

# Detection and Recognition of 3-D Targets in Panchromatic and in Synthetic Aperture Radar Imagery Using a Map-Seeking Circuit Algorithm

*Patricia K. Murphy, Pedro A. Rodriguez, and Cameron K. Peterson*

A large, stylized, brown letter 'A' is positioned on the left side of the page, partially overlapping the main text block.

*3-D target detection and recognition algorithm, based on the biologically inspired map-seeking circuit (MSC), is implemented to efficiently solve the template-matching problem in synthetic aperture radar (SAR) and panchromatic grayscale imagery. Given a 3-D model of a target, this algorithm locates the target in a 2-D image and determines its pose (i.e., viewing angles, scale, and spatial translations). A key aspect of the MSC is the simultaneous forward transformation of the model to match the image coupled with a backward path to make the image match the model. The efficiency of the algorithm is a result of the decomposition of the n-dimensional pose transformation space into a series of one-dimensional searches for each of the transformation parameters. Although originally designed for panchromatic electro-optical imagery, we demonstrate that the MSC architecture can also be successfully applied to SAR by simply changing the feature-extraction preprocessing. Additionally, we introduce modifications to the MSC algorithm that increase the speed of detection and allow efficient classification when multiple targets are present in the same image. We present promising results after applying our algorithm to challenging real-world panchromatic electro-optical and SAR imagery.*

## INTRODUCTION

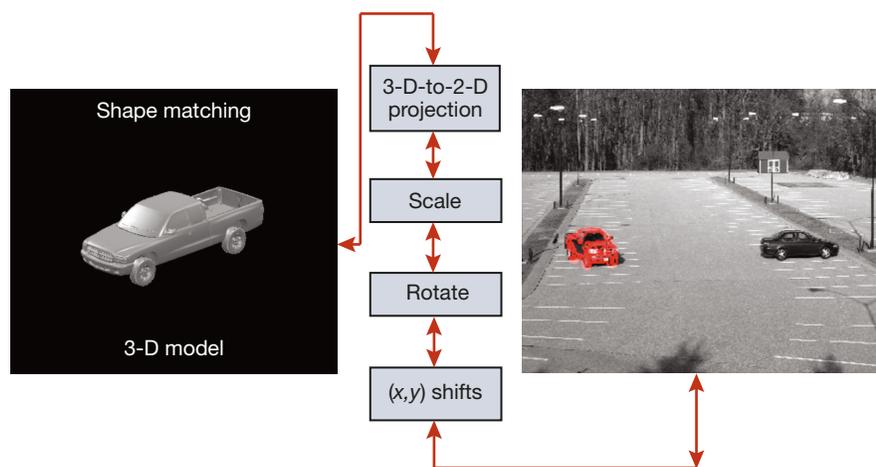
High-resolution panchromatic electro-optical (EO) and synthetic aperture radar (SAR) sensors are two of the most important intelligence, surveillance, and recon-

naissance resources. SAR is a valuable intelligence asset because of its proven operational capability to image targets both day and night in any type of weather and to

provide precision geo-location accuracy. EO sensors are more prevalent operationally and provide high-resolution images that can be used to identify and recognize targets. Automated exploitation of EO and SAR imagery remains a challenge because both modalities suffer from the difficulties that are typically encountered in automated processing of high-resolution imagery—for example, other man-made and natural objects in the scene representing clutter, foreground objects that may partially occlude the target of interest, illumination and shading variations, and large data volumes. Existing conventional target-recognition approaches in both EO and SAR include template matching and feature-based classification techniques. Template-matching algorithms that exhaustively search the target pose state space can be used to detect and recognize targets given a 3-D model or 2-D target templates. The high dimensionality of this state space limits the use of these algorithms in many time-critical applications, even though the imagery may be readily available and of great benefit. In this article, we investigate a physical-model-based template-matching approach called the map-seeking circuit (MSC) and show its ability to efficiently detect and identify targets with arbitrary viewing geometries in both EO and SAR sensor modalities.

The MSC is a new, biologically inspired algorithm developed by David Arathorn at the Center for Computational Biology at Montana State University.<sup>1,2</sup> It is a model-based approach that assumes that the correspondence between a model and an observation of it can be represented by a decomposition of invertible transformations. For example, MSC can be used to efficiently search a high-dimensional space for the best set of transformations that map a 3-D object to an observed 2-D representation in a panchromatic image as diagrammed in Fig. 1. In this article, we introduce a modified MSC algorithm, the multistage-MSC (MS-MSC), which incorporates new processing stages including initial candidate detection, parallel MSC processing of candidates, and calculation of goodness-of-fit (GOF) metrics. This modified algorithm increases the speed of detection and allows efficient classification when multiple targets are present in a single image. Our architecture also enables simultaneous search for different target models.

The robust nature of the MSC makes it applicable to any process that can be decomposed into a discrete



**Figure 1.** Example of MSC transformational layers with an input 3-D truck model and output results projected into the panchromatic input image. (Reproduced from Ref. 6, © IEEE 2009.)

set of invertible transformations. In this article, we use the MS-MSC algorithm to solve the aforementioned template-matching problem in both SAR and panchromatic imagery. Given the 3-D model of a target, this algorithm locates the vehicle in a 2-D image and determines its pose (i.e., viewing angles, scale, and spatial translations). Although the basic algorithmic architecture is similar for both types of imagery, there are critical preprocessing steps that condition the raw imagery for use in the MSC that are sensor-modality specific. We also introduce postprocessing to provide a confidence metric of the MSC result based on various GOF computations.

