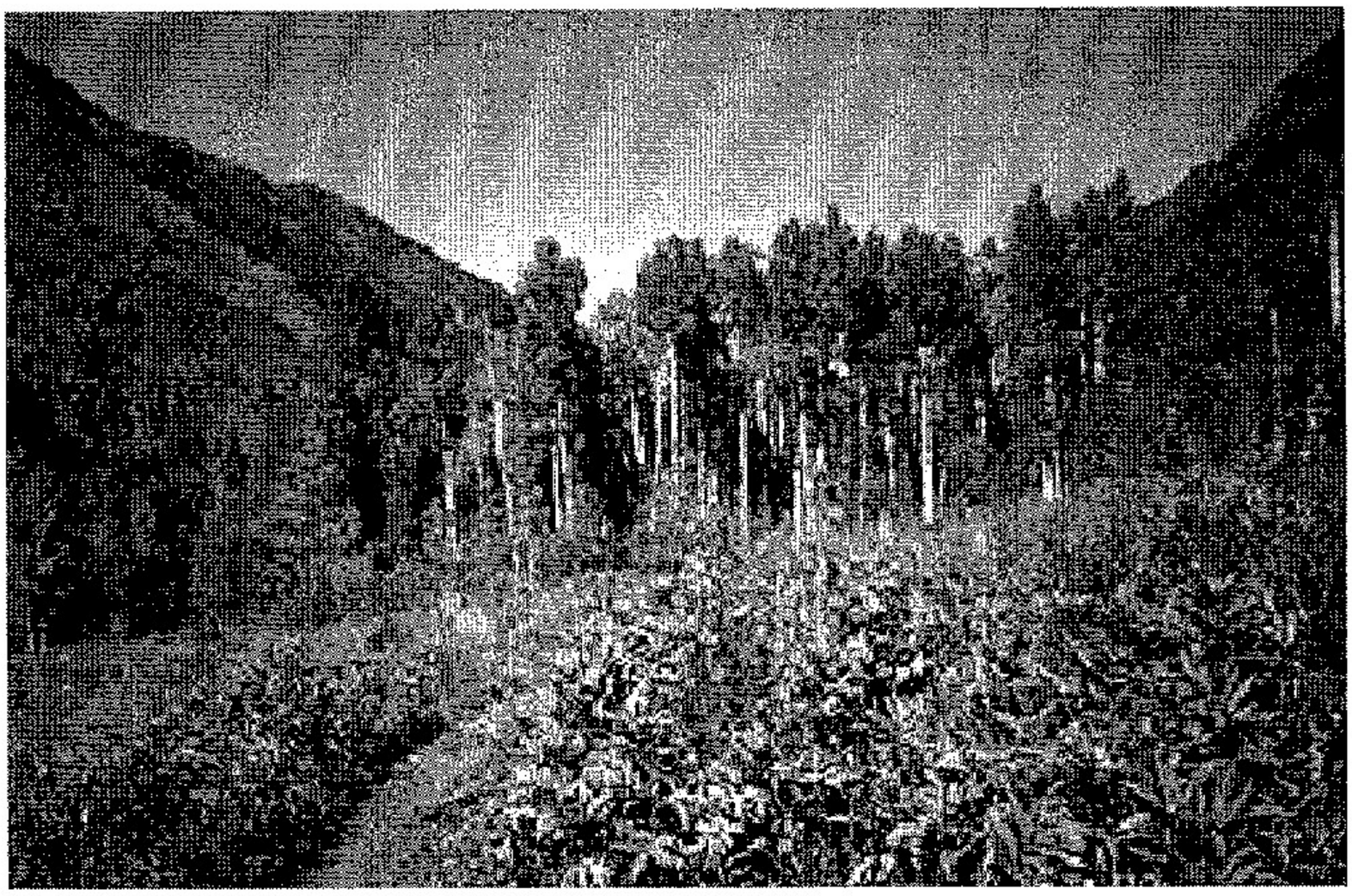


Figure 2: A typical image of a natural scene used in experiments.

and different sources of training and testing image samples (from different parts of the same image area, from different areas of the same image, from different images).

In the experiments described here, the description space was defined by such attributes as hue, saturation, intensity, horizontal and vertical gradients, high frequency spots, horizontal and vertical V-shapes, and Laplacian operators. These attributes were computed for the 5×5 windowing operator (sample size) that scanned the training area. Vectors of attribute values constituted training events. Three learning methods were used: AQ15c, AQ-NN, and NN. Three different sizes of training areas were used: 10×10, 20×20, and 40×40 pixels. The validation methodology used here was a hold-out method in which a random selection of 60% of the samples from the training area were used for training, while the remaining 40% were used for testing [WeK92].

Table 1 gives results from an experiment involving only one level of image transformation using different learning programs. In this experiment, the training area for each class was only 10×10 pixels. When the training area was enlarged to 20×20, the training time was significantly longer, but the correctness of the interpretation of the areas of the image was approximately the same.

Figure 3 presents an example of a training image and an ASI obtained by applying the learned one-level descriptions to the whole image using a majority voting evaluation scheme. As can be seen from Figure 3b, most of the areas in the image were correctly interpreted, although the system learned class descriptions from relatively small training areas (Figure 3a). In this experiment, AQ-NN produced a slightly smaller neural network and the interpretation time was about 50% shorter than with the NN method.

Learning method	Training time	Recognition time	Accuracy
AQ15c	0.43s	1.000s	94.00%
AQ-NN	10.93s	0.016s	99.98%
NN	4.38s	0.033s	99.97%

Table 1: A summary of results from learning to interpret the image in Figure 3a. Data computed for 161 training events and 150 testing events selected from the 10×10 training area.

We also tested the data-driven constructive induction method (AQ17-DCI) in this experiment; this resulted in some new relevant attributes. Recognition accuracy was comparable to previous results [BWMK93].

4 Detection of Blasting Caps in X-ray Images of Luggage

This section presents work on an approach to the problem of recognizing blasting caps in x-ray images. This problem is an instance of a class of problems in which a vision system must inspect a sequence of images for known objects. Unfortunately, the fact that the objects are known is often of little or no help. If there is little standardization of the class of known objects, it becomes impractical to attempt to model the objects geometrically. What often constrains a class of objects is functionality [FrN71, StB91a, RDR95]. Learning can be useful for acquiring the relationship between image characteristics and object functionality [WCHBS95].

Our primary focus is on investigating how vision and learning can be combined to find blasting caps, as well as objects that could occlude blasting caps, in x-ray images. In a previous study [MaM94, MaM96], learning was used to acquire descriptions of blasting caps. Simple segmentation techniques were used to isolate objects from their background; they were then represented using intensity and geometric features.

In the work presented here, an analysis of functional properties of blasting caps was conducted to design the representation space for learning, which combines intensity and shape features. Experimental results demonstrate the ability of the inductive learning system to acquire the relationship between image characteristics and object functionality.

This research provides an opportunity to study the interplay between vision and learning processes [MRA94], especially as it relates to learning object functionality. A vision system capable of reliably recognizing blasting caps or objects that could occlude blasting caps could be used to aid airport security personnel in luggage screening.

(a) (b)

Figure 3: An example of the image interpretation process based on the rules learned from the indicated training areas. (a) An image with training areas for for sky, tree, and ground. (b) ASI obtained using a majority voting scheme.

4.1 Preliminaries

In this section we review the image formation process and imaging model in x-ray images.

A typical x-ray imaging system consists of an x-ray tube (photon source), an anti-scatter device, and a receptor (photon detector) [Dan88]. The photons emitted by the x-ray tube enter the objects, where they may be scattered, absorbed or transmitted without interaction. The primary photons recorded by the image receptor form the image, but the scattered photons create a background signal (i.e., noise) that degrades contrast. In most cases, the majority of the scattered photons can be removed by placing an anti-scatter device between the objects and the image receptor.

What follows is a simple mathematical model of the imaging process. We start by considering a monochromatic x-ray source that emits photons of energy E and is sufficiently far from the objects (luggage) being inspected that the photon beam can be considered to be parallel (see Figure 4). The incident photon beam is parallel to the z direction and the image is recorded in the xy plane. We assume that each photon interacting with the receptor

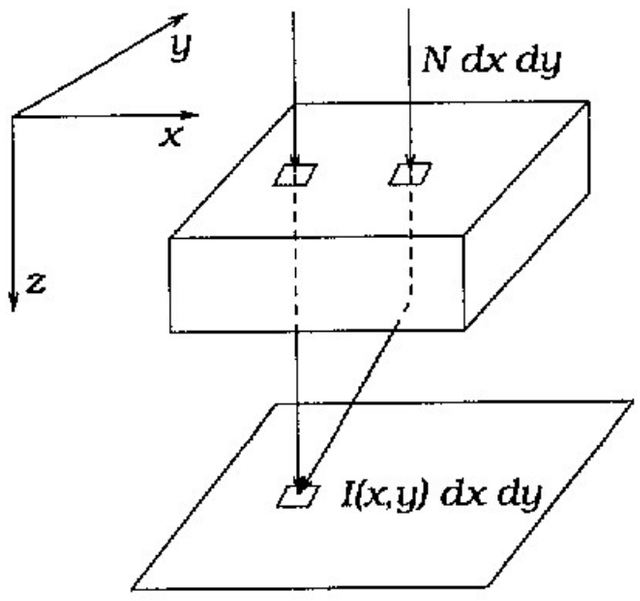


Figure 4: The geometry of x-ray imaging [Dan88].

is locally absorbed and that the response of the receptor is linear, so that the image may be considered as a distribution of absorbed energy. If N photons per unit area are incident on the object and $I(x, y)\,dx\,dy$ is the energy absorbed in area $dx\,dy$ of the detector, then

$$I(x, y) = \exp\left(-\int \mu(x, y, z)dz\right) \cdot N\,\varepsilon(E, 0)\,E\,(1 + R)$$

where the line integral is over all materials along the path of the primary photons reaching the point (x, y), $\mu(x, y, z)$ is the linear attenuation coefficient, $\varepsilon(E, 0)$ is the energy absorption efficiency of the receptor for the photon energy level E at an incident angle of 0, and R is the ratio between the scattered and primary radiation (which is usually very small).

We assume orthographic image projection (see Figure 4). The image of the object point (X, Y, Z) is the point (x, y) such that

$$x = s\,X, \quad y = s\,Y,$$

where s is a constant. The image intensity at the pixel (x, y) is obtained by integrating $I(x, y)$ over the area of the pixel in the image receptor.

The intensity in an x-ray image is proportional to the number of x-ray photons that pass through objects on their way from the source to the receptor. Since different materials have different transparency properties, the intensity of an x-ray image depends on both the thickness and the type of material between the source and the receptor. Moreover, any x-ray photon that is not absorbed by one object on its path may be absorbed by another. Thus, a thick layer of semi-transparent material can have the same effect on the image receptor as a thin layer of opaque material.

4.2 Problem Statement

Although blasting caps are manufactured objects, there is enough variability in their manufacture to make a CAD-based recognition system impractical. What is common to all blasting caps, however, is their functionality. Ultimately, blasting caps are defined by their functional properties, not by their shapes.

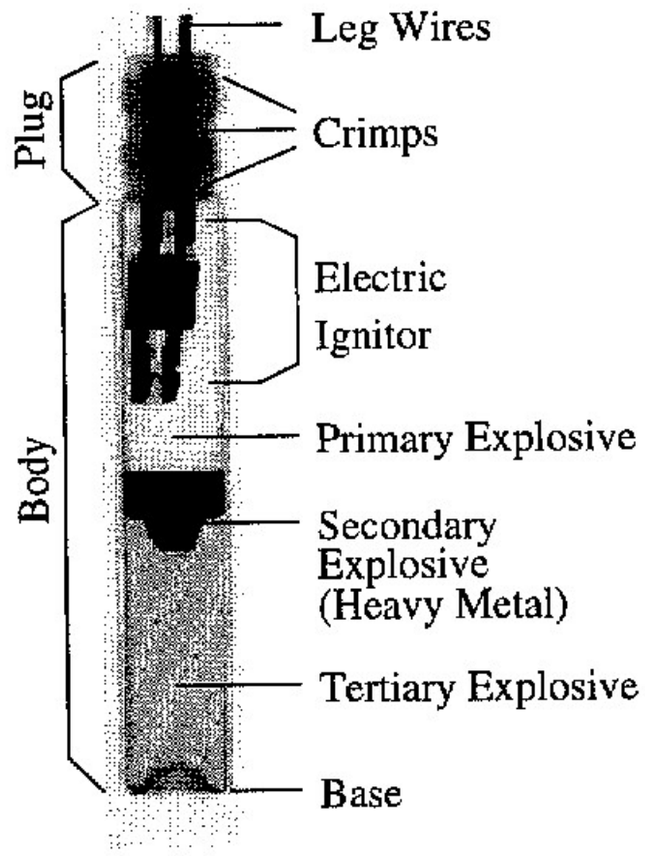


Figure 5: Detailed x-ray of a blasting cap.

A typical blasting cap (see Figure 5) consists of a cylindrical metal shell filled primarily with the explosive. In its approximate middle, there is a small globule of heavy metal secondary explosive. Finally, leg wires from the electric ignitor extend from one of the ends. The most dense (opaque to x-rays) part of a blasting cap is the concentration of the heavy metal explosive, which is approximately centrally symmetric. The leg wires also produce dense features, but are very thin. Finally, the copper or aluminum tube filled with explosive, which is axially symmetric, is typically more dense than the surrounding areas of the luggage.

To understand images of blasting caps, we begin by considering a generic blasting cap that is not occluded by opaque material. Let l be the length of an approximately cylindrical blasting cap, r be its radius, and σ be the angle between the axis of the cap and the image receptor. Consider the length of the path p of an x-ray photon as it passes through the blasting cap. When $\sigma = 0$, p ranges from $2r$ at the axis to 0 at the occluding contour. In general, p is multiplied by $\sec\sigma$; however, p cannot be longer than l. From the equation of $I(x, y)$, we see that the number of photons passing through the blasting cap decreases exponentially as p grows. From the image projection equations, we see that the image of a blasting cap is rectangularly shaped; its width is approximately $2rs$, and its length is approximately $ls\sec\sigma$. Its intensity is lowest along its axis, and highest along its occluding contour, which produces a low-contrast boundary. Also, the image of the heavy metal secondary explosive (see Figure 5) appears as a small, approximately symmetric blob on the axis of the blasting cap. The center of the blob is nearly opaque and thus its intensity is near zero. The boundary of the blob is lighter, but still has a very low intensity. The leg wires are strong features, but are not clearly visible in the images. (In our examples, the image resolution is 565×340 and the leg wires are barely visible. Currently, we are attempting to obtain images of higher resolution so that the leg wires can be more easily detected.)

Therefore, the strongest feature of a blasting cap is the low-intensity blob in the center of a rectangular ribbon of higher intensity. The intensities of both the blob and the ribbon are

Average Predictive Accuracy (%)		
Overall	Correct	83.51±1.3
	Incorrect	16.49±1.3
Blasting Cap	Correct	85.82±2.1
	Incorrect	14.18±2.1
Non-Blasting Cap	Correct	81.19±2.4
	Incorrect	18.81±2.4

Table 2: Summary of quantitative experimental results.

lowest along the axis of the blasting cap and highest along the occluding contour. Finally, if a blasting cap is occluded by any object, its image will be darker than the image of a blasting cap that is not occluded.

4.3 The Method and Experimental Results

We present a two-phase, bottom-up and top-down learning approach to recognizing blasting caps in x-ray images. In the first phase, low intensity blobs, which serve as attention-catching devices, are used to generate object hypotheses. These blobs correspond to the secondary high explosive, which is typically a heavy metal compound, located near the middle of the blasting cap (see Figure 5).

In the second phase, each generated hypothesis spawns a process that attempts to fit a local model to ribbon-like features surrounding the blob. These features correspond to the metal body of the blasting cap (see Figure 5). The local model is acquired using the inductive learning system AQ15c and captures intensity and geometric features of both the low intensity blob and the surrounding ribbons. A flexible matching routine is used to match the local model to the image characteristics; this not only produces an object identification, but also yields a confidence in the identification.

The x-ray images used for experimentation were of luggage containing blasting caps appearing in varying orientations and under varying amounts of clutter, which included clothes, shoes, calculators, pens, batteries, and the like. The luggage was imaged much as it would be in an airport scenario: flat in relation to the x-ray source, but rotated in the image plane. Five images were selected from a set of 30 which were of low to moderate complexity in terms of clutter and positional variability of the blasting cap. Figure 6 shows one of the images used for experimentation.

Regions of interest were interactively determined, and contained low intensity blobs and ribbons corresponding to positive and negative examples of blasting caps. From each of the 64 selected regions, 27 geometric (e.g., compactness and proximity measures) and intensity-based (e.g., minimum, maximum, and average) features were computed, resulting in 28 blasting cap and 38 non-blasting cap objects. The AQ15c [WKBM95] inductive learning

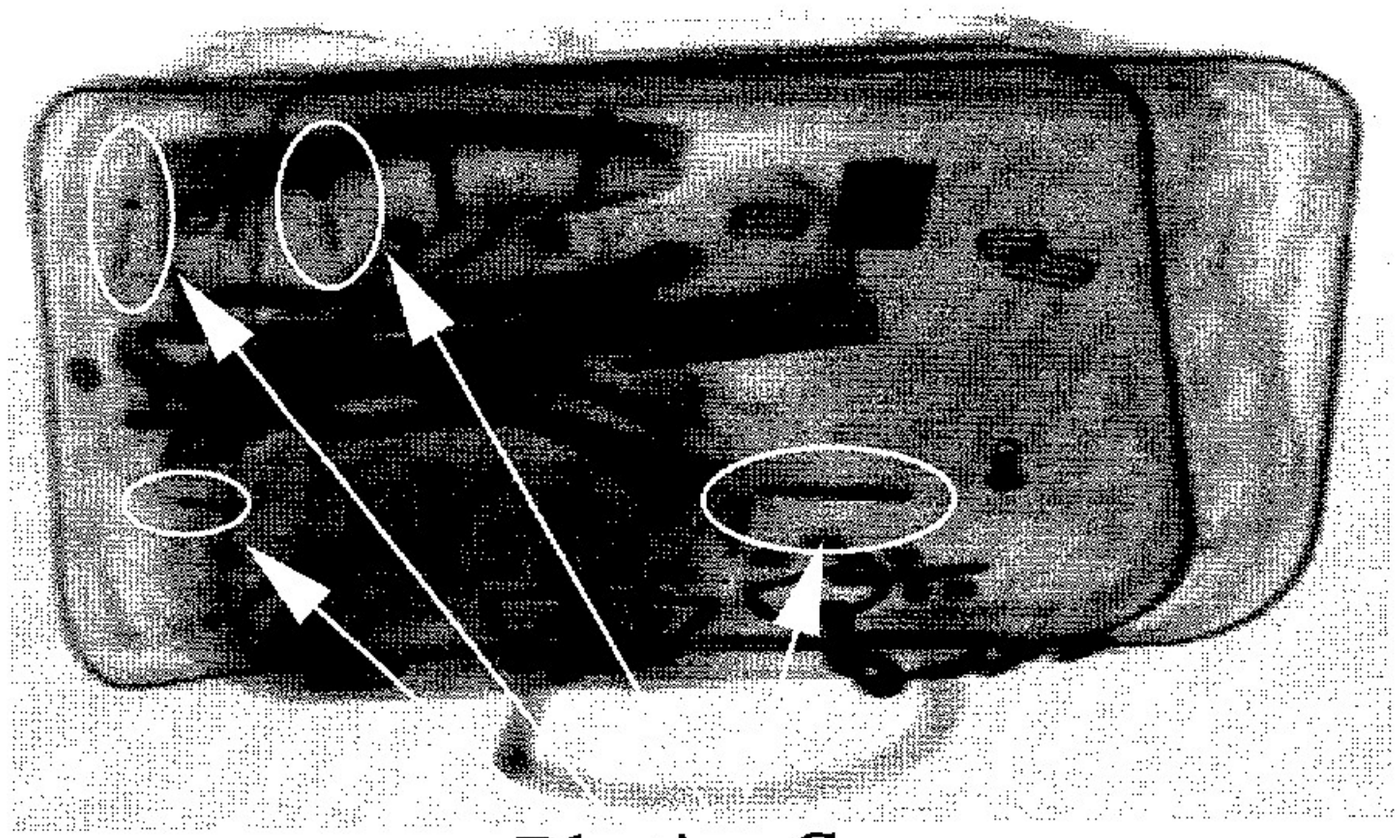


Figure 6: Sample image used for experimentation.

system was used to learn descriptions of blasting caps and non-blasting caps.

Induced descriptions from AQ15c were validated using 100 iterations of two-fold cross-validation. This validation method involves 100 learning and recognition runs. For each run, the extracted image data was randomly partitioned into a training set and a testing set. After learning from examples in the training set, the induced class definitions were tested using examples in the testing set. We can compute the predictive accuracy for each run based on the correct or incorrect classification of the examples in the testing set. The overall predictive accuracy for the experiment is the average of the accuracies computed for each run. These results are summarized in Table 2 and show the average predictive accuracy with a 95% confidence interval for the overall experiment and for each class.

As a qualitative demonstration of the method, the learned class definitions were also applied to an unseen image. The learned class definitions from AQ15c using training data from four images were tested on objects extracted from a fifth, unseen image, which is shown in Figure 7. Objects 1–6 are blasting caps, objects 7–10 are not. Object 5, which is a blasting cap, was mis-classified. All other objects in this image were classified correctly.

5 Recognizing Actions in Video Image Sequences

Recognizing the functions of objects is often a prerequisite to interacting with them. The functionality of an object can be defined as the usability of the object for a particular purpose [BoB94].

There has been considerable recent research on the problem of recognizing object functionality; for a short survey see [BoB94]. Early work on functional recognition can be found

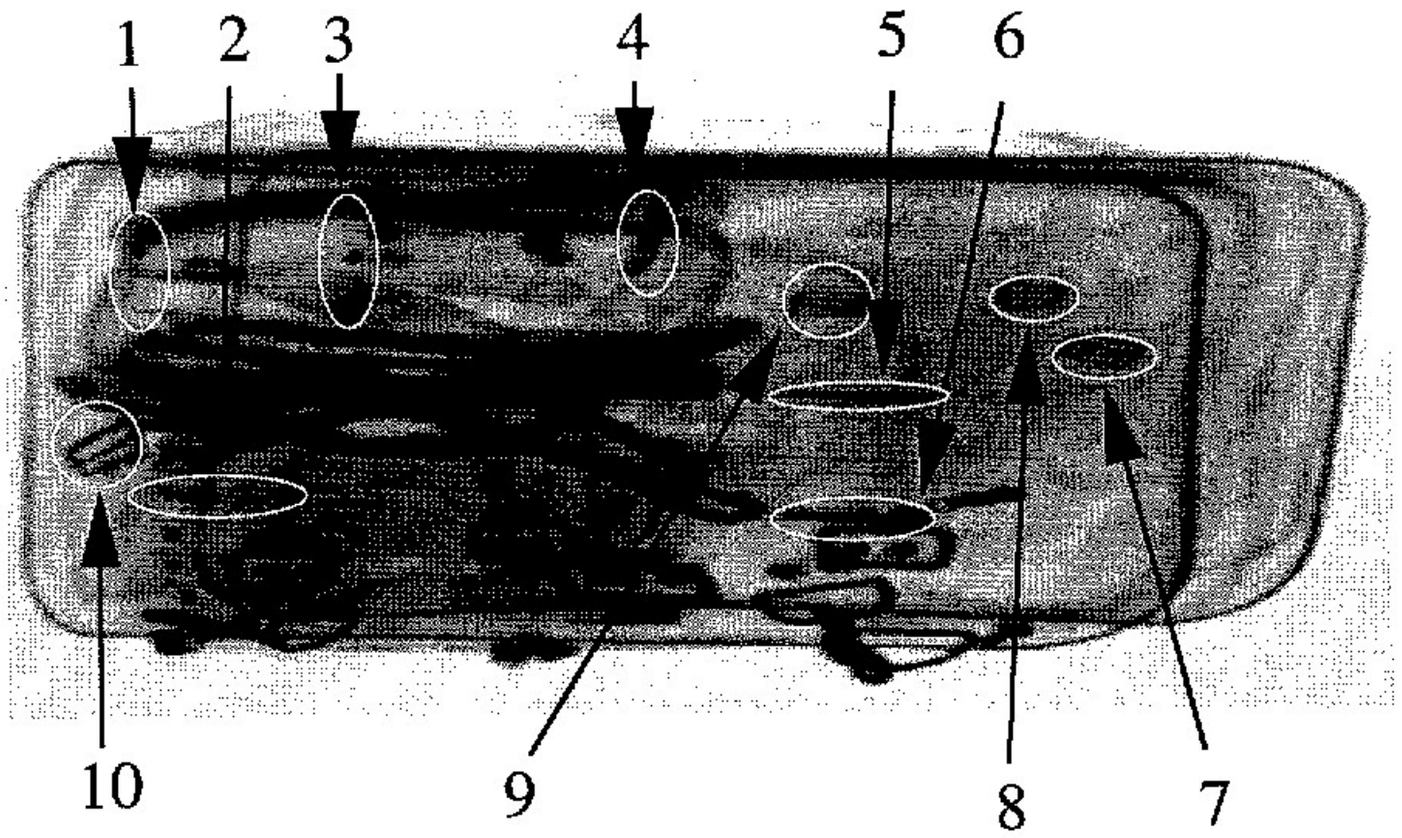


Figure 7: Test image for applying learned class definitions.

in [FrN71, SoB83, WBKL83]. The goal of this research has been to determine functional capabilities of an object based on characteristics such as shape, physics and causation. More recently, Stark and Bowyer [StB91a, StB91b, SHGB93] used this approach to solve some of the problems presented by more traditional model-based methods of object recognition. This work has dealt only with stationary objects; no motion is involved. In more recent work Green et al. [GESB94] discuss the recognition of articulated objects, using motion to determine whether the object possesses the appropriate functional properties. Little attention has been given to the problem of determining or learning the functionality of an object from its motion. In fact, however, motion provides a strong indication of function. In particular, velocity, acceleration, and force of impact resulting from motion strongly constrain possible function. As in other approaches to recognition of function, the object (and its motion) should not be evaluated in isolation, but in context. The context includes the nature of the agent making use of the object and the frame of reference used by the agent.

In this section, we address the following problem: How can we use the motion of an object, while it is being used to perform a task, to determine its function? Our method of answering this question is based on the extraction of a few motion descriptors from the image sequence. These descriptors are compared with stored descriptors that arise in known motion-to-function mappings to obtain function recognition.

Since many objects can display similar motion characteristics an object model is necessary to determine the functions of objects from their motion characteristics. Our work is based on segmenting the object into primitive parts and analyzing their motions.

5.1 Function from Motion

Primitive Shapes and Primitive Motions

Following [Bie85, RRP93, RDR95] we regard objects as composed of primitive parts. On the coarsest level we consider four types of primitive parts: sticks, strips, plates, and blobs, which differ in the values of their relative dimensions. As in [RDR95] we let a_1, a_2, and a_3 represent the length, width, and height of a volumetric part. We can then define the four classes as follows:

$$\begin{aligned}
&\textit{Stick}: && a_1 \simeq a_2 \ll a_3 \vee a_1 \simeq a_3 \ll a_2 \vee a_2 \simeq a_3 \ll a_1 \\
&\textit{Strip}: && a_1 \neq a_2 \wedge a_2 \neq a_3 \wedge a_1 \neq a_3 \\
&\textit{Plate}: && a_1 \simeq a_2 \gg a_3 \vee a_1 \simeq a_3 \gg a_2 \vee a_2 \simeq a_3 \gg a_1 \\
&\textit{Blob}: && a_1 \simeq a_2 \simeq a_3
\end{aligned}$$

If all three dimensions are about the same, we have a blob. If two are about the same, and the third is very different, we have two cases: if the two are bigger than the one, we have a plate, and in the reverse case we have a stick. When no two dimensions are about the same we have a strip. For example, a knife blade is a strip, because no two of its dimensions are similar.

Primitives can be combined to create compound objects. In [RDR95] the different qualitative ways in which primitives can be combined were described—for example, end to end, end to side, end to edge, etc. In addition to specifying the two attachment surfaces participating in the junction of two primitives, we could also consider the angles at which they join, and classify the joints as perpendicular, oblique, tangential, etc. Another refinement would be to describe qualitatively the position of the joint on each surface; an attachment can be near the middle, near a side, near a corner, or near an end of the surface. We can also specialize the primitives by adding qualitative features such as axis shape (straight or curved), cross-section size (constant or tapered), etc.

Functional recognition is based on compatibility with some action requirement. Some basic "actions" are static in nature (supporting, containing, etc.), but many actions involve using an object while it is moving. To illustrate the ways in which a primitive shape can be used, consider the action of "cutting" with a sharp strip or plate. Here a sharp edge is interacting with a surface. The interaction can be described from a kinematic point of view. The direction of motion of the primitive relative to its axis defines the type of action—for example, stabbing, slicing or chopping. These actions all involve primitive motions, which we define to be motions (translations or rotations) along, or perpendicular to, the main axes of the primitive. In this section we will use the detection of primitive motions to infer an object's function.

Inferring Object Function from Primitive Motions

Given a moving object as seen by an observer, we would like to infer the function being performed by the object. The object is given as a collection of primitives. For example, a knife can be described as consisting of two primitives: a handle (a stick) and a blade (a strip). Given this model, the system estimates the pose of the object (as in [RDR95]) and passes this information to the motion estimation module. The model and the results of the motion estimation enable the system to infer the function that is being performed by the object.

The function being performed by an object depends on the object's motion both in the object's coordinate system and relative to the object it acts on (the "actee"). This information gives us the relationships between the direction of motion, the main axis of the object, and the surface of the actee, and these relationships can be used to determine the intended function. For example, we would expect the motion of a knife that is being used to "stab" to be parallel to the main axis of the knife, whereas if the knife is being used to "chop" we would expect motion perpendicular to the main axis. In both cases, the motion is perpendicular to the surface of the actee. If the knife is being used to slice, we would expect back-and-forth motion parallel to its main axis and also parallel to the surface of the actee.

5.2 Computing Motion

Motions of Sticks and Strips

Consider a moving object $\mathcal{B}$. There is an *ellipsoid of inertia* associated with $\mathcal{B}$. The center of the ellipsoid is at the center of mass C of $\mathcal{B}$; the axes of the ellipsoid are called the *principal axes*. We associate the coordinate system $Cx_1y_1z_1$ with the ellipsoid and choose the axes of $Cx_1y_1z_1$ to be parallel to the principal axes. Let $\vec{i}_1$ be the unit vector in the direction of the longest axis l_c (this axis corresponds to the smallest principal moment of inertia); let $\vec{k}_1$ be the unit vector in the direction of the shortest principal axis (this axis corresponds to the largest moment of inertia); and let $\vec{j}_1$ be the unit vector in the direction of the remaining principal axis with the direction chosen so that the vectors $(\vec{i}_1, \vec{j}_1, \vec{k}_1)$ form a right-handed coordinate system.

Here we consider only objects that are approximately planar, straight strips and sticks. For a planar strip the axis of the maximal moment of inertia is orthogonal to the plane of the strip; if the strip is approximately straight, the axis of the minimal moment of inertia is approximately parallel to the medial axis l_c of the strip. In the case of a straight stick, similarly, l_c corresponds to the longest principal axis of the ellipsoid of inertia; the other two principal axes are orthogonal to l_c and can be chosen arbitrarily. We assume that the motion of the stick or strip is planar and that the plane is "visible" to the observer. (The "visibility" constraint allows an oblique view as long as the angle between the surface normal and the z-axis of the camera is $\leq 30°$ (say).) When the object is a strip we assume that the motion is in the plane of the strip; the translational velocity is then parallel to the plane of the strip

and the rotational velocity is orthogonal to the plane of the strip. When the object is a stick the consecutive positions of the stick define the motion plane; the translational velocity lies in the plane and the rotational velocity is orthogonal to the plane. In this case we choose the axis of minimal moment of inertia to be orthogonal to the plane of the motion.

Computing Primitive Motions

We now briefly review our method of computing primitive motions of sticks and strips. A complete description of the method can be found in [DRR96, DFR96].

We associate two rectangular coordinate frames with a rigidly moving body $\mathcal{B}$, one $(Oxyz)$ fixed in space (the camera frame), the other $(Cx_1y_1z_1)$ fixed in the body $\mathcal{B}$ and moving with it (the object frame). The position of the moving frame at any instant is given by the position $\vec{d}_c = (X_c \;\; Y_c \;\; Z_c)^T$ of the origin C (the center of mass of $\mathcal{B}$), and by the nine direction cosines of the axes of the moving frame with respect to the fixed frame. The pair $(\vec{\Omega}, \vec{T})$, where $\vec{\Omega} = (A \;\; B \;\; C)^T$ is the rotational velocity of the moving frame and $\dot{\vec{d}}_c = (\dot{X}_c \;\; \dot{Y}_c \;\; \dot{Z}_c)^T \equiv (U \;\; V \;\; W)^T \equiv \vec{T}$ is the translational velocity of the point C, defines the motion of $\mathcal{B}$. The rotational velocity in the moving frame is $\vec{\Omega}_1 = (A_1 \;\; B_1 \;\; C_1)^T$; we can write $\vec{\Omega} = R\vec{\Omega}_1$ and $\vec{\Omega}_1 = R^T\vec{\Omega}$, where R is the matrix of the direction cosines. From our assumptions about the motion of $\mathcal{B}$ we have $\vec{\Omega}_1 = C_1\vec{k}_1$ and $\vec{T}_1 = U_1\vec{\imath}_1 + V_1\vec{\jmath}_1$.

Let f be the focal length of the camera and let Z_c be the depth of the center of mass C of $\mathcal{B}$. The weak perspective projection of the scene point (X, Y, Z) onto the image point (x, y) is given by

$$x = \frac{X}{Z_c}f, \quad y = \frac{Y}{Z_c}f.$$

For the instantaneous velocity of the image point (x, y) under weak perspective projection we have [DFR96]

$$\dot{x} = \frac{Uf - xW}{Z_c} - C_1(y - y_c)N_z - C_1[(x - x_c)N_xN_y/N_z + (y - y_c)N_y^2/N_z],$$

$$\dot{y} = \frac{Vf - yW}{Z_c} + C_1(x - x_c)N_z + C_1[(x - x_c)N_x^2/N_z + (y - y_c)N_xN_y/N_z],$$

where (x_c, y_c) is the image of (X_c, Y_c) and $\vec{N} = (N_x \;\; N_y \;\; N_z)^T = R\vec{k}_1$ is the normal to the plane of motion; we have also used the fact that $\vec{\Omega} = R\vec{\Omega}_1$.

If we choose a unit direction vector $\vec{n}_r = n_x\vec{\imath} + n_y\vec{\jmath}$ (usually the direction of the image intensity gradient) at the image point (x, y) and call it the normal direction, then the *normal motion field* at (x, y) is $\dot{\vec{r}}_n = (\dot{\vec{r}} \cdot \vec{n}_r)\vec{n}_r$. We then have $\dot{\vec{r}}_n = (\dot{x}n_x + \dot{y}n_y)\vec{n}_r$.

Let $I(x, y, t)$ be the image intensity function. Given the image gradient ∇I and the partial derivative in time I_t of I we have

$$\vec{u}_n = \frac{-I_t \nabla I}{\|\nabla I\|^2}$$

where $\vec{u}_n$ is called the *normal flow*.

The magnitude of the difference between $\vec{u}_n$ and the normal motion field $\dot{\vec{r}}_n$ is inversely proportional to the magnitude of the image gradient. Hence $\dot{\vec{r}}_n \approx \vec{u}_n$ when $\|\nabla I\|$ is large. Expression for the normal flow thus provides an approximate relationship between the 3-D motion and the image derivatives. In [DFR96, DRR96] the normal flow (an observable quantity) was used as an approximation of the projected motion field. The method of least squares estimation was used to obtain the estimates of C_1, U/Z_c, V/Z_c, and W/Z_c. These estimates and the fact that the object was "visible" were then used to obtain the values of U_1/Z_c and V_1/Z_c.

Parametrizing the Motion of a Stick or Strip

We use three angles α, β, and θ to parametrize the motions of sticks and strips.

The direction α of the medial axis is found using the following algorithm:

1 - Make a sorted (circular) list of all edge elements (sorted by their orientations modulo π) for which the normal flow is computed.

2 - Find the shortest segment $[\gamma_1, \gamma_2]$ such that more than 3/4 of the orientations in the list are contained within it.

3 - Find the median orientation α in the sorted sublist chosen in the previous step.

4 - If α does not approximately agree with the pose that was estimated earlier, then $\alpha \leftarrow \alpha + \pi$.

5 - Use α as the orientation of the medial axis.

We estimate (x_c, y_c) — the image position of C (the reference point and the center of mass of the object)—as the average of the coordinates of all edge points for which the normal flow is computed.

We define β as the angle between the vector $(U_1 \quad V_1 \quad 0)^T$ and the Cx_1 axis of the tool coordinate system; thus

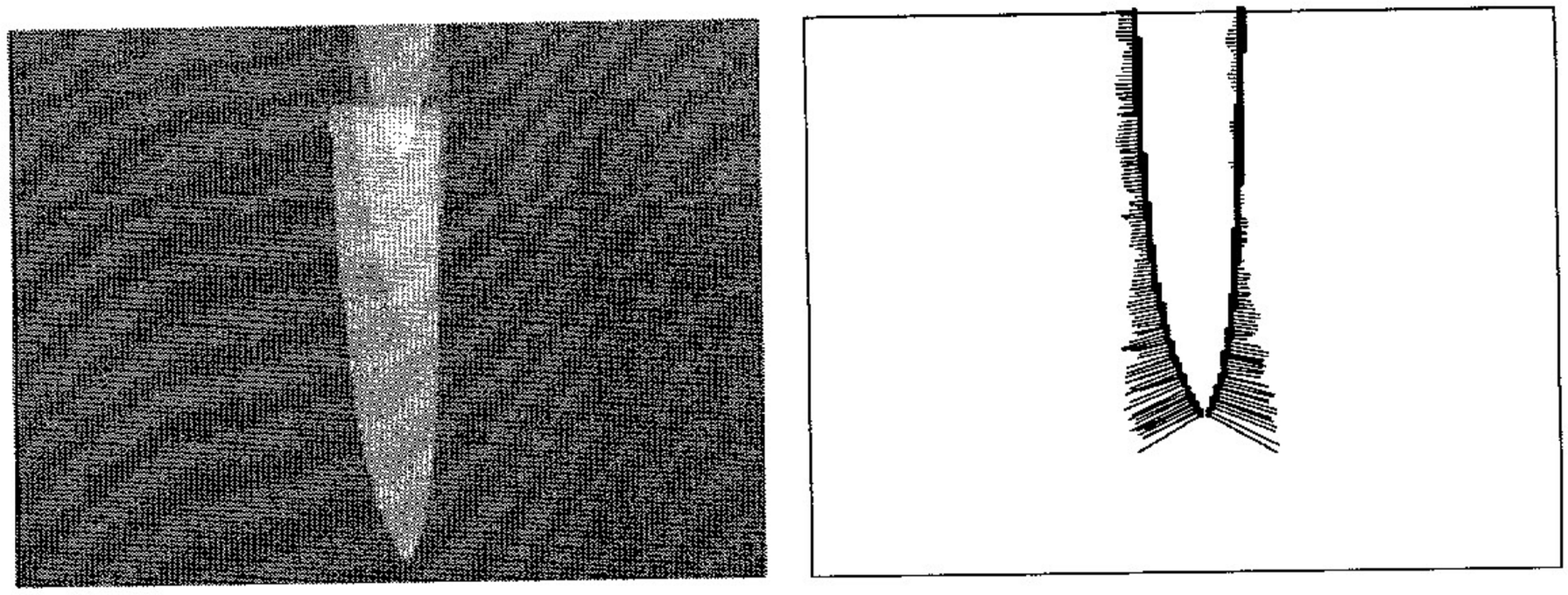
$$\beta = \arctan \frac{V_1}{U_1}.$$

We define θ to be the total rotation angle as a function of time:

$$\theta = \int_0^t C_1 dt.$$

5.3 Experiments

In our experiments we observed the motion of a knife performing a task. The vision system took images at 25 frames per second for 5 seconds, yielding 125 images per experiment. After



(a) (b)

Figure 8: (a) Stabbing motion. (b) Flow vectors for Stabbing.