In this article, we introduce the MSC algorithm and describe the modifications made that resulted in our MS-MSC approach. Preprocessing techniques used to represent image data within the MSC architecture are discussed, and several results on real imagery are presented. MSC was developed as a neuromimetic algorithm initially designed for EO imagery; we extend and demonstrate the use of the MSC architecture to perform target detection and recognition in SAR. To our knowledge, this is the first time the MSC algorithm has been applied to SAR imagery. Conclusions and plans for future research are summarized in the final section.

## MSC ALGORITHM

In the following sections, we present both the original MSC algorithm introduced by Arathorn and the modified MS-MSC algorithm we developed and implemented.

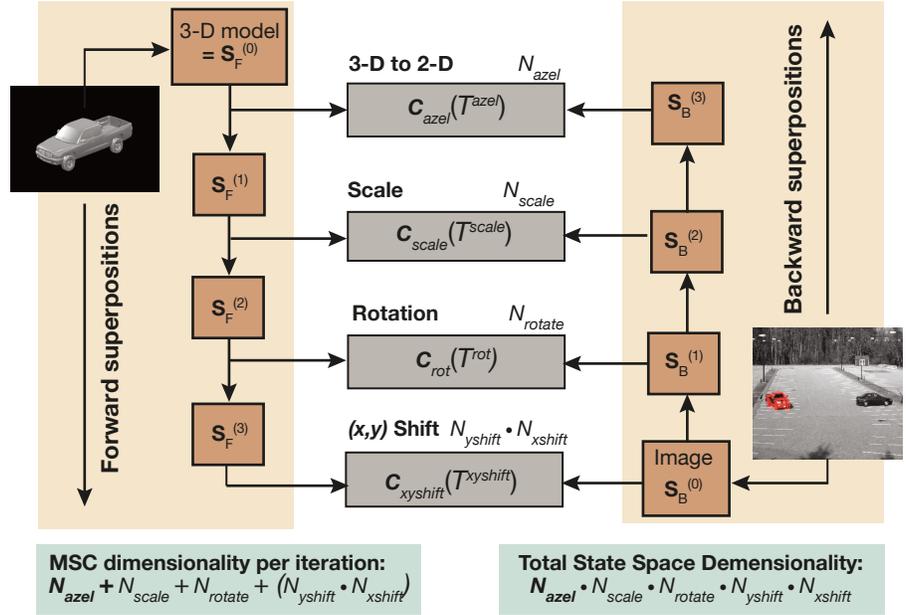
### Baseline MSC Algorithm

Detecting and identifying target objects with unknown location, scale, in-plane (image) rotation,

azimuth, and elevation in the presence of clutter and confusers requires consideration of a very-high-dimensional projection space, frequently on the order of  $10^{10}$  possible transformations. Exhaustive search of this high-dimensional space is very computationally expensive and generally precludes near-real-time applications. Techniques that transform the data into an invariant domain do provide some dimensionality reduction but may require significant preprocessing and often do not obviate the need to consider a high-dimensional space. When the template is another 2-D image instead of a 3-D model, there are powerful alternate approaches that can be used. For example, variations on Lowe’s Scale Invariant Feature Transform (SIFT) algorithm have been proven to be robust in the presence of clutter, scale, and planar rotation.<sup>3</sup> The original SIFT algorithm had limited invariance to 3-D-to-2-D projection; a recent extension called Affine-SIFT<sup>4</sup> builds in additional 3-D-to-2-D projection invariance but requires simulated/training data and still does not provide a fully 3-D capability.

The MSC is inspired by neurobiological evidence that both reciprocal forward and backward transformations are required to perform visual cognition (i.e., forward mapping of the 3-D model into the imagery coupled with backward mapping of the 2-D imagery to fit the model) (see Fig. 2). The MSC uses an iterative process of simultaneous forward/backward testing of multiple superposed transformation hypotheses. Hypotheses are competed to quickly converge on the correct transformation mapping to identify the target in the scene even when other confusers and clutter are present. One of the most interesting aspects of the MSC is that target segmentation is an *emergent* property of the algorithm; it is not a precondition as in many conventional machine-vision approaches.

Although the physical process behind the projection of the 3-D model into EO imagery is completely different from that required for SAR, the MSC layers used to compete the 3-D-to-2-D projections are similar for both modalities. The main differences are in the preprocessing of the input images and model templates. In this section, we describe the baseline MSC algorithm using EO nomenclature and examples; more detail about the SAR implementation is provided in a later section.



**Figure 2.** Four-layer MSC block diagram showing the forward and backward paths that project the 3-D model into the image and vice versa. Here  $K = 1 =$  azimuth/elevation,  $K = 2 =$  scale,  $K = 3 =$  rotation, and  $K = 4 =$  (x,y) shift. The forward and backward superpositions (Eq. 6) are computed on the left and right paths, respectively, and the correspondences,  $c_k(T^{(K)})$ , for each layer are computed in the central boxes (Eq. 7). (Reproduced from Ref. 6, © IEEE 2009.)