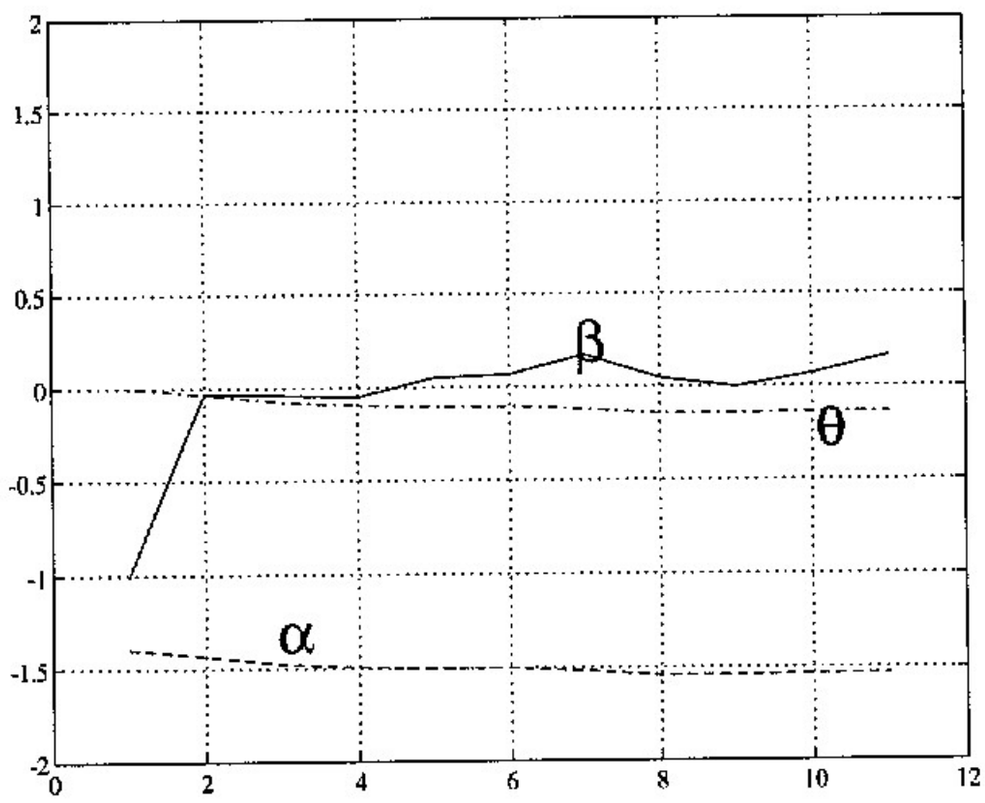
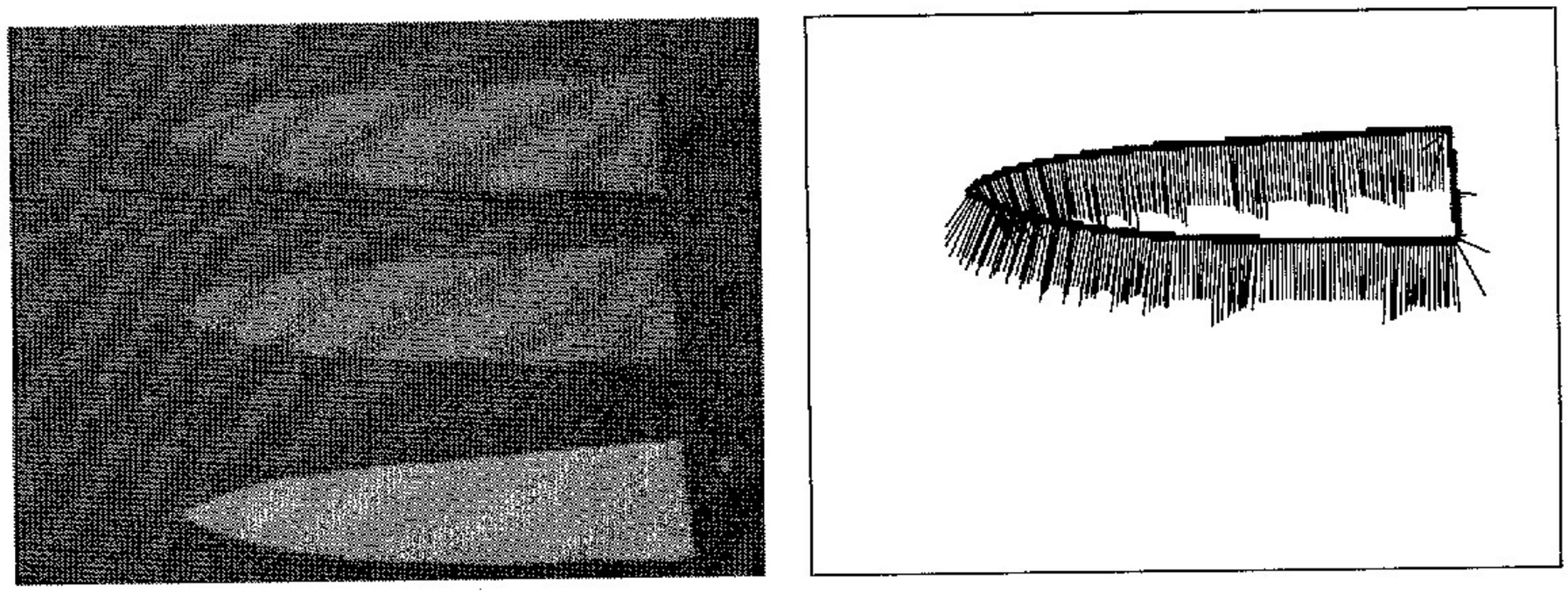


Figure 9: Angles α, β, and θ for Stabbing.



(a) (b)

Figure 10: (a) Chopping motion. (b) Flow vectors for Chopping.

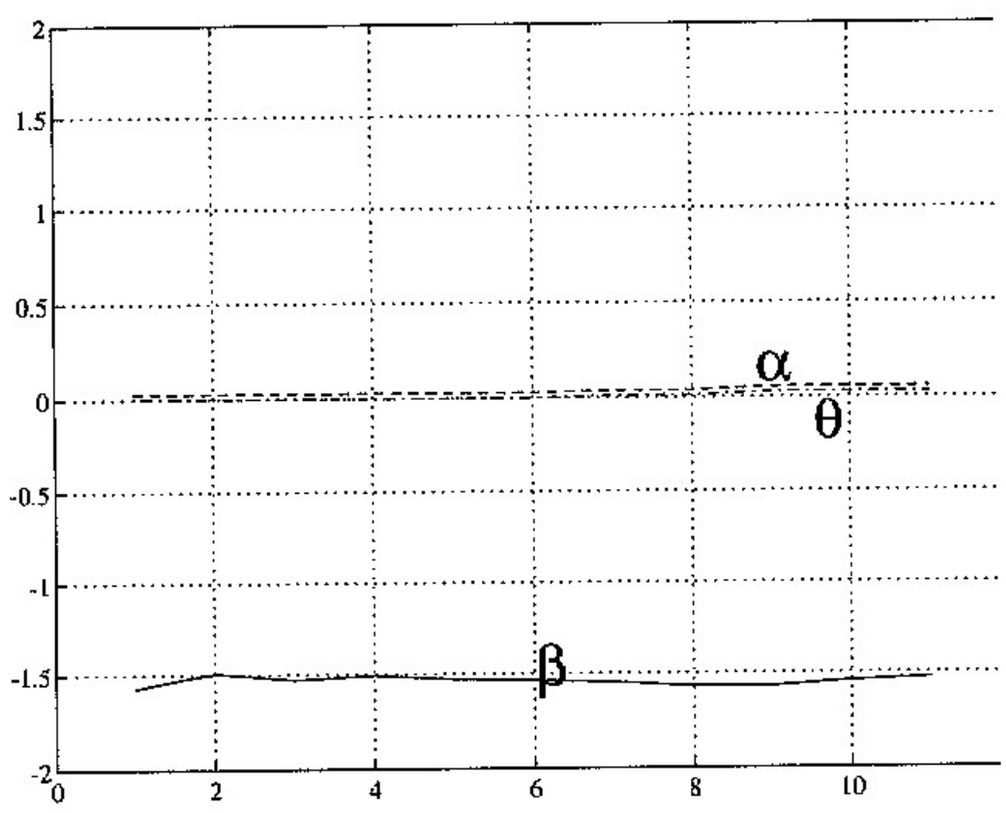
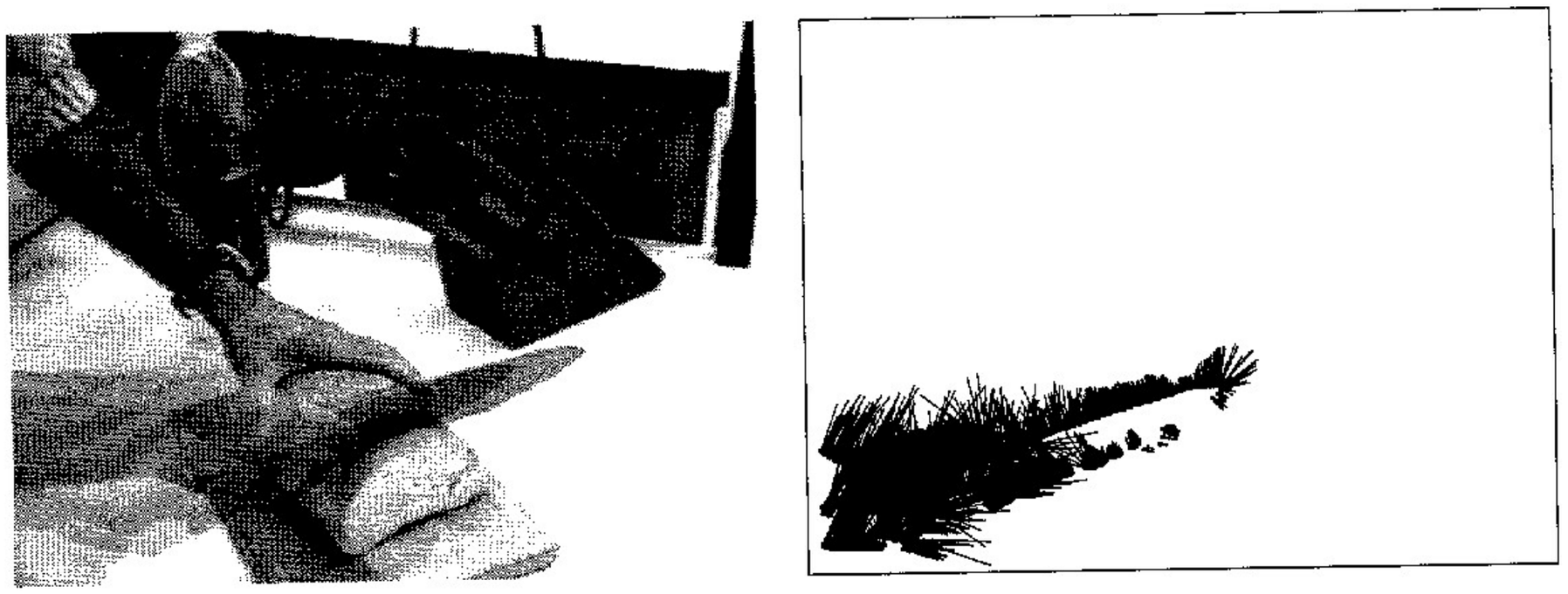


Figure 11: Angles α, β, and θ for Chopping.



(a) (b)

Figure 12: (a) Slicing motion. (b) Flow vectors for Slicing.

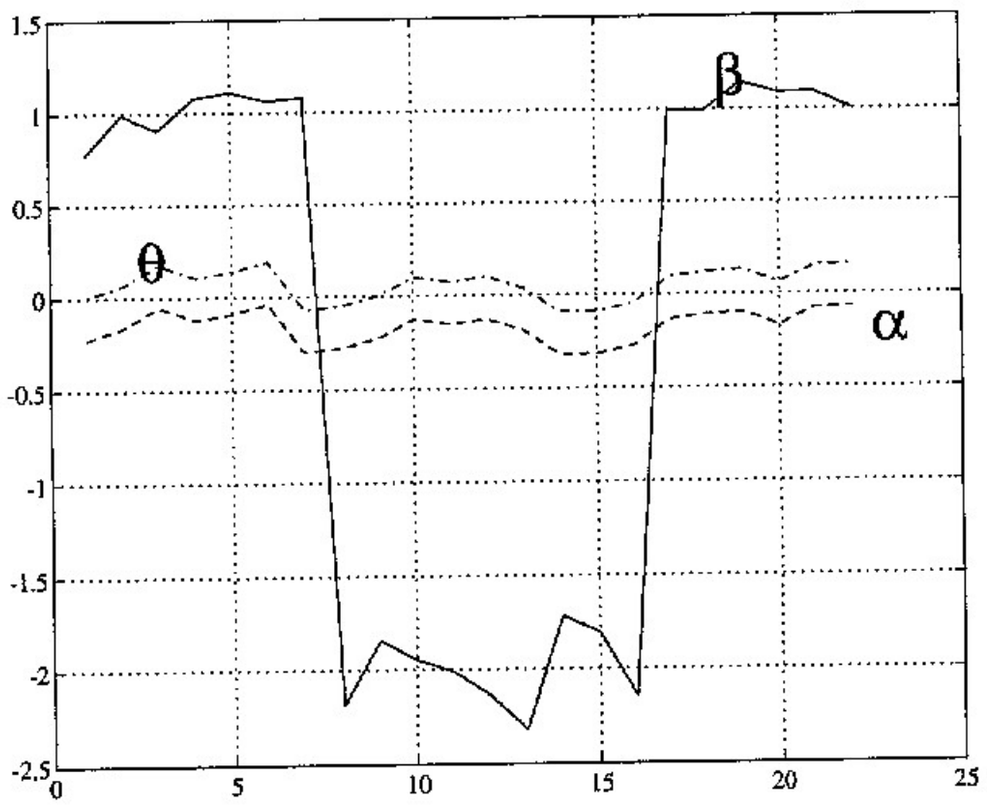


Figure 13: Angles α, β, and θ for Slicing.

each image sequence was recorded, a representative sampling of the 125 images was used for further processing. Eleven evenly spaced samples, each composed of three consecutive images, were used. (For instance, samples 1 and 2 in any given experiment used images 0–2 and 10–12, respectively.) This resulted in 33 images for each experiment.

Stabbing

Stabbing is defined as the cutting motion of a knife in which α (the angle between the projection of l_c onto the plane $Z = Z_c$ and the Ox axis) is close to either $-\pi/2$ or $\pi/2$, β is approximately 0, and θ is small and approximately constant.

Figure 8 shows the flow vectors taken from the 6th sample and a composite image of the knife taken from the 1st, 6th and 11th samples of the stabbing experiment. Figure 9 shows a plot of the triple (α, β, θ) with respect to time (frame numbers). We see that as was expected, the values of α are very close to $-\pi/2$, β is close to 0, and θ is close to 0. A VL_1 rule (Michalski [Mic72]) describing stabbing would be

$$< stabbing > \; <:: \; [\alpha = -1.55 \; .. \; -1.35] \; \& \; [\beta = -1 \; .. \; 0.2] \; \& \; [\theta = -0.2 \; .. \; 0].$$

Chopping

Chopping is defined as the cutting motion of a knife in which α (the angle between the projection of l_c onto the plane $Z = Z_c$ and the Ox axis) is close to either 0 or π, β is close to $\pi/2$ (when $\alpha \approx \pi$) or $-\pi/2$ (when $\alpha \approx 0$), and θ is small and approximately constant.

Figure 10 shows the flow vectors taken from the 6th sample and a composite image of the knife taken from the 1st, 6th and 11th samples of the chopping experiment. Figure 11 shows a plot of the triple (α, β, θ) with respect to time (frame numbers). We see that, as was expected, the values of α are very close to 0, β is close to $-\pi/2$, and θ is close to 0. A VL_1 rule describing chopping would be

$$< chopping > \; <:: \; [\alpha = 0] \; \& \; [\beta = -1.6 \; .. \; -1.5] \; \& \; [\theta = 0].$$

Slicing

Slicing is defined as the cutting motion of a knife in which α is approximately 0 (or $< \pi/2$), β oscillates between approximately 0 and approximately π, and θ is small and approximately constant.

Figure 12 shows the flow vectors taken from the 6th sample and a composite image of the knife taken from the 1st, 6th and 11th samples of the slicing experiment. (The mass of vectors at the left end of Figure 12a come from the motion of the hand, which is visible in the images.) Figure 13 shows a plot of the triple (α, β, θ) with respect to time (frame numbers). We see that, as was expected, the values of α are very close to 0, and that β oscillates between

approximately $\pi/2$ and approximately $-3\pi/2$ (note that the two approximate values differ by π). A VL_1 rule describing slicing would be

$$< slicing > \quad <:: \quad [\alpha = -0.25 \,..\, 0] \;\&\; [\beta = -2.25 \,..\, -1.75,\ 0.75 \,..\, 1.25] \;\&\; [\theta = -0.2 \,..\, 0] \;\&\; [T_\beta = 8..12]$$

where T_β is the period of β (β changes between two ranges with the period T_β).

6 Recognizing Targets in SAR Images: System Architecture and Pilot Study

Automatic Target Recognition (ATR) is concerned with the use of computer technology for detecting and recognizing designated objects/targets in sensor data, such as SAR, FLIR, TV camera and others. ATR is of a great importance for modern defense because it is essential for precision strikes against designated targets with minimum collateral damage to other objects. Research on ATR is also highly relevant to non-military problems, for example for recognizing objects in visual navigation systems, mobile robots, detecting landmarks, etc. The problem of detecting and recognizing objects in a natural environment (in particular, targets) is extremely difficult for many reasons [Strat, 1992; Dudgeon and Lacoss, 1993; Fischler, 1996]. Here are major reasons:

- Targets can vary very significantly in appearance. The appearance depends on the viewing angle, distance, the position and orientation of articulated parts (e.g., the tank turret), the quality and type of ambient light, and so on. Targets may blend with environment.
- Sensors typically used in ATR (e.g., SAR, FLIR, and others) may produce low quality images.
- Natural objects act as clutter and can vary greatly from scene to scene. Their color, light reflectivity, and temperature depend on the weather conditions and the time of the day. The generic classes of natural objects cannot be described in terms of metric shape. Their shape can be defined only "qualitatively".
- Targets can be partially or completely occluded, or camouflaged, i.e., their appearance can be deliberately changed.

A truly effective method should be able to cope with all these problems. At this stage, such a method does not exist. A very significant amount of effort has been put to ATR research and many methods have been developed (e.g., [Dudgeon and Lacoss, 1993]). These methods typically apply a statistical pattern recognition approach or neural net techniques. Their performance is directly dependent on the features that are computed for characterizing objects of interest (targets), such as the object size, length, height, fractal dimension, weighted-rank fill ratio, polarimetric properties, spatial distribution of reflected power, etc. (e.g. [Kreithen et al., 1993]).

An important idea that adds to these techniques is to use context information. For example, Strat [1992] used the position of the ground vehicle with mounted camera obtained from the GPS data, the camera position and orientation, and the digitized geographical data from the USA geographical survey to form the ground model. The ground model was then used to simplify figure/ground discrimination. Rong and Bhanu [1996] use time of the day, air temperature, target range, and depression angle of the target as contextual information for reinforcement learning and recognition of targets in cluttered FLIR images. Note also that the context need not always be supplied in advance — the concepts which are recognized in the image can become the context for recognizing other concepts [Strat, 1992].

Using multiband images or information fusion from various sources is frequently used to improve the detection and recognition of targets. Beveridge et al. [1996] combine color, FLIR, and LADAR images for target detection and recognition. They show that objects that are similar in color images differ in FLIR images. LADAR images are used for target recognition once it is detected. Although this approach is very promising, it requires registration of various types of images; usually one can expect only an approximate registration and overlapping of images [Beveridge et al., 1994].

An important component of the ATR is the preprocessing of images and the identification of regions of interest. This is usually done as fast "screener" and "detection" phases that lead to a delineation of the potential target areas [Dudgeon and Lacoss, 1993; Kreithen et al., 1993; Novak et al., 1993; Beveridge et al., 1996]. Subsequently, in the slower "recognition phase" the system recognizes the potential targets in the regions of interest. Such a two-phase approach is also used in other computer vision problems. Draper et al. [1994] find the candidate road pixels in color images and these are then used as input to stereo matcher; the detection is tuned to minimize number of misses (or false negatives). Maloof et al. [1996] developed a method for recognizing blasting caps in x-ray images. In the detection phase they search for low intensity blobs which correspond to heavy metal cores of blasting caps.

To deal with variations in object appearance, explicit 3D modelling of targets is frequently done [Beveridge et al., 1996; Verly et al., 1996]. The range (e.g., LADAR) images are matched to the stored models of potential targets. A successful match provides recognition and position and orientation of the target. Since this part is very expensive in terms of time and processing power it is essential that the system reject as much as possible of the clutter. An important way to help ATR is to exploit the properties of different sensor modalities and use different sensors for different aspects of the problem. For example, color and FLIR can be used to detect potential objects. FLIR will differentiate between a rock and a running tanks, because it is sensitive to the hot objects. LADAR can be useful for 3D recognition. A problem arises in which the system must automatically determine what combination of sensors is appropriate for a given situation.

The machine learning approach to the ATR problem is to detect and recognize targets from a certain class in an environment whose characteristics correspond to the training environment. In ATR the easiest problem is to detect unoccluded targets in natural environments. Camouflage and occlusion add to the difficulty. Finally, the hardest problem is to detect targets in built-up areas where they can be fully or partially occluded by various

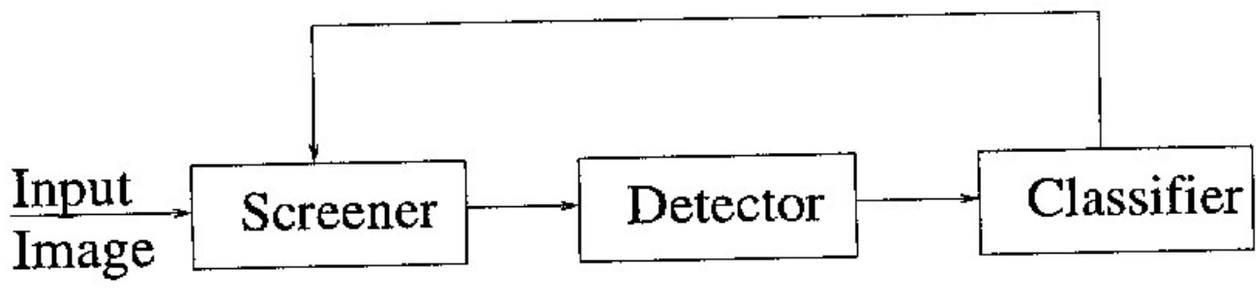


Figure 14: The architecture of the proposed system.

cultural clutter.

The proposed system consists of Screener, Detector and Classifier modules (Figure 14). The general architecture thus resembles that of Novak et al. [1993]. The Screener preprocesses input images, and rejects most of the natural clutter such as trees, grass, etc. The Detector determines areas with potential targets. The Classifier determines the type of targets. This basic system architecture is used for all sensory modalities, but for the purpose of this proposal we will illustrate its use on polarimetric SAR data. In the pilot study to illustrate some of the aspects of the proposed methodology, we used polarimetric, high-resolution SAR data collected by the MIT Lincoln Laboratory in the area of Stockbridge, New York, and supplied to us by DARPA.

6.1 Screener

The SAR images used in the pilot study were polarimetric SAR images, in which the value at every pixel is a complex four-element vector. Usually, only components $(HH\ HV\ VV)^T$ are used. To combine images from individual channels to a single channel, we implemented the *polarimetric whitening filter* (PWF [Novak et al., 1990]. The power of the PWF image is given by

$$y(i,j) = |HH|^2 + \left|\frac{HV}{\sqrt{\varepsilon}}\right|^2 + \left|(VV - \rho^* \sqrt{\gamma} HH) \cdot \left(\gamma\left(1 - |\rho|^2\right)\right)^{-\frac{1}{2}}\right|^2$$

where y(i,j) are the pixel coordinates, X^* stands for a complex conjugate of X and

$$\varepsilon = \frac{E(|HV|^2)}{E(|HH|^2)}, \quad \gamma = \frac{E(|VV|^2)}{E(|HH|^2)}, \quad \rho = \frac{E\left(|HH \cdot VV^*|\right)}{\sqrt{E(|HH|^2) \cdot E(|VV|^2)}};$$

where E() stands for expected value or ensemble average. In our implementation of PWF, we followed a suggestion by the authors [Novak et al., 1990] and used the expected values for grass; namely $\varepsilon = 0.19$, $\gamma = 1.03$ and $\rho\sqrt{\gamma} = 0.53$. An example of SAR image after applying of the PWF is shown in Figure 15. Images obtained from PWF were then processed according to a global CFAR (constant false alarm rate) algorithm [Ravid and Levanon, 1992; Weng et al., 1993]. We followed the implementation presented in [Weng, Chellappa, and Zheng, 1993]. This algorithm produces an image $d(i,j) = rint(1000\sqrt{y(i,j)})$ ($rint()$ stands for "round to integer"). We used 60% of the smallest pixel values ($d(i,j)$) to estimate the parameters of the Weibull distribution for the background. The parameters of the distribution were then

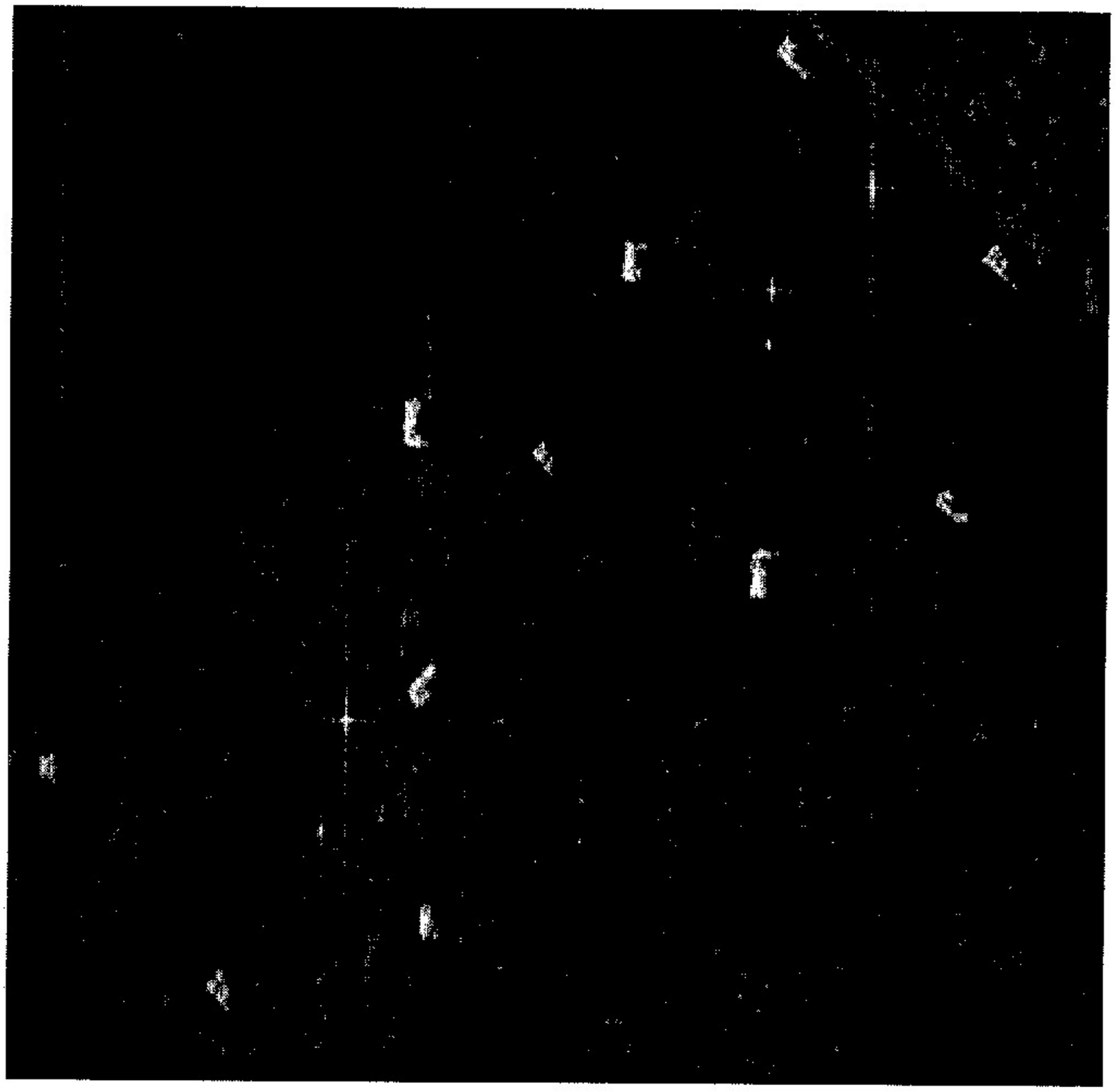


Figure 15: A target array as seen in a SAR image. 4 ADTS files were used to compose "pass 1" (files m78p1f[20-24].adts) and PWF was applied to the result. Central portion of the image containing a target array (2 M-55 howitzers, 2 M-60 tanks, 3 M-48 tanks, 1 M-113 APC, 2 M-84 APCs, and 1 M-59 APC) is shown here.

used to compute threshold T; all pixels whose $d(i,j)$ value is larger than T are marked as "target" pixels. The threshold T was computed for the false alarm rate $P_{FA}^{GC} = 10^{-6}$. The results are shown in Figure 16. The white pixels passed the CFAR test. All targets were indicated, but there are also many pixels indicating non-targets.

6.2 Detector

We base our detector on a multistrategy learning system AQ-NN. We find the regions of interest based on the output of a Screener. For part of the image in the region of interest we compute the values of various attributes. An expert marks the boundary of the target region (as a box or a polygon). All pixels inside the boundary curve are then labeled as target pixels. The process is repeated for all targets (11 in the example in Figure 15). After the targets are

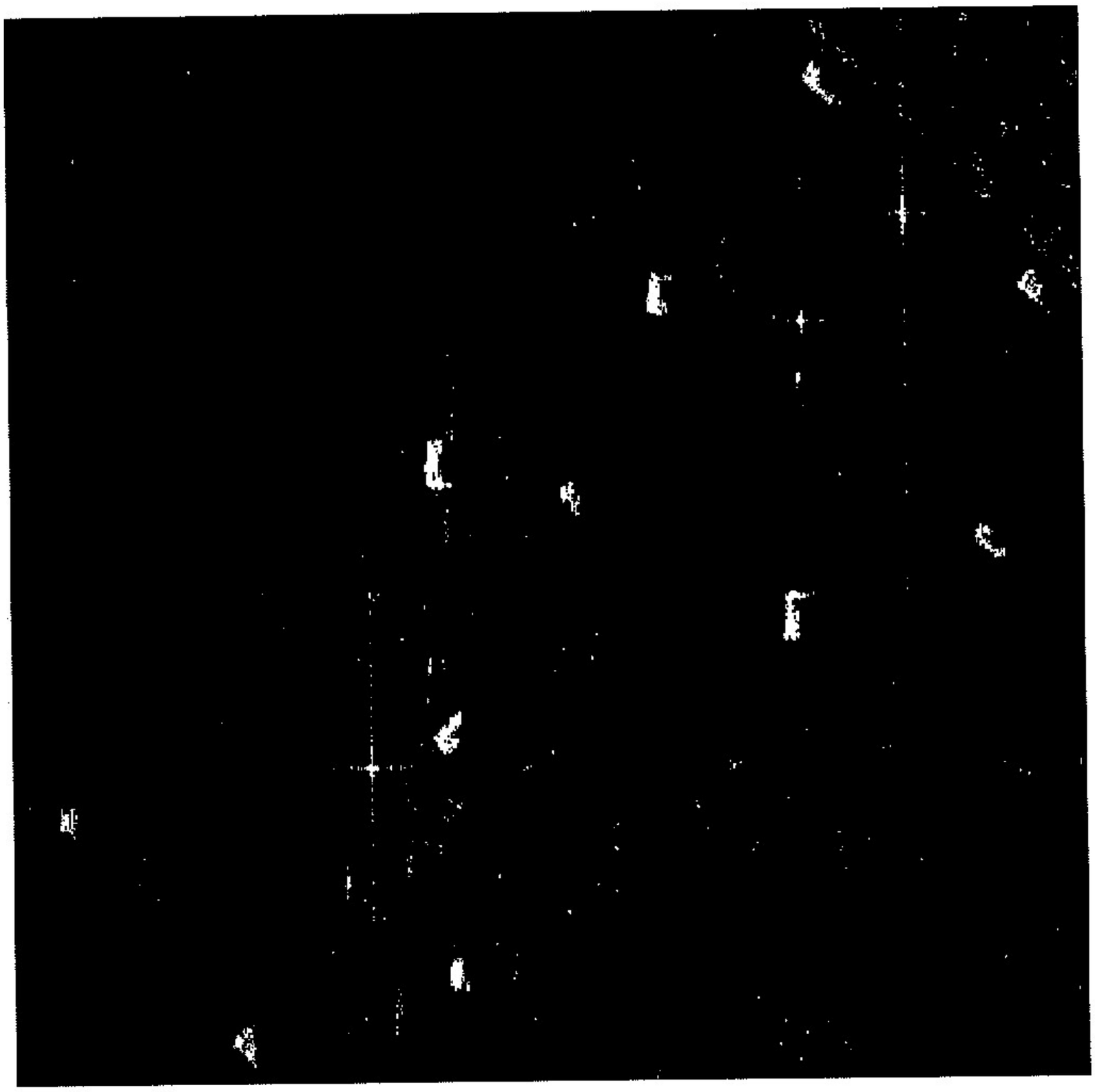


Figure 16: Results of CFAR algorithm on PWF image of the target array in Figure 15.

labeled remaining pixels which passed the CFAR test are labeled as non-target pixels. After labeling the symbolic learning system AQ is run to create the rules that describe target and non-target concepts. After creating the rules the supervisor reviews the rules and applies them to the testing data. The supervisor reviews the results and decides whether to save the rule set, prune the rules or change the representation space (using constructive induction) and/or prune the data set [Michalski et al., 1996].

A preliminary implementation of the Detector was done on the SAR data obtained from DARPA. We computed 5 attributes over circular regions centered at pixels which passed the CFAR test. The features were:

(i) Sum of PWF values in the region (*energy*).

(ii) Number of pixels in the region which passed the CFAR test (*area*).

(iii) Weighted rank-fill ratio in the region ($wrfr$).

(iv) Standard deviation of PWF values in the region (*std*).

(v) Fractal dimension of the pixels which passed CFAR test in the region ($fractal$).

We computed the features for regions of diameter 31 pixels (attribute names with label 1) and diameter 21 (attribute names with label 2). We applied our detector to the "pass 7" SAR image (see Figure 17) of the target array shown in Figure15. The difference in the heading direction of the platform between passes 1 (Figure 15) and 7 (Figure 17a) is 45 degrees. After applying our Screener and Detector we used morphological operations of "closing" and "opening" to connect disconnected target pixels. The result is shown in Figure 17b. As can be seen all eleven targets were detected and only one non-target is shown.

6.3 Classifier

The proposed classifier is based on the dynamic recognition approach described above. It is used to classify targets into predetermined target classes, and to reject non-target, such as man-made objects (cultural clutter) and other non-targets similar to targets. In this approach, each target is described by a characteristic target description (C-signature). The results of classification are the pose of the object and its class membership. Furthermore, the dynamic recognition method allows hierarchical description, i.e. in the example we have a class tank has sub-classes M-60 and M-48. The object (target) descriptions are composed of features which are based computationally on models of possible targets. It is well known [Chellappa et al., 1992] that the targets have higher radar returns than natural clutter. Also, the responses of various target parts differ. Figure 18 shows a response for a M-55 howitzer. Furthermore, some variations in response are due to relative orientations of the target and the platform. We use this fact for feature design.

Once a potential target is detected (using Screener and Detector) we threshold the image by finding the $k\%$ (we may use few different values of k, but 5% and 10% are typical) of the brightest returns in the circular region centered at the potential target region. The region typically separates into several peaks. We first find the minimal area box surrounding all the peaks. The relative orientation of the box and the SAR platform (heading direction) give us one feature. Box aspect ratio and its size are good features too. Other features are based on the strength and distribution of the peaks within the box. Some of the additional features are:

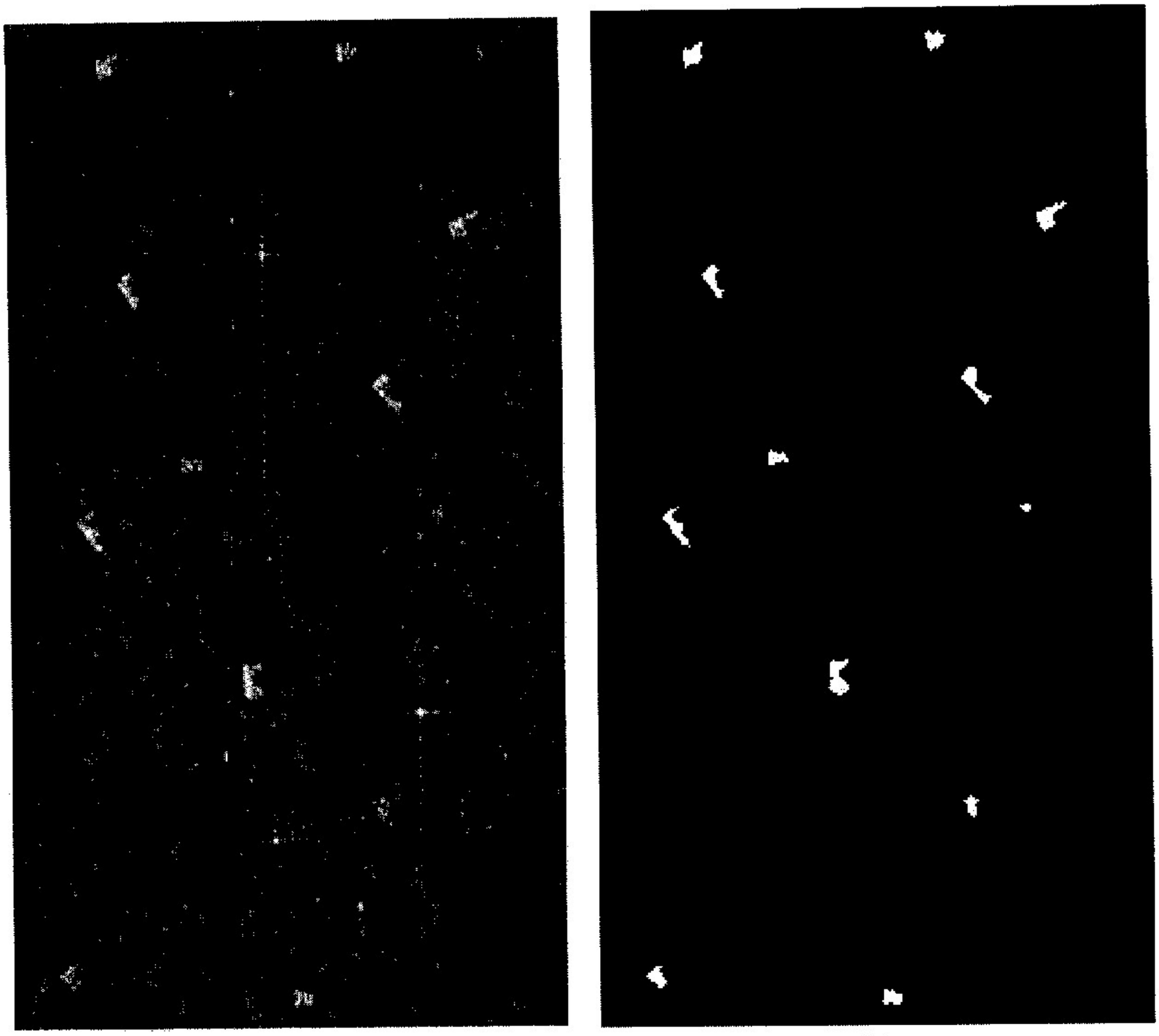
(i) the number of peaks,

(ii) ratio of energies of two highest peaks,

(iii) the energy of the highest peak divided by the total energy within the box.