The MSC algorithm attempts to efficiently solve the correspondence problem to find the best set of transformation parameters to match a given 3-D model,  $\mathbf{M}$ , to the observed 2-D image,  $\mathbf{I}$ . Adopting the notation introduced in Refs. 5 and 6,

$$c(T) = \langle T(\mathbf{M}), \mathbf{I} \rangle \quad (1)$$

defines the correspondence  $c$  for a particular set of transformations  $T$  to be the inner product of the image,  $\mathbf{I}$ , with the transformed model,  $T(\mathbf{M})$ . The MSC can be generalized to problems that can be described as a series of  $L$ -decomposable transformations:

$$T = T_{i_L}^{(L)} \circ \dots \circ T_{i_2}^{(2)} \circ T_{i_1}^{(1)}, \quad (2)$$

where the subscript  $i_k$  identifies a specific parameter value in the transformation layer  $T^{(K)}$  (e.g., an in-plane rotation of  $2^\circ$ ). Typical transformation layers include 3-D-to-2-D, scaling, in-plane rotation, and translation.

The MSC relies fundamentally on the so-called Ordering Property of Superpositions, which states the following for sparse vectors ( $\mathbf{v}$ ):

$$\begin{aligned} \text{For superposition } \mathbf{S} &= \sum_{i=1}^n \mathbf{v}_i \\ \text{and for } k \in [1, \dots, n] \text{ and } j \notin [1, \dots, n], \quad (3) \\ \text{then } \Pr\{\langle \mathbf{S}, \mathbf{v}_k \rangle > \langle \mathbf{S}, \mathbf{v}_j \rangle\} &\geq 0.5. \end{aligned}$$

The rationale for this property can be found in Ref. 2. Basically, this property states if a given transformation

is part of the superposition, the response will on average be higher than that obtained with one that is not—i.e., the probability ( $\Pr\{\cdot\}$ ) is greater than 0.5 that the dot product of the superposition with a vector that is part of the sum is greater than the dot product with a vector that is not part of the sum. This is very dependent on the sparseness precondition.

A critical aspect of the MSC is the forward and backward paths that project the model into the image (forward path) and invert process to project the image into the model (backward path). Instead of exhaustively searching the entire high-dimensional transform parameter space, the MSC searches a superposition hyperspace defined by the inner product of forward and backward superpositions. A superposition is defined as a weighted linear combination of all possible transformations for a given transformation layer (e.g., spatial translations). The weights are optimized via nonlinear inhibitor or competition functions.

The weight vectors,  $\mathbf{g}^{(K)}$ , are the coefficients associated with the transformations for a given layer  $K$ ,  $T_{i_K}^{(K)}$  for  $i_K = 1, 2, \dots, N^{(K)}$ , where  $N^{(K)}$  is the dimensionality of the state space for layer  $K$  (e.g., for an azimuth layer examining transformations every  $5^\circ$  over a range of  $360^\circ$ ,  $N^{(K)} = 72$ ). Initially, the  $\mathbf{g}$  vector for each layer is set to unity—equivalent to an uninformed prior assumption. As the MSC iterates, the  $\mathbf{g}$  vectors are updated on the basis of the computed correspondences until they converge to a single nonzero value for each  $\mathbf{g}^{(K)}$  vector. The final converged values for the layers, denoted here by the addition of an asterisk,  $[i_L^*, i_{L-1}^*, \dots, i_1^*]$ , constitute the best solution for the azimuth, elevation, scale, in-plane rotation, and (x,y) translations found by the MSC.

For template-based classification, the transformations are all readily invertible with, of course, the exception of the 3-D-to-2-D projection. (The structure of our MSC is designed such that this inverse transformation is not needed, as discussed later.) In essence, each layer of the MSC is designed to search for a given parameter—i.e., to take the original  $L$ -dimensional correspondence problem,

$$c(T) = \langle T^{(L)} \circ \dots \circ T^{(1)}(\mathbf{M}), \mathbf{I} \rangle, \quad (4)$$

and approximate it by decomposing it into  $L$ -simplified one-dimensional searches—one for each transformation layer:

$$\begin{aligned} c_1(T^{(1)}) &= \langle T_{i_1}^{(1)}(\mathbf{M}), \overline{T^{(2)} \circ \dots \circ T^{(L)}(\mathbf{I})} \rangle \\ c_2(T^{(2)}) &= \langle T_{i_2}^{(2)}(\mathbf{M}), \overline{T^{(1)}(\mathbf{M}), T^{(3)} \circ \dots \circ T^{(L)}(\mathbf{I})} \rangle, (5) \\ &\dots \\ c_L(T^{(L)}) &= \langle T_{i_L}^{(L)}(\mathbf{M}), \overline{T^{(L-1)} \circ \dots \circ T^{(1)}(\mathbf{M}), \mathbf{I} \rangle \end{aligned}$$

where only the parameters associated with the transformation in a given layer  $K$  are competed. ( $T^{(K)}$  is the adjoint of the transformation  $T^{(K)}$ .) For each layer,