Since different orientations of targets result in different combinations of attributes (and attribute ranges) each target can be define by several characteristic descriptions. Thus, recognizing a target through the use of a particular characteristic description gives the target pose in addition to the target class. We are investigating additional features for use in our classifier as well as robust methods for feature computation.

The crucial aspect of target classification is to determine a sufficiently relevant representation space, that is object attributes to be measured, their ranges and their types. Our



(a) (b)

Figure 17: Results of the detection phase when applying learned operators to a new image. (a) The target array shown in Figure 15 as seen on the "pass 7" of the SAR platform. (b) Results of target detection on image in (a). All eleven targets and only one non-target were detected.

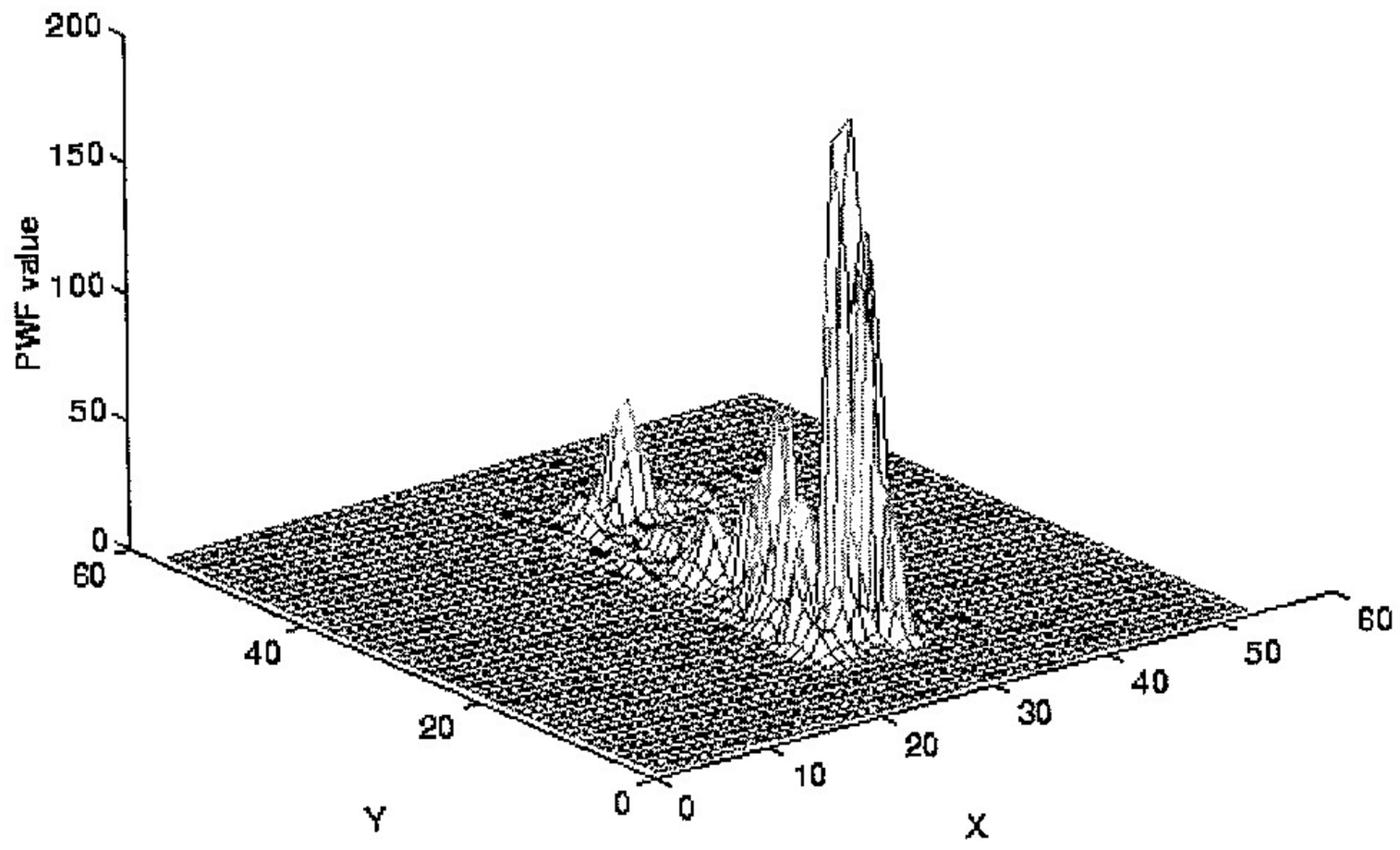


Figure 18: The SAR image of a howitzer M-55. The SAR platform is heading in Y direction.

methodology is able to handle attributes that are nominal (unordered domain values), linear (discrete or continuous), and structured (a hierarchically ordered domain values).

For the pilot study, we chose a very simple representation space, spanned over easily computable attributes. Targets usually appear as blobs in the image. Therefore, is it difficult to measure their shape with high accuracy. As could be easily observed, different structures reflect SAR signals in different degrees. Important attributes involve structural object characteristics, such as the presence of a turret, a large underlying base, etc. To capture such structural characteristics, a thresholding operation was performed on each detected potential target (i.e., after the detection phase). The resulting image had several peaks for each potential target. To characterize each peak, we used attributes such as peak energy, the size of the area containing the peak, ratio of the energy of the peak to the energy of the target area, and the ratio of the peak area to the area of the target. The number of peaks in the target area was also an attribute. Peaks were ordered according to their highest PWF value. Each target was described in terms of such attributes. In the experiments we distinguished six types of targets: M-60 tank, M-48 tank, M-55 howitzer, M-59 APC, M-84 APC, and M-113 APC.

The targets in Figure 19 (eleven targets from Figure 15) were used to learn descriptions of targets under consideration. Detected targets were used for learning of characteristic descriptions (C-signatures) for the classification phase. As can be seen in Figure 19, there is a great variability in aspect and intensity of targets of similar type. Preliminary classification experiments were conducted on images of targets in Figure 20 (these are targets from the array in Figures 17a and 17b). The classification results were as follows: A single non-target which was passed by the detector (see Figure 17b) was classified as a non-target. The M-55 howitzer in Figure 20b was classified correctly as were M-48 tanks in Figures 20e–g and the

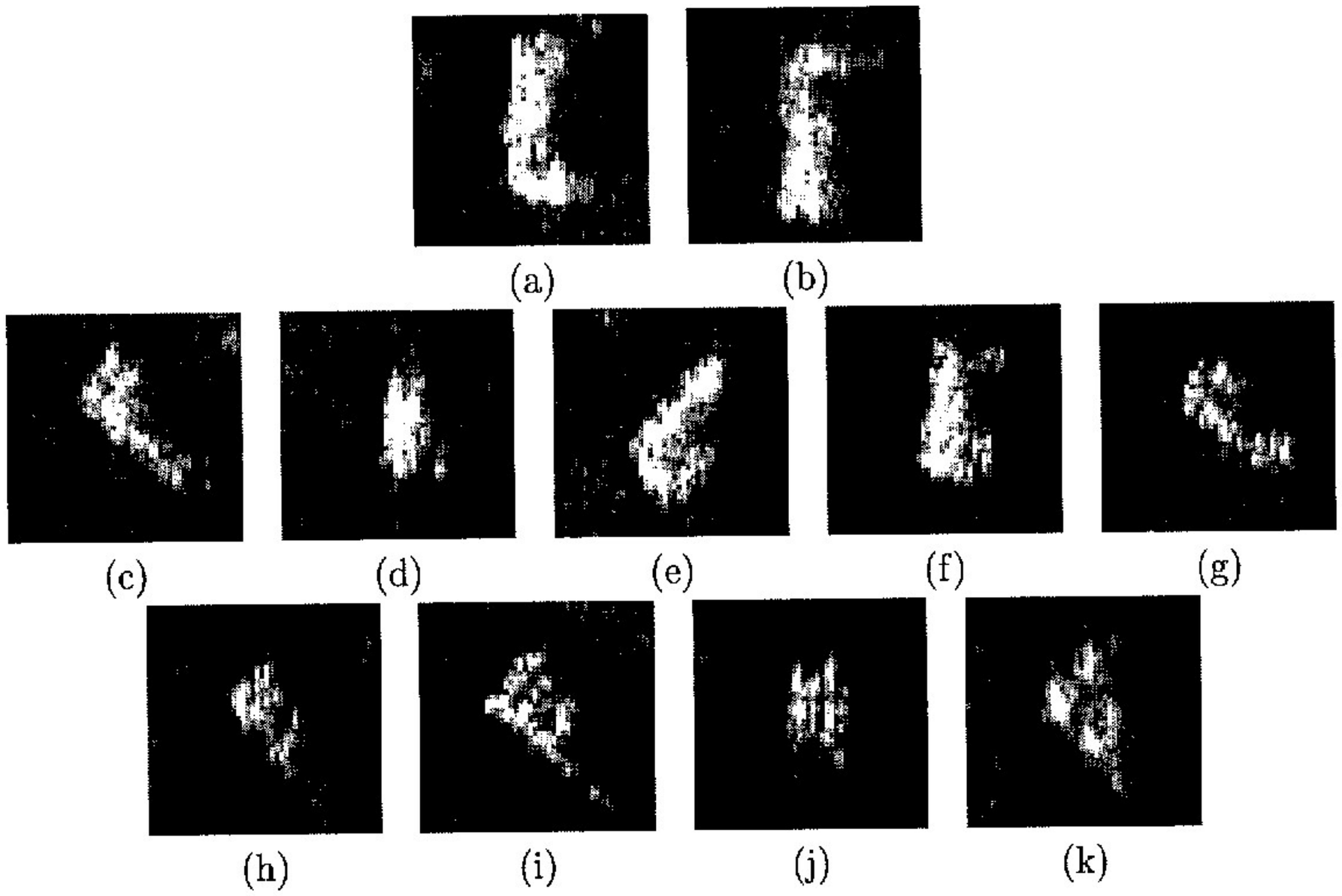


Figure 19: Images used for learning. (a) and (b) are M-55 howitzers. (c) and (d) show M-60 tanks, and (e)-(g) are M-48 tanks. (h) is an M-113 APC, (i) and (j) are M-84 APCs, and (k) is an M-59 APC.

M-113 APC in Figure 20. The M-60 tank in Figure 20c was classified as a M-48 tank, and the M-60 tank in Figure 20d was classified as a non-target. The M-84 APC in Figure 20j was classified as a M-48 tank as was the M-59 APC in Figure 20k. Finally, the M-84 APC in Figure 20i was classified as a non-target.

These preliminary results are encouraging, but much remains to be done. In this experiment, we had only between one and three training examples. This is insufficient. The problem is that radar signal return patterns for each type of target in the training image can be substantially different from that in the testing image. The radar signal return pattern of a target varies with the azimuth direction, aspect angle of the imaging plane, as well as other factors. The system determined some patterns, e.g., that non-targets have usually fewer number of peaks than target and their peaks have small areas. More images with ground truth are needed to develop a highly relevant representation space. We need to study their relations to the azimuth direction and other factors and determine attributes that are relatively invariant to changes of appearances of targets in the same class.

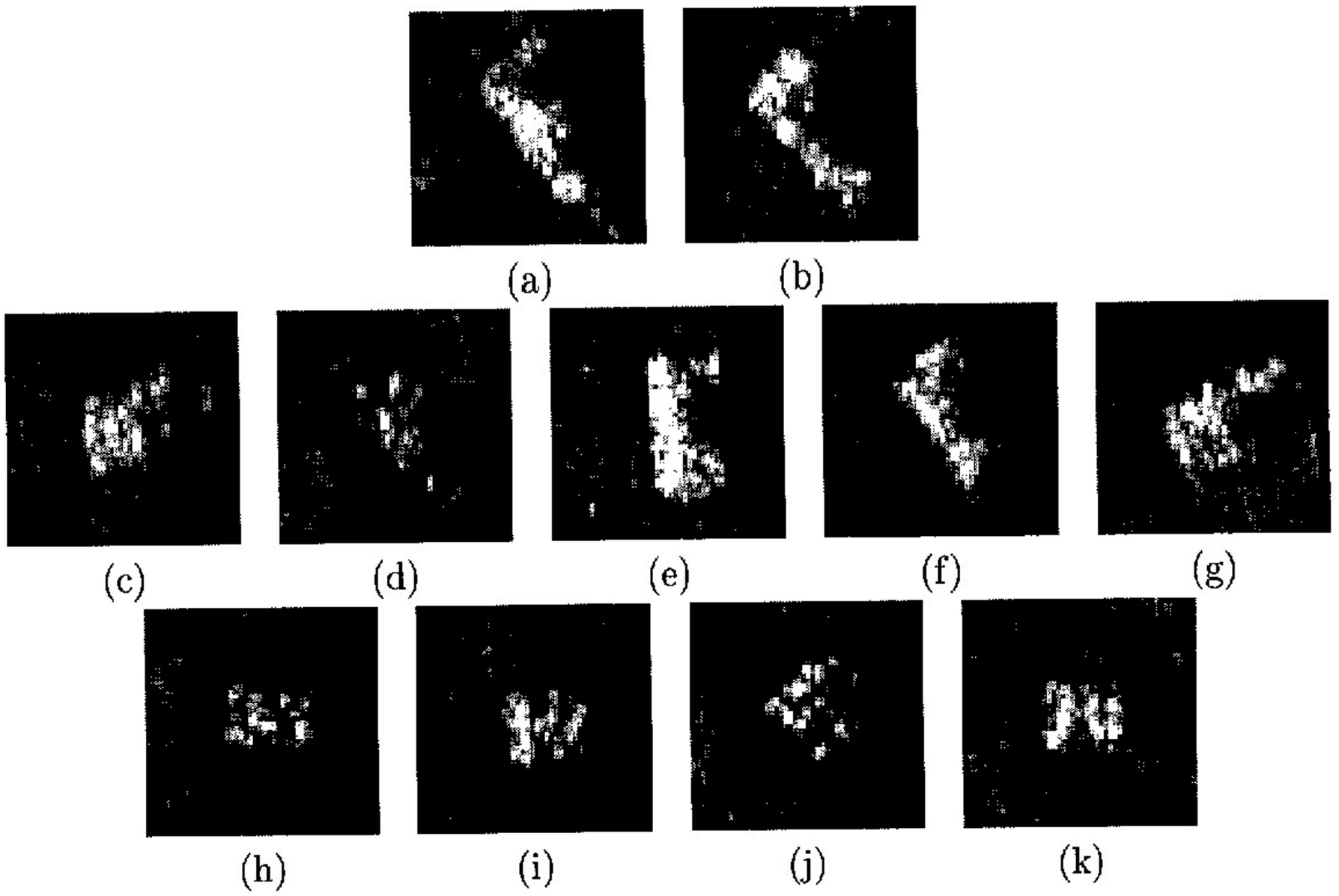


Figure 20: Images used for learning. (a) and (b) are M-55 howitzers. (c) and (d) show M-60 tanks, and (e)-(g) are M-48 tanks. (h) is an M-113 APC, (i) and (j) are M-84 APCs, and (k) is an M-59 APC.
Results of classification. (a) and (b) are M-55 howitzers. (c) and (d) show M-60 tanks, and (e)-(g) are M-48 tanks. (h) is an M-113 APC, (i) and (j) are M-84 APCs, and (k) is an M-59 APC. The targets are shown in the same order as in Figure 9.

7 "Robotic" Estimation: The Inefficiency of Random-Walk Sampling

A robotic agent may be required to obtain samples of its environment, whether simply for the sake of exploration, as in the case of a Mars Rover collecting samples of data on the surface of Mars, or because the data that it collects allows the agent to choose an effective strategy for navigating in the environment. Based on the samples that it collects, the agent can attempt to estimate statistical properties of the environment, but an important limitation on the ability of an agent to do this is that it is usually impractical for the agent to sample the environment randomly. An agent is often "myopic," and can measure properties only in the vicinity of its current location. In addition, an agent is usually constrained to move "contiguously," that is, to follow a connected path through the environment. Thus a succession of samples of local property values collected by an agent is not a random sample; it consists of a sequence of consecutive samples that lie along a path. We refer to an estimate based on such a sample as a "robotic" estimate.

Suppose the agent is required to collect samples from a particular region. One possible strategy for doing this is to traverse the region systematically; but if the agent's power source (or fuel supply) is limited, it may not be able to complete the traversal, or its resources may not be fully expended when the traversal is complete. A better strategy from a statistical viewpoint is to take a random sample; but a connected path that visits all the points of such a sample will also visit many other points without sampling them, so that random sampling is wasteful. A reasonable compromise strategy is to "approximate" random sampling subject to the constraint of contiguous movement—for example, to follow a path that is a random walk.

Unfortunately, if consecutive samples along a path are correlated, taking samples along the path—even if the path is a random walk—is less efficient than taking random samples; a larger sample is required to obtain an estimate of a given accuracy. We have studied the inefficiency of robotic estimation, specifically of estimation based on a random-walk path, relative to estimation based on random sampling. For concreteness, we refer to the region being sampled as "terrain," and the quantity being measured at the sample points as "elevation."

In our experiments, the terrain was represented by a discrete hexagonal grid, and an elevation is assigned to each grid point. At any point in time, the agent is located at one of the cells of the grid, and it is able to measure the elevation of the terrain at that cell. Furthermore, the agent can move from a cell to any of its six neighbors. In this domain, the agent might benefit from being able to estimate quantities such as the mean and variance of the terrain elevation; for example, it might decide, on the basis of its estimate of the elevation variance, that the terrain is too rugged for exploration.

We studied the agent's sampling efficiency on two types of terrain: "rugged" terrain, in which the cell values are independent and uniformly distributed over the interval $[-\frac{1}{2}, \frac{1}{2}]$; and "smooth" terrain, which is derived from rugged terrain by replacing the value at each cell with the average of the values from the seven-cell neighborhood centered at that cell. Note that the smoothing process introduces substantial correlation between the values of nearby cells, since their neighborhoods may overlap.

We studied the efficiency with which the agent can estimate the mean and variance of the terrain elevation when it samples the elevations at the successive points of a random walk; in particular, we compared the efficiency of random-walk sampling with that of random sampling.

We performed a theoretical analysis of random-walk sampling of "terrains" and carried out a number of experiments to validate this theory. For both independent and correlated cell values (rugged and smooth terrain), we compared the variance of the mean estimated by a random walk to the variance of the mean estimated by a random sample, and the variance of the variance estimated by a random walk to the variance of the variance estimated by a random sample. The results are plotted in Figures 21–24 as functions of the sample size, N. In each plot, the dashed curves are the curves predicted by the theory, and the solid curves are those obtained empirically. To generate the empirical data, 500 random terrain grids of

128×128 cells were examined for each sample size in each plot, and the mean and variance of a random sample and a random walk sample on each grid were recorded. The variances of these recorded means and variances are shown in the plots. The empirical data were in close agreement with the theory, except at the smallest sample sizes.