we generate a vector of correspondences that allows us to compete the transformation parameters for that layer. The state space search in the  $K$ th layer is essentially locally optimized about the current estimates of the other  $L-1$  transformations denoted by, for example,  $\overline{T^{(L-1)} \circ \dots \circ T^{(1)}(\mathbf{M})}$ , where the overbar represents an averaging over these transformation states. These averages are equivalent to the forward/backward MSC superpositions, where the weight vectors,  $\mathbf{g}$ , can be interpreted as the probabilities associated with the transformation parameters. The forward and backward superpositions needed for evaluating layer  $K$  are given by the following recursive equations:

$$\begin{aligned} \overline{T^{(K-1)} \circ \dots \circ T^{(1)}(\mathbf{M})} &= \mathbf{S}_F^{(K-1)} \\ &= \sum_{i_{K-1}=1}^{N^{(K-1)}} g_{i_{K-1}}^{(K-1)} T_{i_{K-1}}^{(K-1)} \circ (\mathbf{S}_F^{(K-2)}) \\ \overline{T^{(K+1)} \circ \dots \circ T^{(L)}(\mathbf{I})} &= \mathbf{S}_B^{(L-K)} \\ &= \sum_{i_{K+1}=1}^{N^{(K+1)}} g_{i_{K+1}}^{(K+1)} T_{i_{K+1}}^{(K+1)} \circ (\mathbf{S}_B^{(L-K-1)}) \end{aligned} \quad (6)$$

where  $\mathbf{S}_F^{(0)} \equiv \mathbf{M}$  and  $\mathbf{S}_B^{(0)} \equiv \mathbf{I}$ .

The Ordering Property of Superpositions ensures that a higher correspondence value will be computed on average for the best choice of transformation parameter—i.e., the vector of correspondences computed for layer  $K$ ,

$$c_K(T^{(K)}) = \langle T_{i_K}^{(K)} \circ \mathbf{S}_F^{(K-1)}, \mathbf{S}_B^{(L-K)} \rangle, \quad (7)$$

will, on average, have a larger value for the  $i_K^*$  element in the vector. These correspondence vectors are used to update the  $\mathbf{g}$  vectors for each layer for the next iteration. Updating is accomplished via a nonlinear competition function to force less likely transformation parameter  $\mathbf{g}$  vector elements to zero. If the target under consideration is not present in the image, all  $\mathbf{g}$  vector values can be allowed to converge to zero. For applications requiring a high probability of detection, the weights can optionally be renormalized after each iteration, thus forcing the MSC to detect something in the image. False alarms that are generated can be addressed by additional GOF metrics as described later.

On each iteration, the MSC efficiently searches a hyperplane through the original high-dimensional state space, revising its estimates of the transformations as it modifies the  $\mathbf{g}$  vectors for each layer on the basis of the correspondences computed. Hence, for each iteration, the dimensionality of search is the sum of the dimensionality of the parameter space for each layer,  $N_{\text{MSC}} = N^{(1)} + \dots + N^{(L)}$ , rather than the product  $N_{\text{Total}} = N^{(1)} \cdot N^{(2)} \cdot \dots \cdot N^{(L)}$ . See Box 1 for a detailed illustration of the MSC algorithm as applied to a specific example image.

### BOX 1. HOW MSC WORKS

MSC uses a 3-D model to detect, locate, and identify the object in a 2-D image and determine the transformations required to project the model into the image. It does so by decomposing the multidimensional transformation state space search, in this case 3-D-to-2-D azimuth and elevation, scale, image-plane rotation, and spatial (x,y) shifts, into a series of one-dimensional transformation parameter searches. Linearly weighted sums of individual transformations are used to form what are effectively “composite filters.” The unique aspect of MSC is the use of these composite filters combined with both forward (model to image) and backward (image to model) projection pathways. The MSC is an extremely efficient variation of a conventional template-matching approach and reduces the computations required from  $O(N_1 N_2 N_3 \dots N_L)$  to an iterative search involving computations on the order of  $O(N_1 + N_2 + N_3 + \dots + N_L)$ , where  $N_i$  is the number of parameters considered in layer  $i$  of an  $L$  layer circuit architecture.

The detailed example shown here illustrates the MSC finding a pickup truck in the image shown in Fig. 3. The single input 3-D model and corresponding azimuth definitions are shown in Fig. 4. For this example, a full 360° in azimuth (5° intervals) and an elevation range of 0–20° (also 5° intervals) were used for the 3-D-to-2-D layer. Ten scaling factors were used in the range from 0.85 to 1.15; image-plane rotations ranged from –6° to 6° (in 2° intervals), and spatial (x,y) shifts included every 2 pixels throughout the image.



Figure 3. Input image with a truck (left) and a car in a parking lot.



Figure 4. Input 3-D truck model.

For the first iteration, the MSC is initialized with weights that are uniformly set to unity. (If prior information is available, these weights can be initialized to represent any known *a priori* information.) Figure 5 shows the superpositions generated as well as the weights ( $g$  vectors) used for the first iteration. For example,  $S_F^{(1)}$  is the forward superposition created by summing each azimuth and elevation projection of the model multiplied by the (unity) weighting factors,  $S_F^{(2)}$  is the superposition created by summing each of the 10 different scaling factors applied to  $S_F^{(1)}$  multiplied by the appropriate scaling weight, and  $S_F^{(3)}$  is the superposition generated by the weighted sum of the image-plane rotations applied to  $S_F^{(2)}$ . Although the final forward superposition,  $S_F^{(4)}$ , generated by a weighted sum over all (x,y) spatial shifts is shown, it is not actually needed for the MSC. We provide and use it to illustrate the current hypotheses that the algorithm is considering for the complete projection of the 3-D model into the image (our red overlays). Similar inverse operations taking place on the input image are computed on the backward path. Note that although the weights for the 3-D-to-2-D and (x,y) spatial shift layers are broken out into two separate one-dimensional weight vectors for visualization purposes—i.e., azimuth and elevation for 3-D-to-2-D and spatial x versus spatial y for the x,y layer—they are actually competed jointly (ganged) in each of these two layers.