Our results indicate that random walk sampling is approximately 40% as efficient as pure random sampling for mean estimation on rugged terrain, and 30% as efficient for variance estimation. On smooth terrain, the figures are approximately 8% (mean) and 16% (variance). A more interesting comparison, illustrated in Figures 25–28, is between the sizes of the samples needed for the two estimators to achieve the same variance. These plots were created by fitting functions of the form $y = a/(1 + bx)$ to the theoretical curves, extrapolating the inverse functions to very large sample sizes, and then computing the ratios of the sample sizes as functions of the variance. (Curve fitting was used because, for such large sample sizes, it would have been prohibitively costly to use the formulas in [CNR97], and closed-form formulas are not available for the random walk curves.) The results demonstrate the extent to which the constraint of contiguous movement limits the ability of an agent to estimate properties of its environment. For example, to achieve a variance of 5×10^{-6} in the estimation of variance on smooth terrain, random walk sampling requires a sample size nearly ten times as large as that required by pure random sampling.

Our results showed that the constraint of contiguity on a robotic agent's movement, that is, the need to follow a connected path through its environment, imposes an often dramatic limitation on the efficiency with which the agent can estimate statistical properties of the environment, in comparison to estimation methods based on random sampling. We have, of course, examined only one robotic estimation strategy (random-walk sampling), one kind of agent, and two simple terrain models. We now describe briefly some ways in which this study could be extended.

For the sake of simplicity we modeled rugged terrain as having elevation values uniformly distributed over an interval. It might be of interest to repeat our analysis using a more realistic model, for example a Gaussian distribution, followed by weighted-average smoothing to produce smooth terrain. More generally, we could consider a Markov random field terrain model. We could also measure quantities other than elevation—for example, we could attempt to estimate the mean slope of the terrain, where the slope at a cell can be defined to be the magnitude of the gradient of the plane of least-squares fit to the seven-cell neighborhood centered at that cell.

Our agent had no facility for remote sensing and no positional or directional sense. For an agent with these or other sensory capabilities, it might be possible to devise a more efficient estimation strategy. As an example of this, consider an agent with a memory in which it records the locations that it has already visited. As we have seen, on rugged terrain a random walk is less efficient than a random sample, even though the values of the cells are independent and therefore uncorrelated. This is because a random walk can intersect itself. Self-intersection reduces the effective sample size, because some samples are repeated. An agent with (unlimited) memory would be able to follow a self-avoiding random walk, and we would therefore expect it to be more efficient than the agent studied here. In fact,

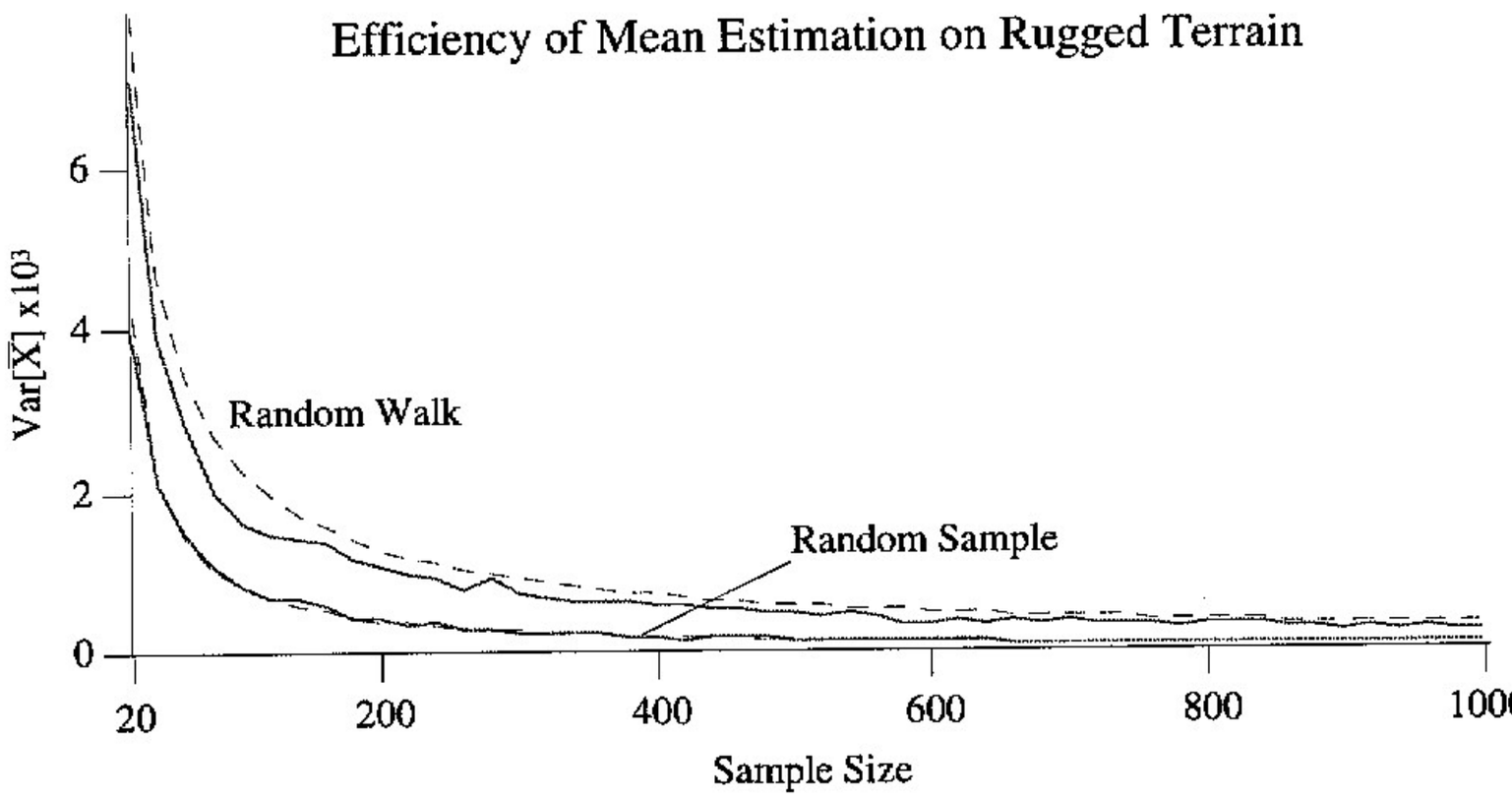


Figure 21: Variances of mean estimates of independent elevation values, for random sampling and random walk sampling. The dashed curves are the theoretical predictions.

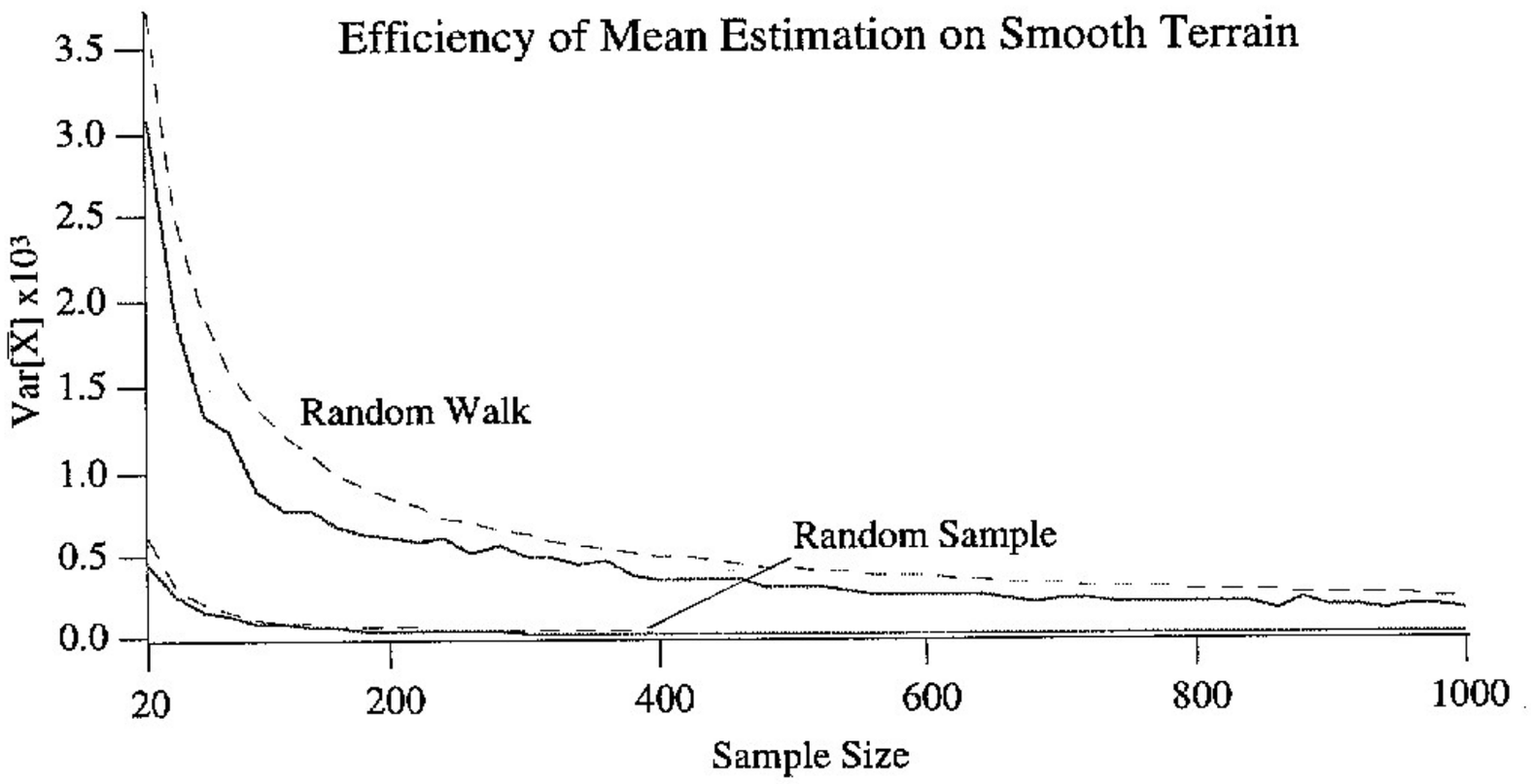


Figure 22: Variances of mean estimates of correlated elevation values, for random sampling and random walk sampling

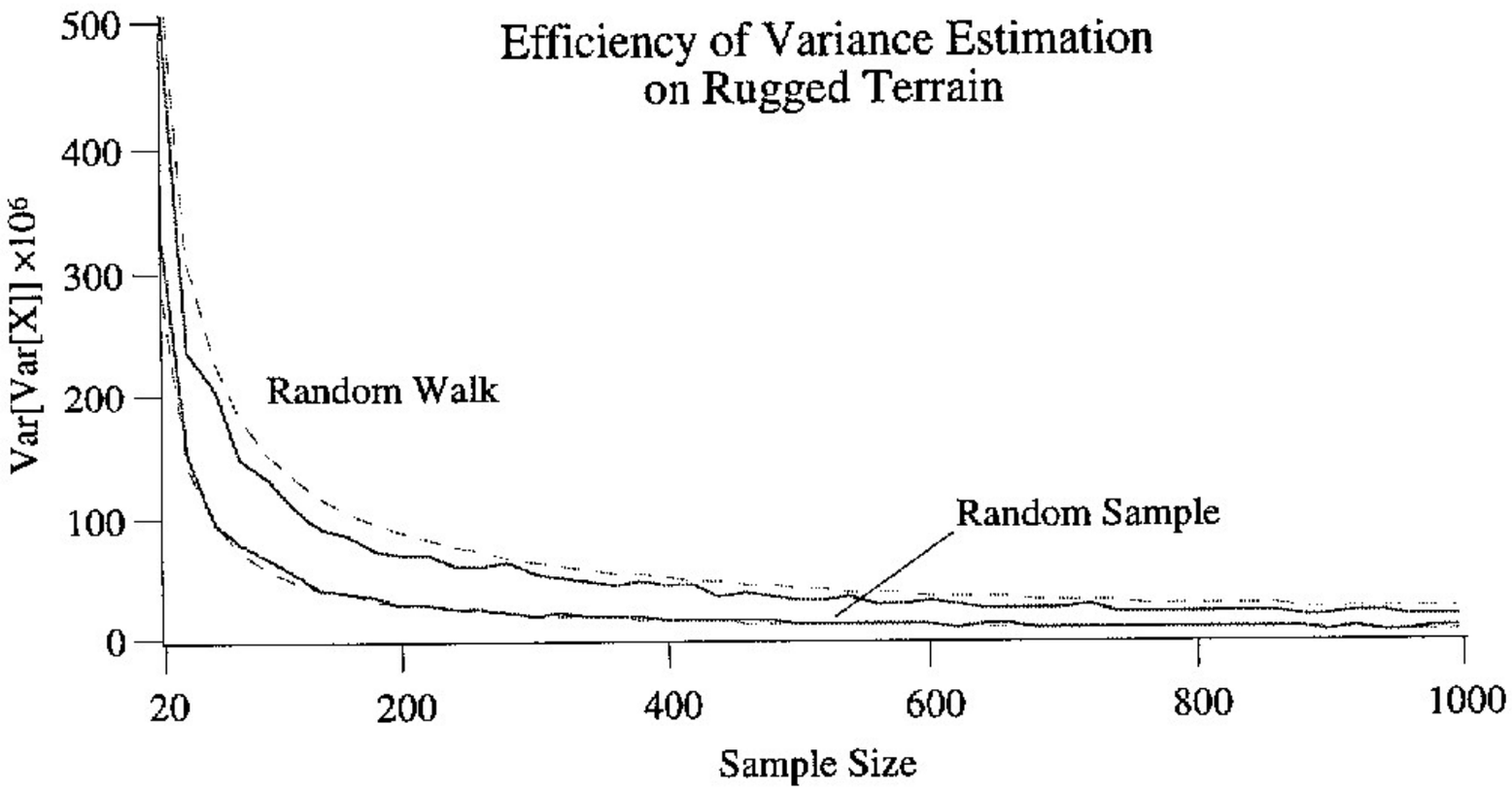


Figure 23: Variances of variance estimates of independent elevation values for random sampling and random walk sampling

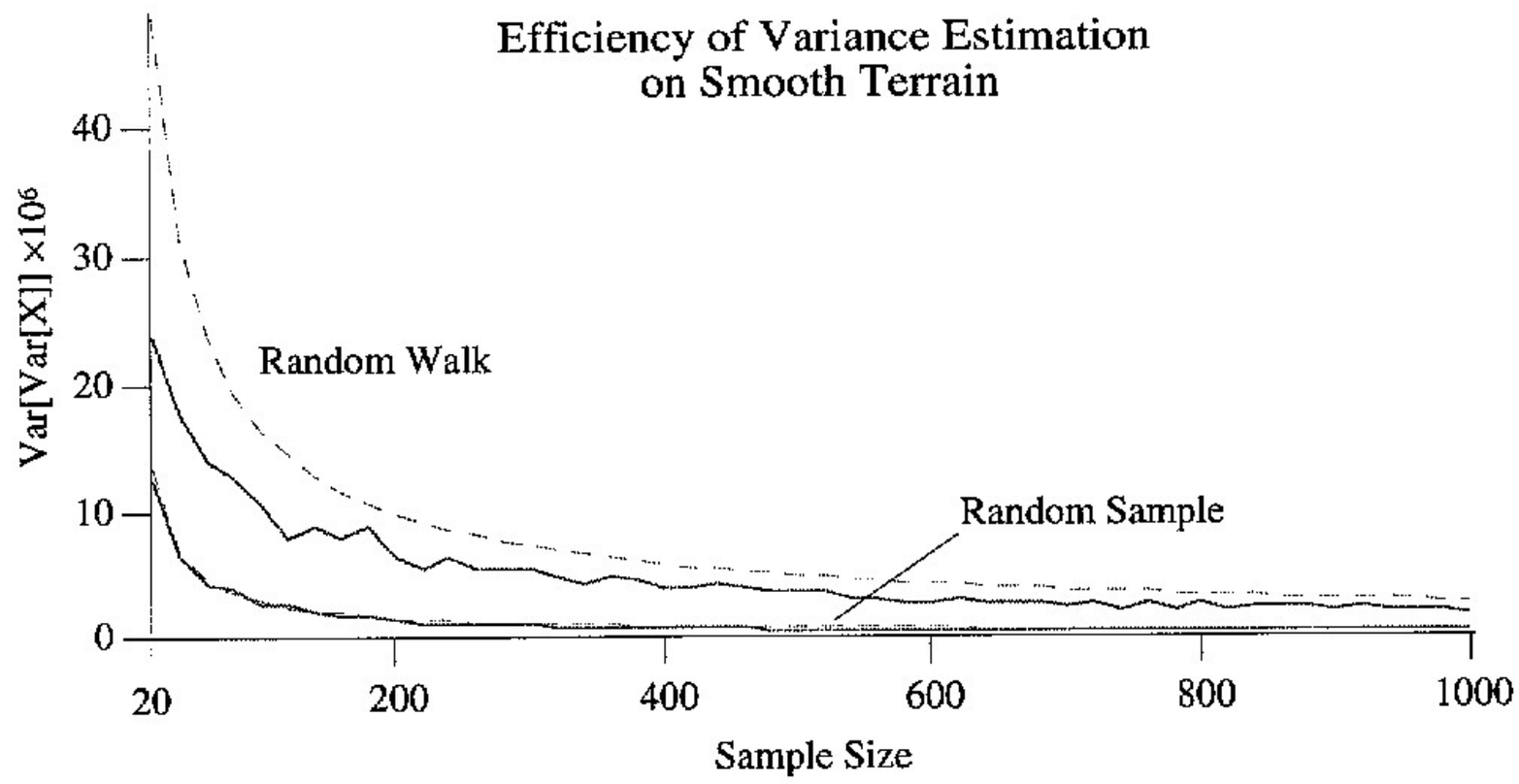


Figure 24: Variances of variance estimates of correlated elevation values, for random sampling and random walk sampling

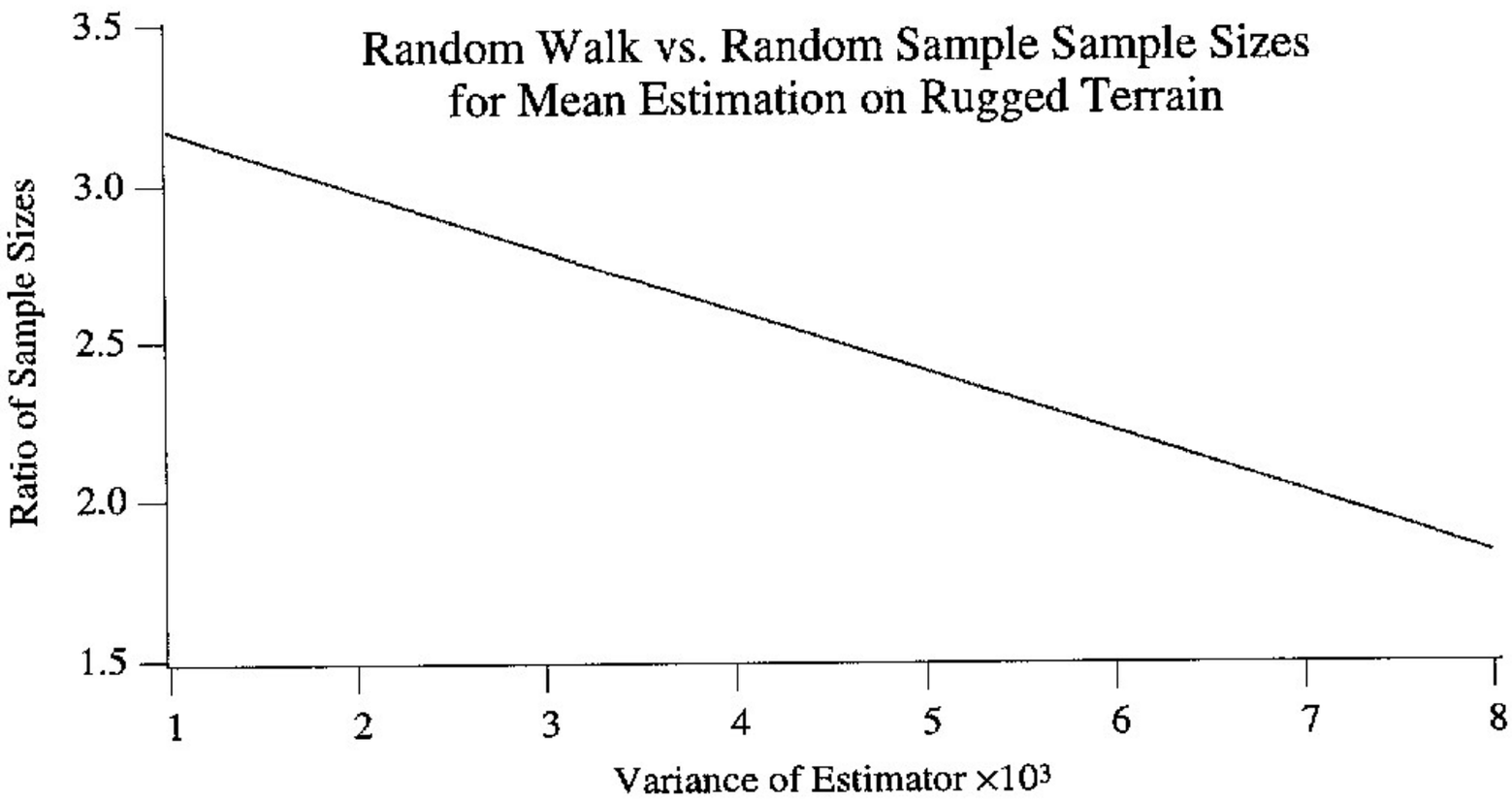


Figure 25: Theoretical ratio of the sample sizes required by random walk sampling and random sampling to achieve a given variance of thestimators, for mean elevation of rugged terrain

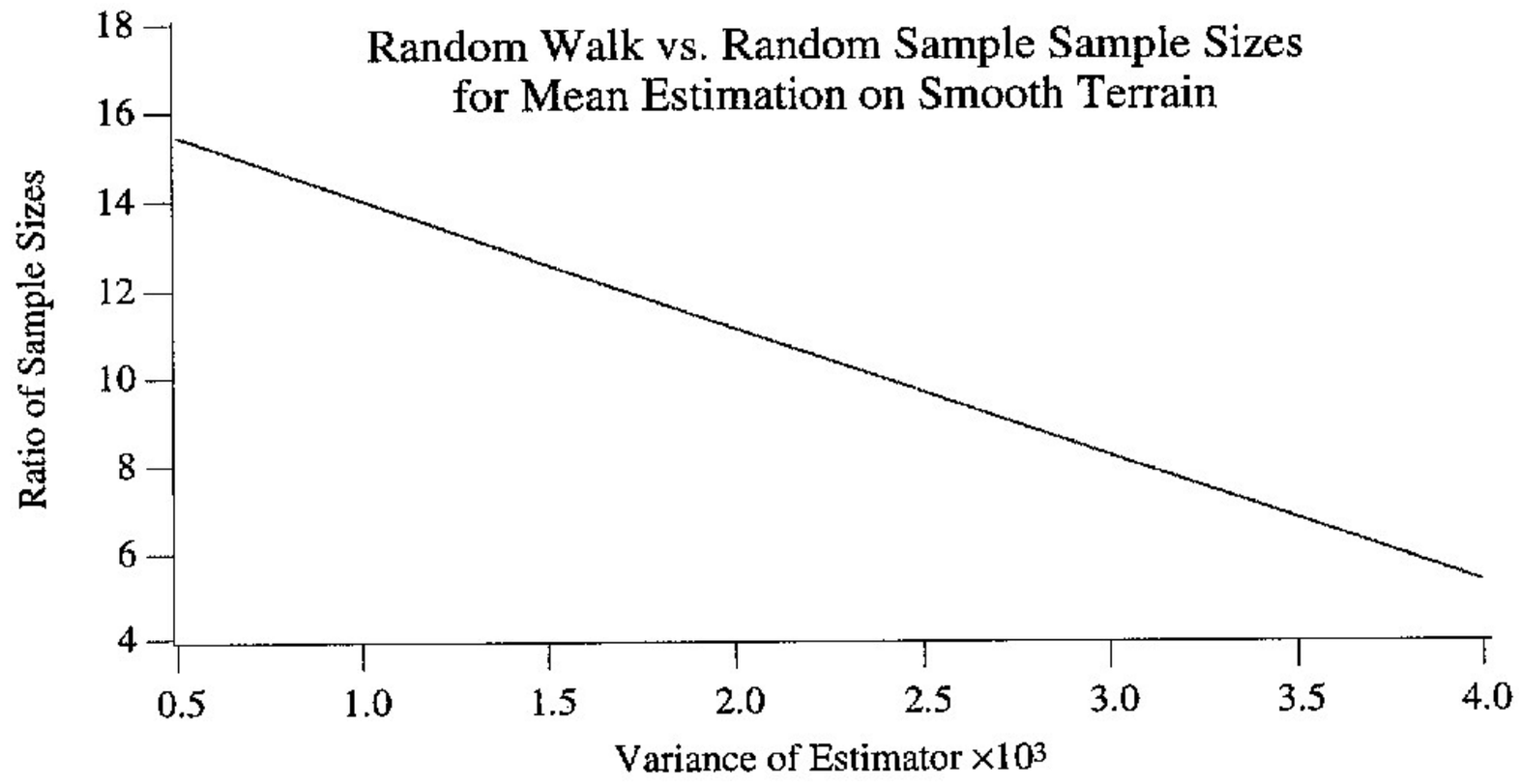


Figure 26: Theoretical ratio of the sample sizes required by random walk sampling and random sampling to achieve a given variance of thestimators, for mean elevation of smooth terrain

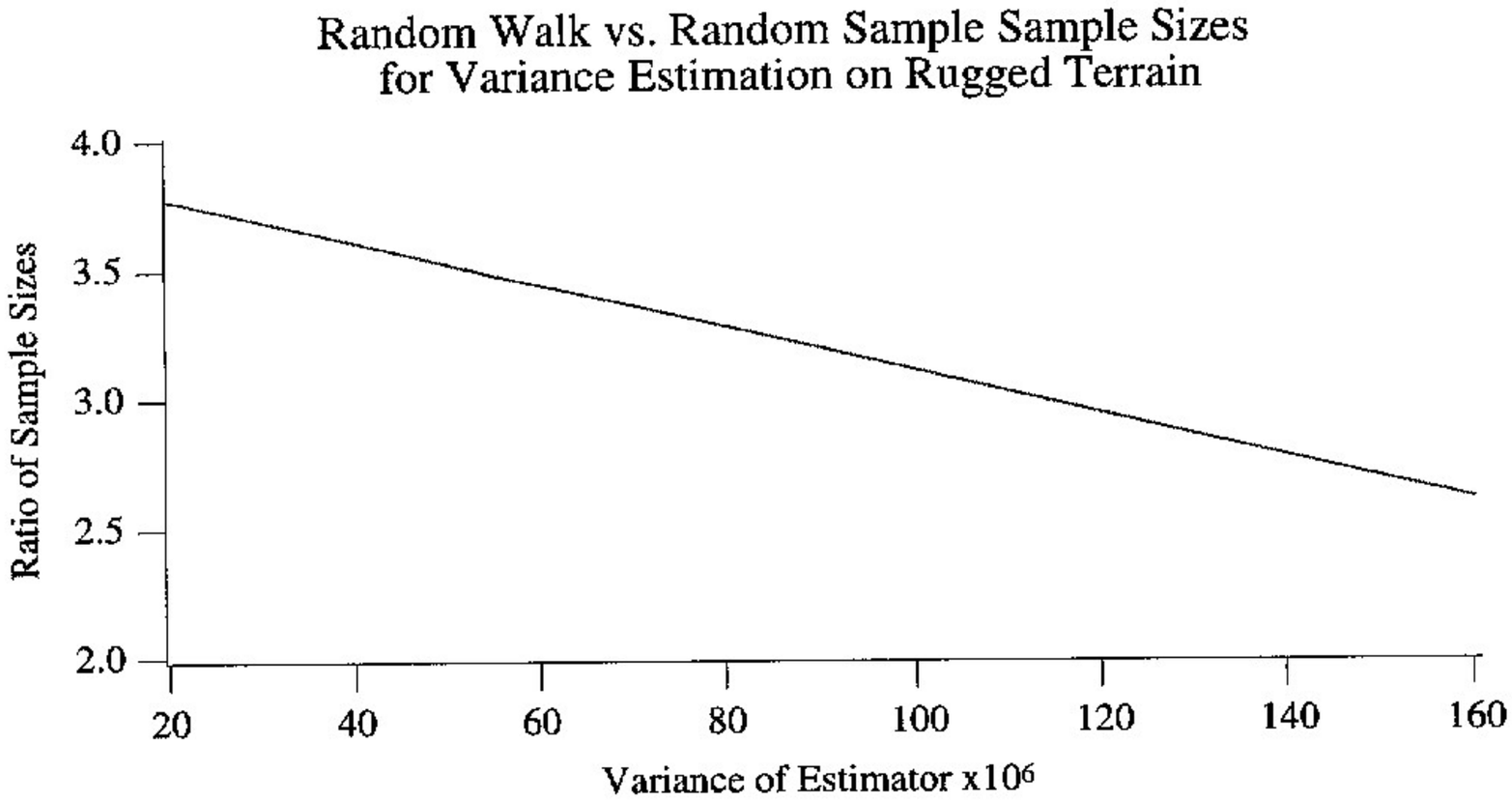


Figure 27: Theoretical ratio of the sample sizes required by random walk sampling and random sampling to achieve a given variance of thestimators, for elevation variance of rugged terrain

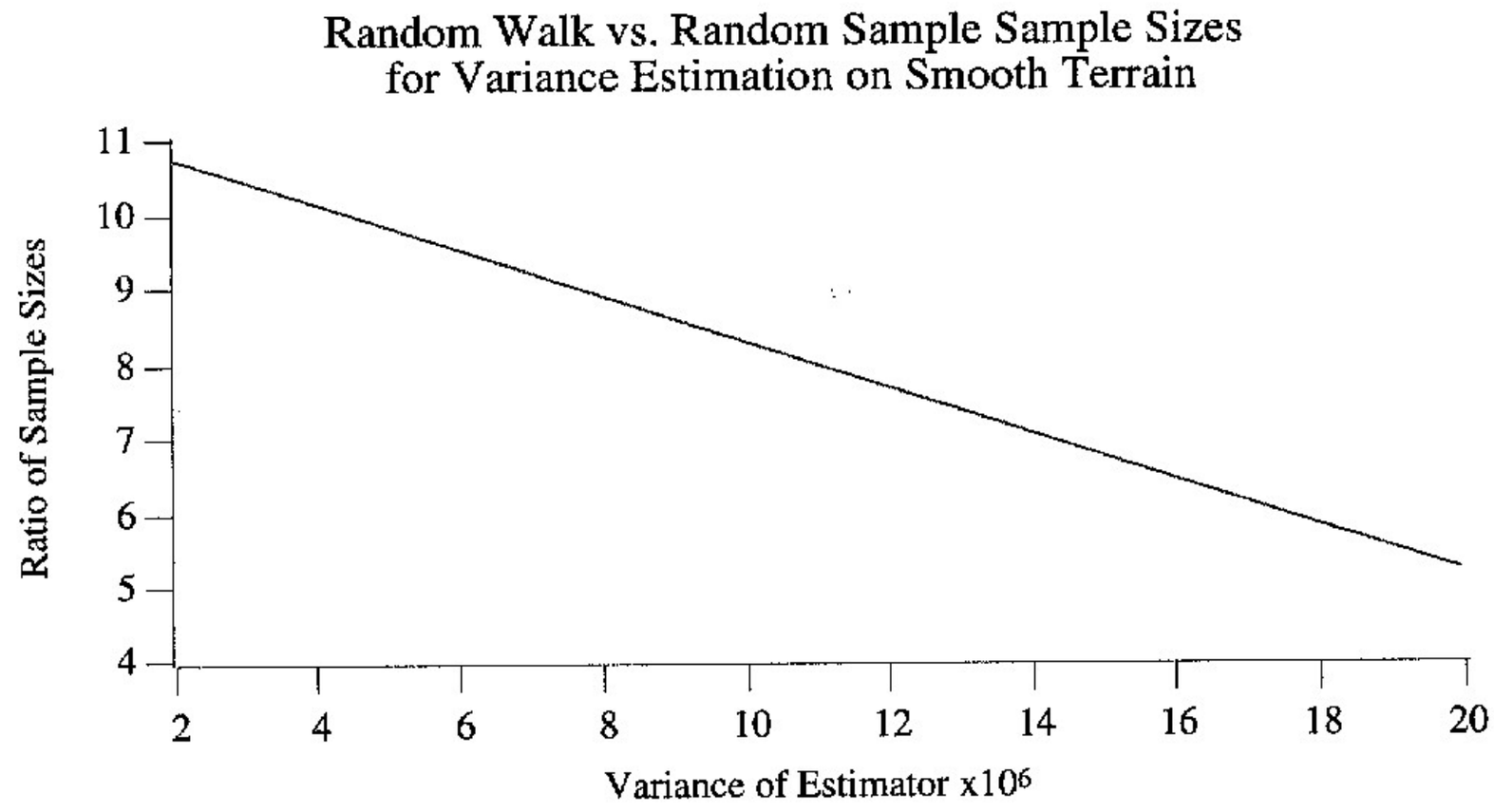


Figure 28: Theoretical ratio of the sample sizes required by random walk sampling and random sampling to achieve a given variance of thestimators, for elevation variance of smooth terrain

it is not difficult to see that self-avoiding random walk sampling and random sampling are equally efficient on rugged terrain. Unfortunately, the derivation of the probabilities for a self-avoiding random walk is much more complicated than for a non-self-avoiding walk—indeed, the study of self-avoiding random walks is largely an open problem — so the analysis of the smooth terrain case remains a topic for future research.

Since consecutive samples obtained along a connected path are usually correlated, it is not qualitatively surprising that random-walk sampling is less efficient than random sampling; but our quantitative analysis of the inefficiency should be useful to designers of robotic systems that have to sample their environments.

8 Visual Memories

The work on visual memories aims to provide an hierarchy of representations for the acquisition and storage of visual information. Levels in the hierarchy are described according to the type of structural information implicit in the memory organization. Visual tasks are then described in terms of the memory levels required to efficiently accomplish them. These memory levels are the Receptor, Neighborhood, Image, Place, View, Scene, and Map levels. These levels and the communications between them provide a framework for the implementation of a wide variety of visual tasks. We have developed a software library for the control of a robot, along with several active vision tasks.

The active vision system we are developing will use stereo vision to construct an internal representation of the entire scene where it is situated. It will use this scene representation to select a new viewpoint from which it can acquire more detailed information or view previously occluded regions of the scene. Currently, the system controls a robot head while providing the user with display window for controlling and viewing the various behaviors. The system makes only limited use of multiprocessing, and is not optimized for real-time operation.

The system runs as a set of related tasks, with several tasks cooperating to complete a "visual behavior." The system cycles through these tasks repeatedly, until ordered to stop. Visual behaviors which are currently implemented include aperture control, target selection, binocular fixation, stereo correspondence computation, depth-based segmentation, and the construction of a projective representation of the current view. Tasks still to be implemented include the fusing of multiple view representations, using a relaxation-based graph matching algorithm of Wilson and Hancock, and the linking of various scenes into an overall map of the agent's environment.

Aperture control is achieved using a simple receptor-level averaging and thresholding, along with open-loop pulsing of the iris control motors. When the mean intensity of the image lies below 50% or above 85% of the range of pixel values, the system will adjust the iris. Because of the wide range of permissible values, the uncontrolled effects of the camera automatic gain control mechanism, and the differences in the two cameras and their digitizing hardware, the "brightness constancy approximation" cannot be used directly on the image

data. Rather than rescaling the image intensities, we use stereo matching algorithms that are not sensitive to these differences.

A non-symmetric fixation is achieved by selecting a central window from the image of the dominant camera, and searching a horizontal band in the non-dominant image for the patch best correlated with the fixation target. If the fixation target contains insufficient structure (non-horizontal edges) for fixation, a nearby target is selected which does have this structure. Once this target is found, its image position is used to control the motion of the non-dominant camera, bringing its image center to the fixation point.

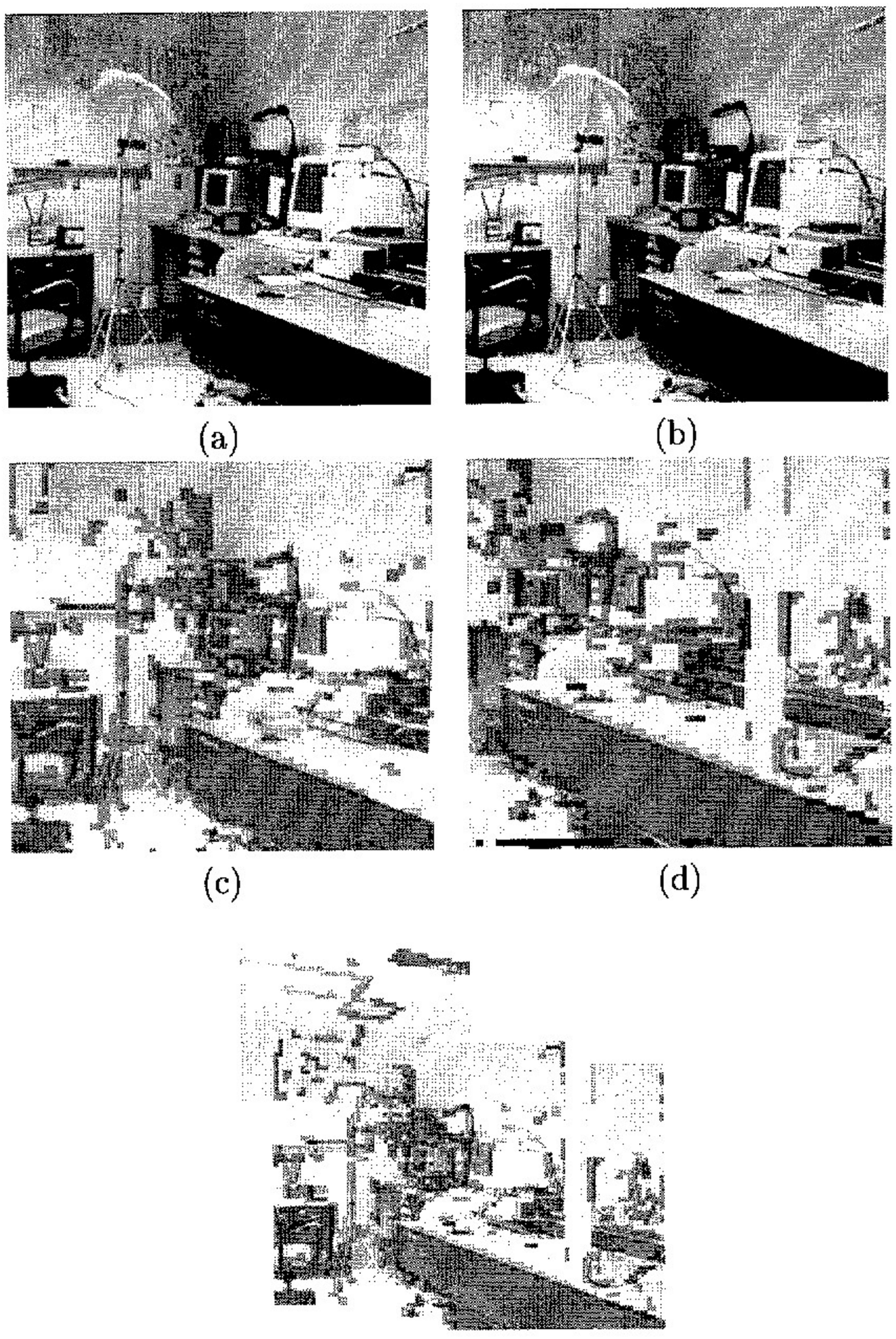
The fixation process is an image-level task, using the complete geometric layout of the image to define image windows, search regions, and motion parameters. Some component tasks, such as edge detection and window correlation, are neighborhood-level tasks. One of our goals is to move beyond the image level, and design representations for larger portions of the environment. The first of these, called the scene, represents the entire local environment of the agent. The scene is composed of a multitude of individual "places," each of which is a compact region of 3-D space. We call these places rather than objects, as the boundaries between them may not correspond to the physical connections of objects.