*Continued*

BOX 1. HOW MSC WORKS—CONTINUED

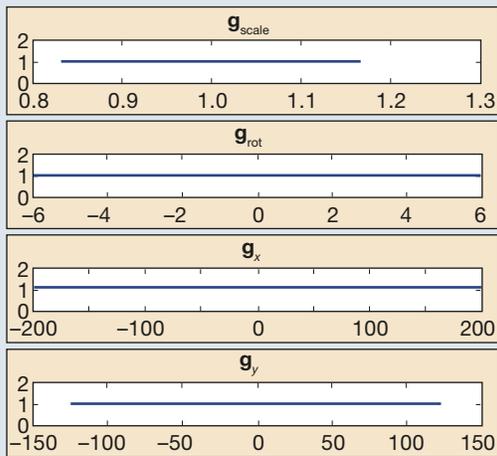
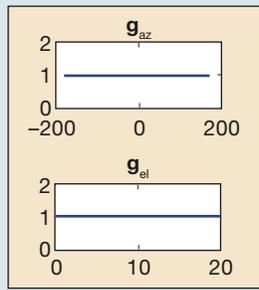
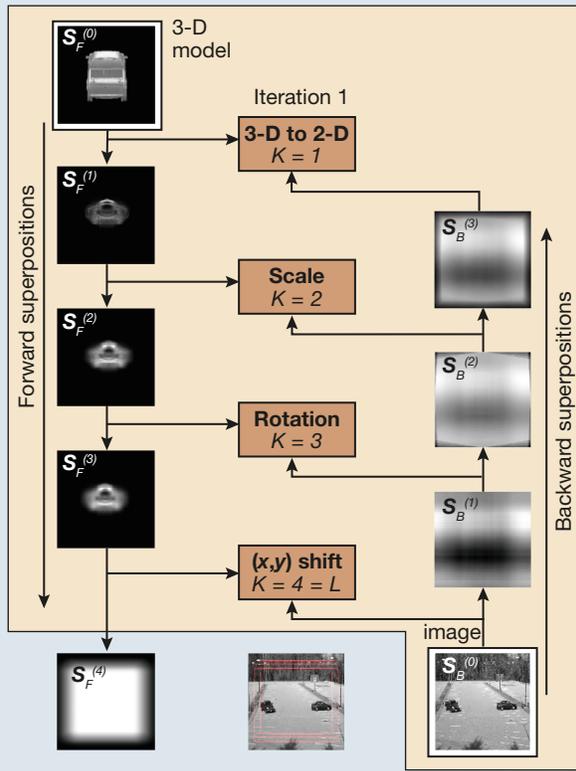


Figure 5. MSC results after the first iteration.