Using images from the fixating stereo head, portions of the scene with similar disparity are grouped together into places. Figures 29c-d show the labeling of single views, separating the image into regions of similar disparity, and retaining the ordinal structure between the different places in the view. Using this ordinal structure, we build a representation of the entire scene from various different viewpoints. Figure 29e shows an image of this representation from a single viewpoint, overlaid on a mosaic of the image regions with sufficient structure for stereo computations. These places are organized into a triangulated graph, with edges between them indicating the relative distance from the agent. These graphs will then be matched using a Bayesian graph matching algorithm optimized for triangulated graphs. The matching will allow the data from a new view of the scene to be incorporated into the previous representation. As more views are added to the scene, the system will learn coordinates for the individual places which are more and more uncorrelated (independent). The graph matching procedure and scene updating are currently being implemented.

9 Bisight Head Control

The Bisight control library interfaces with the C++ programs implementing the visual tasks, and with the PMAC motion controller in the VME bus. Motion commands can be sent to the head, specifying desired position, velocity, lens configuration, and synchronous or asynchronous operation. Position and velocity feedback from 8 motor axes is available to the user program. Safety features such as position and velocity limits are built into the routines running on the motion controller itself, where they are unlikely to be affected by failures in user-written software.

The user layer of the control library is a collection of C++ classes, each of which provides



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

Figure 29: An example of a scene representation. (a) and (b) represent two views of a scene, (c) and (d) represent the views with ordinal shape information (blue far, red near), and (e) represents the fusing of the two views into one representation of the scene containing information about ordinal shape. The mosaic was constructed without pixel correspondence but through the matching of intermediate shape maps.

a complete interface to the head. The user creates this head object by defining the subset of the eight motor axes which are to be enabled by the system. Thus users with less than eight degrees of freedom in the head, or applications which don't need all available axes, can still be served. Separate classes are available for simple position control, position/velocity control, and for controlling the head as a client from a remote machine. Each of these classes provides functions for powering, initializing, moving to a specified position, querying the current position, driving the iris motors, and shutting down.

The user layer is built upon a Bisight specific control layer. This layer contains the definitions of classes for the individual Bisight motor and lens axes. Single-axis homing procedures are defined and the default PMAC settings for each of the axes in the head are specified at this level.

The PMAC communications layer manages the interface between the PMAC residing on a VME bus and the Sun Sparc host. This layer initializes communication, defines proxy classes for several types of memory locations on the PMAC, and manages the downloading of control programs from the host to the PMAC. On the PMAC side, the communications layer controls synchronization, and the passing of new control parameters from the communications area to the motion programs. The communications program on the PMAC also computes the inverse kinematics on each motor axis, and makes the computed positions available to the host in the dual port RAM on the PMAC.

The motion control programs running on the PMAC board are Bisight specific. Seven motion control programs run concurrently and independently. The pan and tilt axes are controlled by a single program, as these two axes are mechanically coupled. In addition, the region of allowed motion depends on both the pan and tilt parameters. The motion program for the pan/tilt axes restricts the range of pan values allowed at high tilt angles, to avoid collision of the camera bodies with the base of the head. Paths which threaten to cross into one of these forbidden regions are redirected to the appropriate corner of the region, and allowed to continue from there. The motion programs for the other six axes simply enforce that their commanded positions lie within the allowed range.

10 Conclusions and Future Research

In conclusion we sumarize main achievements and describe some of the ideas relevant to future research on topics described earlier.

Advantages of Incorporating Learning into Vision Systems

We have given several illustrations of how a learning system can be used to help in handling vision problems for which algorithmic solutions are unknown or difficult to obtain. Specifically, we have studied the application of symbolic, neural network and multistrategy learning methods to the problems that involved interpreting outdoor scenes, recognizing objects in

cluttered environments, recognizing actions in video image sequences, and detecting and recognizing targets in SAR images. The first problem involved segmentation of an image into regions corresponding to grass, trees, etc.; since these categories do not have simple definitions, optimal algorithms for discriminating them cannot easily be defined. The other three problems involved classes of objects or actions that do not have simple geometrical definitions, but are defined only functionally: detecting blasting caps in x-ray images of luggage, detecting targets in SAR images and recognizing types of cutting (stabbing, chopping, slicing) in video image sequences. In each of these cases we have been able to design an appropriate representation space to make learning (and recognition) feasible.

Semantic Interpretation of Color Images of Outdoor Scenes

In Section 3 we showed that the MIST methodology can be very useful in applying machine learning methods to problems of natural scene interpretation. The results obtained so far have been promising, as they indicate a high level of performance accuracy even when only a single level of image transformation was applied. Particularly good results have been obtained with the AQ-NN method, which combines symbolic rule learning and a neural network.

There are several important advantages of this methodology. They include the ease of applying and testing diverse learning methods and approaches in a uniform manner, the potential for implementing advanced and complex learning processes, the use of background knowledge in learning and interpreting images, the suitability for parallel image learning and interpretation, and the ease of testing the performance of the methods.

Current research involves a systematic investigation of the methodology using different types of initial attributes and taking training and testing areas from images obtained under significantly different perceptual conditions.

Detection of Blasting Caps in X-ray Images of Luggage

In Section 4 we presented work in progress on the problem of recognizing blasting caps in x-ray images. In the first phase of a two-phase learning approach, low-intensity blobs were used as attention-catching devices. This bottom-up process was followed by a top-down recognition process in which a learned local model was matched to ribbon-shaped image regions surrounding a low-intensity blob. An analysis of the functional properties of blasting caps was used to design the representation space for learning, which combined intensity and geometric features. The experimental results suggest that learning can be used to acquire functional descriptions of objects. This is important for classes of objects for which geometric modeling is impractical.

Future work in this area will involve further automation of the feature extraction process and object labeling functions. In addition, other functional properties present in blasting

caps still require exploitation. An example is the presence of leg wires (see Figure 5). Unfortunately, the current image set is not of a resolution that allows for the detection of these functional properties. We hope to acquire additional images that will be better suited for this type of analysis.

Recognizing Actions in Video Image Sequences

Perceiving function from motion provides an understanding of the way an object is being used by an agent. To accomplish this we combined information about the shape of the object, its motion, and its relation to the actee (the object it is acting on). Assuming a decomposition of the object into primitive parts, we analyzed a part's motion relative to its principal axes. Primitive motions (translation and rotation relative to the principal axes of the object) were dominating factors in the analysis. We used a frame of reference relative to the actee. Once such a frame is established, it can have major implications for the functionality of an action.

Several image sequences were used to demonstrate our approach. In the three sequences shown in Section 5, motion was used to discriminate between three cutting actions: stabbing, chopping, and slicing. In other sequences, not shown here [DFR96], we used motion information to differentiate between two different functionalities of the same object: scooping and hitting with a shovel, and hammering and tightening with a wrench.

Natural extensions of this work include the analysis of more complex objects. Complexity can be expressed in terms of either the shapes of the parts or the way in which the parts are connected. An interesting area is the analysis of articulated objects. The different types of connections between the parts constrain the possible relative motions of the parts. A pair of pliers or a pair of scissors is a simple case, with only a single articulated connection (one degree of freedom in the relative motion of the parts).

Recognizing Targets in SAR Images

Our methodology combines a multistrategy learning approach that integrates current ATR methods for image filtering and target detection with advanced methods for symbolic and neural net learning, and is thereby able to utilize their complementary advantages to achieve high correctness and speed of recognition. The Multi-level Image Sampling and Transformation Architecture (MIST) is used to combine a variety of methods and image transformations for target learning and detection.

Acknowledgements

This research was supported in part by the Advanced Research Projects Agency under grants F49620-92-J-0549 and F49620-95-1-0462 administered by the Air Force Office of Scientific

Research, in part by the Air Force Office of Scientific Research under grant F49620-93-1-0039, in part by the Advanced Research Projects Agency under grant No. N00014-91-J-1854 administered by the Office of Naval Research, in part by the Office of Naval Research under grant N00014-91-J-1351, and in part by the National Science Foundation under grants DMI-9496192 and IRI-9510644.

References

[BMP94] Bala, J.W., Michalski, R.S., and Pachowicz, P.W., "Progress on vision through learning at George Mason University", in *Proc. ARPA Image Understanding Workshop*, 191–207, 1994.

[BMW92] Bala, J., Michalski, R.S., and Wnek, J., "The principal axes method for constructive induction", in *Proc. International Conference on Machine Learning*, D. Sleeman and P. Edwards (Eds.), Aberdeen, Scotland, 1992.

[BMW93] Bala, J., Michalski, R.S., and Wnek, J., "The PRAX approach to learning a large number of texture concepts", in *Proc. Machine Learning in Computer Vision: What, Why and How?, AAAI Fall Symposium on Machine Learning in Computer Vision*, 1993.

[Bie85] Biederman, I., "Human image understanding: Recent research and a theory", *Computer Vision, Graphics and Image Processing*, **32:**29–73, 1985.

[BWMK93] Bloedorn, E., Wnek, J., Michalski, R.S., and Kaufman, K., "AQ17 — A multistrategy learning system: The method and user's guide", *Reports of the Machine Learning and Inference Laboratory*, MLI 93-12, George Mason University, Fairfax, VA, 1993.

[BoB94] Bogoni, L. and Bajcsy, R., "Active investigation of functionality", in *Proc. CVPR Workshop on Visual Behaviors*, June 1994.

[BABC84] Brady, M., Agre, P.E., Braunegg, D.J., and Connell, J., II, "The mechanic's mate" in *Proc. European Conference on Artificial Intelligence*, 79–94, 1984.

[Cha89] Channic, T., "TEXPERT: An application of machine learning to texture recognition", *Reports of the Machine Learning and Inference Laboratory*, MLI 89-27, George Mason University, Fairfax, VA, 1989.

[ChD94] Cho, K. and Dunn, S.M., "Learning shape classes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16:**882–888, 1994.

[CrK91] Cromwell, R.L. and Kak, A.C., "Automatic generation of object class descriptions using symbolic learning techniques", in *Proc. National Conference on Artificial Intelligence*, 710–717, 1991.

[CoB87] Connell, J.H. and Brady, M., "Generating and generalizing models of visual objects", *Artificial Intelligence*, **34:**159–183, 1987.

[CNR96] Cucka, P., Netanyahu, N., and Rosenfeld, A., "Learning in navigation: goal finding in graphs", in *Intl. J. Pattern Recognition and Artificial Intelligence*, **10:**429–446, 1996.

[CNR97] Cucka, P., Netanyahu, N., and Rosenfeld, A., "Robotic estimation", Report CAR-TR-842, University of Maryland, College Park, MD, 1996.

[Dan88] Dance, D.R, "Diagnostic radiology with x-rays", in *The Physics of Medical Imaging*, S. Webb (Ed.), 20–73, IOP Publishing, Philadelphia, PA, 1988.

[DFR96] Duric, Z., Fayman, E., and Rivlin, E., "Function from motion", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 579–591, 1996.

[DRR96] Duric, Z., Rivlin, E., and Rosenfeld, A., "Learning an object's function by observing the object in action", in *Proc. ARPA Image Understanding Workshop*, 1437–1445, 1996.

[DuB94] Dutta, R. and Bhanu, B., "A learning system for consolidated recognition and motion analysis", in *Proc. ARPA Image Understanding Workshop*, 773–776, 1994.

[Fah88] Fahlman, S.E., "An empirical study of learning speed in back-propagation networks", Report CMU-CS-88-182, Carnegie-Mellon University, Pittsburgh, PA, 1988.

[FeB96] Ferryman, A. and Bhanu, B., "A Bayesian approach for the segmentation of SAR images using dynamically selected neigborhoods", in *Proc. ARPA Image Understanding Workshop*, 891–895, 1996.

[FiS88] Fischler, M.A. and Strat, T.M., "Recognizing trees, bushes, rocks and rivers", in *Proc. AAAI Spring Symposium Series: Physical and Biological Approaches to Computational Vision*, 62–64, 1988.

[FrN71] Freeman, P. and Newell, A., "A model for functional reasoning in design", in *Proc. International Joint Conference on Artificial Intelligence*, 621–640, 1971.

[GESB94] Green, K., Eggert, D., Stark, L., and Bowyer, K., "Generic recognition of articulated objects by reasoning about functionality", in *Proc. AAAI Workshop on Represening and Reasoning about Device Function*, 56–64, 1994.

[GrP96] Grimson, W.E.L. and Poggio, T., "Progress in image understanding at MIT", in *Proc. ARPA Image Understanding Workshop*, 65–74, 1996.

[HoB94] Howard, C.G. and Bock, P., "Using a hierarchical approach to avoid over-fitting in early vision", in *Proc. International Conference on Pattern Recognition*, 826–829, 1994.

[MDMR96] Maloof, M.A., Duric, Z., Michalski, R.S., and Rosenfeld, A., "Recognizing blasting caps in x-ray images", in *Proc. ARPA Image Understandning Workshop*, 1257–1261, 1996.

[MaM94] Maloof, M.A. and Michalski, R.S., "Learning descriptions of 2D blob-like shapes for object recognition in x-ray images: An initial study", *Reports of the Machine Learning and Inference Laboratory*, MLI 94-4, George Mason University, Fairfax, VA, 1994.

[MaM96] Maloof, M.A. and Michalski, R.S, "Learning descriptions of shape for object recognition in x-ray images", *Expert Systems with Applications*, in press, 1996.

[Mic72] Michalski, R.S., "A variable-valued logic system as applied to picture description and recognition", in F. Nake and A. Rosenfeld (Eds.), *Graphic Languages*, North-Holland, Amsterdam, 21–47, 1972.

[Mic73] Michalski, R.S., "AQVAL/1: Computer implementation of a variable-valued logic system VL_1 and examples of its application to pattern recognition", in *Proc. International Joint Conference on Pattern Recognition*, 3–17, 1973.

[Mic80] Michalski, R.S., "Pattern recognition as rule-guided inductive inference", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **2:**349–361, 1980.

[MBP93] Michalski, R.S., Bala, J.W, and Pachowicz, P.W., "Progress on vision through learning at George Mason University", in *Proc. ARPA Image Understanding Workshop*, 191–207, 1993.

[MMHL86] Michalski, R.S., Mozetic, I., Hong, J., and Lavrac, N., "The multipurpose incremental learning system AQ15 and its testing application to three medical domains", in *Proc. National Conference on Artificial Intelligence*, 1041–1045, 1986.

[MRA94] Michalski, R.S., Rosenfeld, A., and Aloimonos, Y., "Machine vision and learning: Research issues and directions — A report on the NSF/ARPA Workshop on Learning and Vision, Harpers Ferry, WV, October 15-17, 1992", *Reports of the Machine Learning and Inference Laboratory*, MLI 94-6, George Mason University, Fairfax, VA, 1994.

[MRADMZ96] Michalski, R.S., Rosenfeld, A., Aloimonos, Y., Duric, Z., Maloof, M.A., and Zhang, Q., "Progress on vision through learning: A collaborative effort of George Mason University and the University of Maryland", in *Proc. ARPA Image Understandning Workshop*, 177–187, 1996.

[MZMB96] Michalski, R.S., Zhang, Q., Maloof, M.A., and Bloedorn, E., "The multi-level image sampling and transformation methodology and its application to natural scene interpretation", in *Proc. ARPA Image Understandning Workshop*, 1473–1479, 1996.

[PaB91] Pachowicz, P.W. and Bala, J.W., "Texture recognition through machine learning and concept optimization", *Reports of the Machine Learning and Inference Laboratory*, MLI 95-4, George Mason University, Fairfax, VA, 1991.

[Pom91] Pomerleau, D.A., "Efficient training of artificial neural networks for autonomous navigation", *Neural Computation*, **3:**88–97, 1991.

[RDR95] Rivlin, E., Dickinson, S.J., and Rosenfeld, A., "Recognition by functional parts", *Computer Vision and Image Understanding*, **62:**164–176, 1995.

[RRP93] Rivlin, E., Rosenfeld, A., and Perlis, D., "Recognition of object functionality in goal-directed robotics", in *Proc. AAAI Workshop on Reasoning about Function*, 126–130, 1993.

[RBP96] Romano, R., Beymer, D., and Poggio, T., "Face verification for real-time applications", in *Proc. ARPA Image Understanding Workshop*, 747–756, 1996.

[RBK96] Rowley, H.A., Baluja, S., and Kanade, T., "Neural network-based face detection", in *Proc. ARPA Image Understanding Workshop*, 725–735, 1996.

[Seg94] Segen, J., "GEST: A learning computer vision system that recognizes hand gestures", in R.S. Michalski and G. Tecuci, (Eds.), *Machine Learning: A Multistrategy Approach*, Vol. IV, 621–634, Morgan Kaufmann, San Francisco, CA, 1994.

[She83] Shepherd, B.A., "An appraisal of a decision tree approach to image classification", in *Proc. International Joint Conference on Artificial Intelligence*, 473-475, 1983.

[SoB83] Solina, S. and Bajcsy, R., "Shape and function", in *Proc. SPIE Conference on Intelligent Robots and Computer Vision*, Vol. 726, 284–291, 1983.

[StB91a] Stark, L. and Bowyer, K., "Achieving generalized object recognition through reasoning about association of function to structure", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13:**1097–1104, 1991.

[StB91b] Stark, L. and Bowyer, K., "Generic recognition through qualitative reasoning about 3-D shape and object function", in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 251–256, 1991.

[StB92] Stark, L. and Bowyer, K., "Indexing function-based categories for generic recognition", in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 795–797, 1992.

[SHGB93] Stark, L., Hoover, A., Goldgof, D., and Bowyer, K., "Function-based recognition from incomplete knowledge of shape", in *Proc. IEEE Workshop on Qualitative Vision*, 11–22, 1993.

[StF91] Strat, T. and Fischler, M., "Natural object recognition: A theoretical framework and its implementation", in *Proc. International Joint Conference on Artificial Intelligence*, 1991.

[StF95] Strat, T. and Fischler, M., "The role of context in computer vision", in *Proc. ICCV Workshop on Context-Based Vision*, 1995.

[SuP94] Sung, K. and Poggio, T., "Example-based learning for view-based human face detection", in *Proc. ARPA Image Understanding Workshop*, 747–756, 1994.

[WeK92] Weiss, S.M., and Kulikowski, C.A., *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*, Morgan Kaufmann, San Mateo, CA, 1992.

[Win84] Winston, P.H., *Artificial Intelligence*, 2nd ed., Addison-Wesley, Reading, MA, 1984.

[WBKL83] Winston, P.H., Binford, T.O., Katz, B., and Lowry, M., "Learning physical descriptions from functional descriptions, examples, and precedents", in *Proc. National Conference on Artificial Intelligence*, 433–439, 1983.

[WKBM95] Wnek, J., Kaufman, K., Bloedorn, E., and Michalski, R.S., "Inductive learning system AQ15c: The method and user's guide", *Reports of the Machine Learning and Inference Laboratory*, MLI 95-4, George Mason University, Fairfax, VA, 1995.

[WCHBS95] Woods, K., Cook, D., Hall, L., Bowyer, K., and Stark, L, "Learning membership functions in a function-based object recognition system", *Journal of Artificial Intelligence Research*, **3:**187–222, 1995.

[ZhB96] Zheng, Y. and Bhanu, B., "Performance improvement by input adaptation using modified Hebbian learning", in *Proc. ARPA Image Understanding Workshop*, 1381–1387, 1996.

[Zur92] Zurada, J.M., *Introduction to Artificial Neural Systems*, West Publishing, St. Paul, MN, 1992.