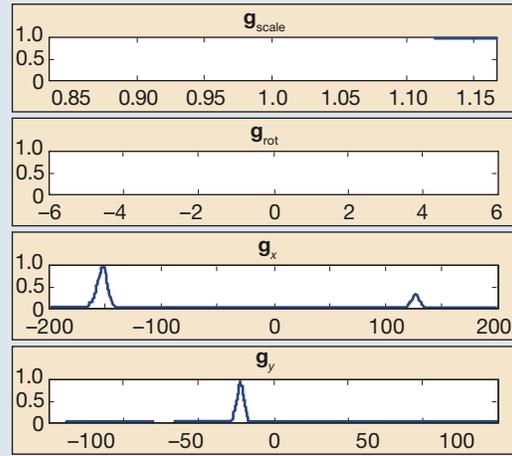
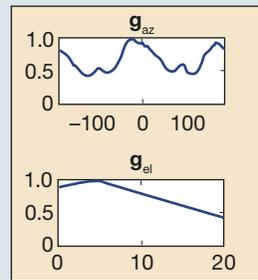
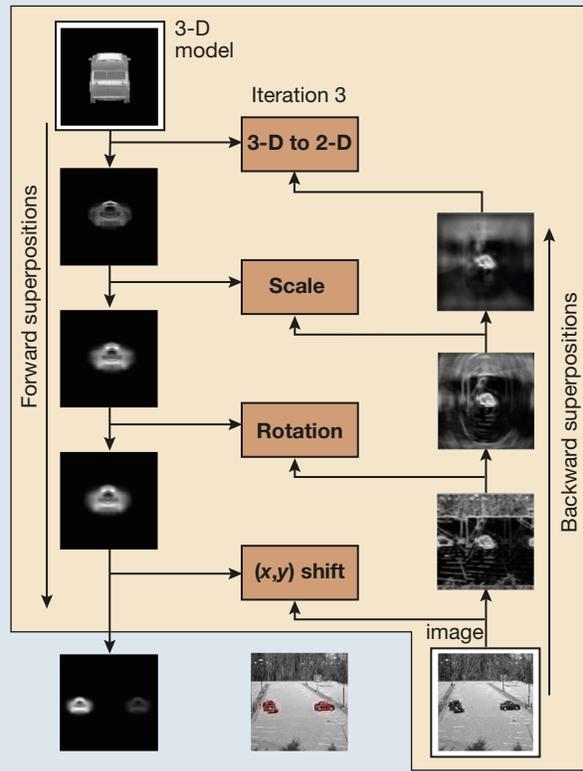
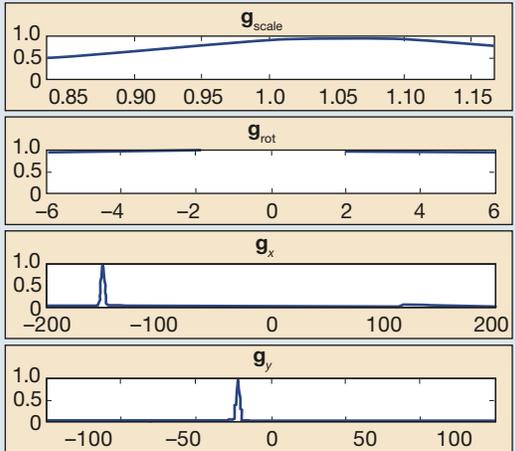
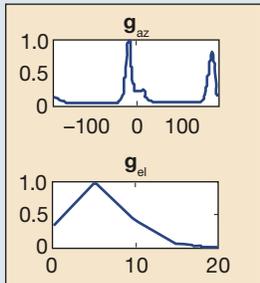
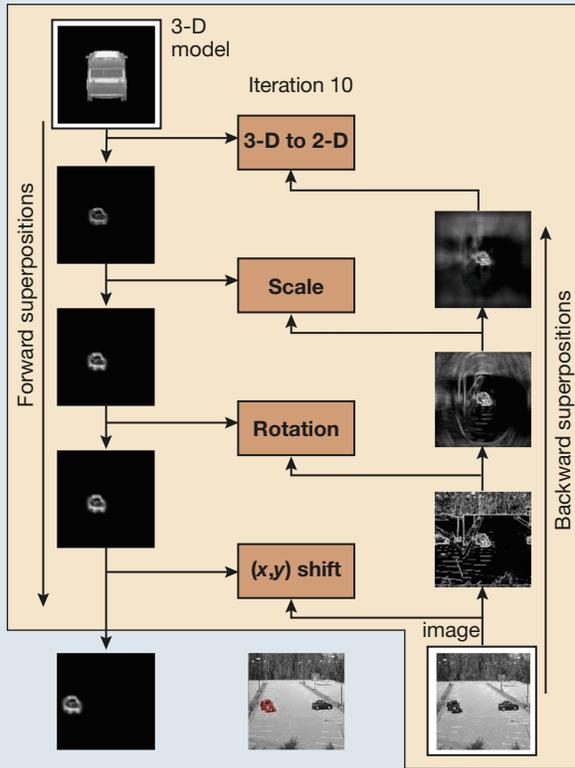


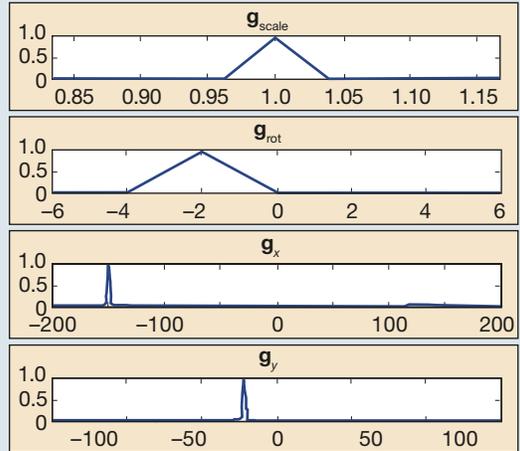
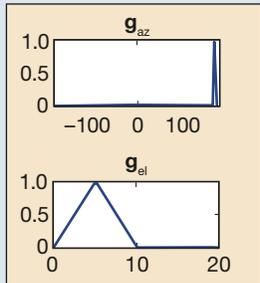
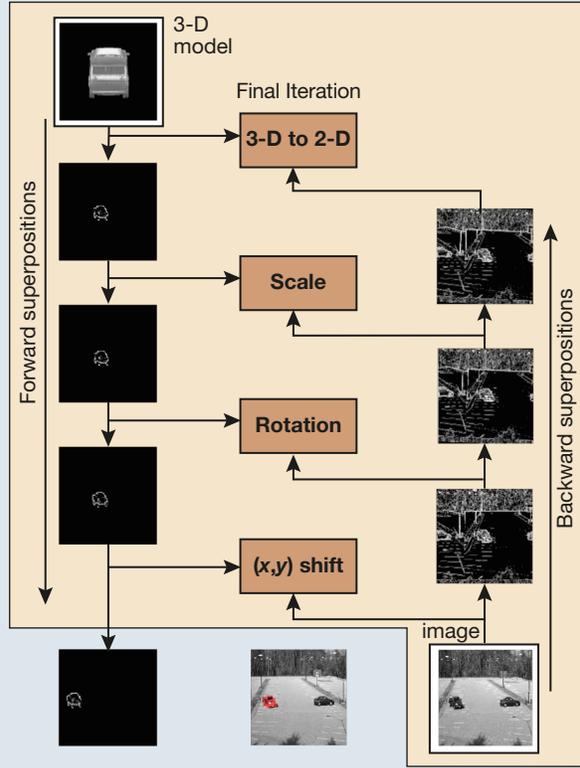
Figure 6. MSC results after the third iteration.

Continued

**BOX 1. HOW MSC WORKS—CONTINUED**



**Figure 7.** MSC results after the 10th iteration.



**Figure 8.** MSC results after the final iteration.

*Continued*

**BOX 1. HOW MSC WORKS—CONTINUED**

Correspondence vectors are computed for each layer and comprise the dot products of the backward superposition with each of the individual transformations for that layer applied to the forward superposition. These correspondence vectors (not shown) are used to update the weights for the next iteration. The Ordering Property of Superpositions guarantees that the correct transformation will, on average, generate a larger dot product than an incorrect transformation. The results obtained by the third iteration are shown in Fig. 6. Our weight update parameters have been tuned so that the order of convergence is tiered: spatial  $x,y$  converges the most quickly, followed by 3-D-to-2-D azimuth and elevation, followed by scaling and rotation. Note that even after only three iterations, the MSC has developed a strong preference for the truck on the left although it is still considering the car on the right. No other objects present in the image are viable candidates. The azimuth and elevation hypotheses are also evolving. Note that the MSC has not yet determined whether the truck is facing toward or away from the camera (dual peaks  $180^\circ$  apart in azimuth); scale and rotation weights are effectively unchanged as yet from the starting uniform priors. The backward path illustrates the wrapped inverse spatial shifts required to make the truck in the image align with the spatially centered truck model.

MSC results obtained after 10 iterations are shown in Fig. 7. At this point, the algorithm has firmly converged on the spatial location of the truck but has yet to resolve the  $180^\circ$  ambiguity in azimuth; in fact, the higher weight is being given to the incorrect azimuth (approximately  $-15^\circ$ ) facing away rather than the correct (approximately  $165^\circ$ ) facing toward the camera. Scale and rotation are now being more aggressively competed. Final converged results are shown in Fig. 8, demonstrating that the MSC was able to recover the correct azimuth ( $165^\circ$ ) as well as determine the best scale and in-plane rotation. Upon convergence, only a single nonzero weight for each layer remains; these weights identify the transformation parameters needed to effectively “shrink wrap” the model onto the image as shown in Fig. 9.



**Figure 9.** MSC converged result projected onto the original image.

**MS-MSC ALGORITHM**

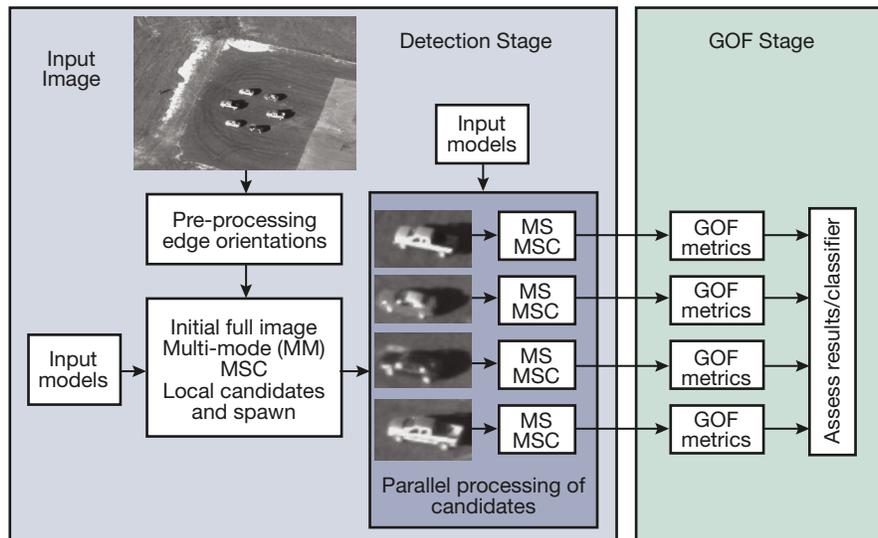
In this section, we discuss the unique properties of our modified MSC algorithm: the MS-MSC algorithm. We have implemented the core MSC algorithm in MATLAB, where the more computationally costly inner-product routines have been coded in C++. A basic block diagram of our  $L = 4$  layer MSC architecture is shown in Fig. 2, where the four layers implemented are azimuth/elevation, scale, in-plane rotation, and  $(x,y)$  translational shifts. Although the MSC can be designed to compete the translational shifts in  $x$  and  $y$  in two separate layers, we have found that better detection performance is achieved by combining  $(x,y)$  shifts into a single layer. This improvement in detection comes at additional computational cost because there are now  $N_{x\text{shift}} \cdot N_{y\text{shift}}$  inner products per iteration rather than the  $N_{x\text{shift}} + N_{y\text{shift}}$  operations required for two separate  $x$  and  $y$  layers.

**Detection Stage**

Testing on data sets containing multiple targets illustrates that the MSC can identify the approximate spatial locations of the targets in the scene very quickly—often within a single iteration. Based on this observation, we modified the original MSC algorithm to essentially have a detection-processing stage where subimages centered on these candidate locations are extracted after just a few initial iterations. Multiple instantiations of the MSC algorithm are then run in parallel on each extracted candidate subimage as shown in the detection stage in Fig. 10. This detection stage processing mitigates the “winner-take-all” limitation of the original MSC, which would converge on only a single candidate in a given image. It also yields increased computational efficiency because the bulk of the MSC processing is performed on much smaller subimage chips as a consequence of the reduction in  $(x,y)$  translational shift parameter ranges.

**Multiple Model Layer**

In typical EO and SAR applications, it is important to be able to detect and classify more than one target type in the same image. Our initial approach to multiple-target classification was to run the detection-phase processing stage described in the preceding subsection with a generic target model (e.g., using a simple car model to detect both the trucks and cars in the input image shown in Fig. 10). This was followed by running parallel MSCs for each target model on all candidates to find the best match. Our current approach implements a new multiple model layer in the MSC algorithm that creates a weighted average of the 3-D-to-2-D projections of multiple target models, which are directly competed in the MSC architecture to find the best model match for each candidate. In our implementation, we force early convergence on the model and translational layers



**Figure 10.** Detection and GOF stages of MM-MSC. The detection stage identifies multiple candidate targets in a given image and processes them in parallel. The GOF stage uses calculated metrics to assess the MSC recognition results. In addition, a classifier could be trained on labeled MSC results to automatically discriminate between good and bad MSC results.

by controlling the competition parameters. This is done to make sure that the estimated pose corresponds to a specific target in the image and is not influenced by other objects present in the scene.

### GOF Metrics Stage

The original MSC algorithm relies on the weighting factors in the  $\mathbf{g}$  vectors to determine whether the template is successfully identified in the input image. A bad match is found if the weights of each layer do not converge to a single nonzero value. Our experience tells us that this approach is not robust enough to discriminate between good and bad solutions in challenging imagery. In our implementation, GOF metrics are calculated to assess the quality of the match between each candidate detection and its final MSC result as illustrated in Fig. 10.

The GOF metrics stage works by first calculating multiple metrics that describe how well the winning projected 3-D model as determined by MSC matches the original subimage data. The edge and the original grayscale values of the projected template and input subimage are used to calculate the various metrics, which include the correlation coefficient, entropy, and Hausdorff distance.<sup>7</sup> A GOF feature vector is created that comprises these various metrics. A classification approach, such as support vector machine (SVM), could then be trained using the GOF feature vectors of both true and false-positive target recognition examples to provide an automated classification capability and confidence metric. The results presented in this article do not yet include a formal classification stage based on the

GOF metrics. However, we have since successfully applied SVM classification-based MS-MSC to automatically detect and identify logos in Flickr images for a Research Program in Applied Neuroscience (RPAN) project.<sup>8</sup>

## MS-MSC FOR PANCHROMATIC GRAYSCALE IMAGES

EO and SAR imagery require different rendering software to create the model templates as well as modality-specific preprocessing of both the input images and the templates prior to using the MS-MSC algorithm. In the following sections, we discuss the creation of the templates, describe the EO image edge preprocessing performed, and present

results that highlight the performance of MSC on panchromatic imagery.

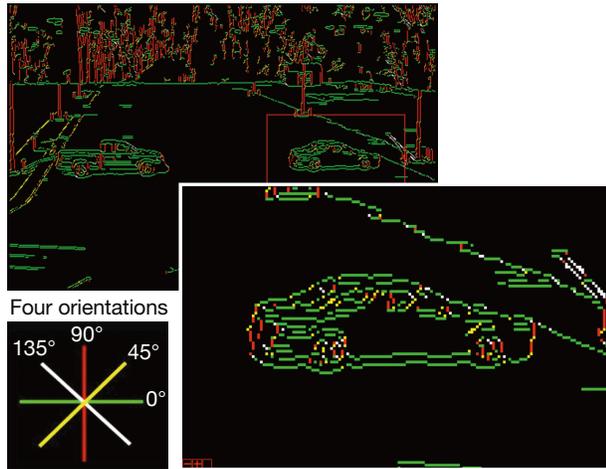
### EO Model Template Generation

The MSC requires 3-D models of the targets at the beginning of the forward loop. The first layer in the forward loop uses various 2-D projections of the models to create the initial superposition images. Because only the 2-D projections are needed, folders are created that contain panchromatic 2-D image projections of the models at various discrete combinations of azimuth and elevation angles. These folders then become the source of the input templates for the MSC algorithm. The template folders are generated once offline and used in all subsequent applications of the MSC.

In the case of EO imagery, the 2-D projections are created by applying orthographic projections of the 3-D models at different azimuth and elevation angles using built-in MATLAB routines. 3-D target models are relatively easy to obtain online because of the rapid growth in the virtual reality and video gaming communities. An increasing number of websites exist where models, including military targets, can be affordably purchased and downloaded (for example, [www.turbosquid.com](http://www.turbosquid.com)). Although these models are not formally vetted, we have found them to be quite realistic and of more than adequate fidelity for our applications.

### EO Template and Image Preprocessing

Because the MSC requires a sparse data representation, edge-detection preprocessing is done to both



**Figure 11.** Example of preprocessing to extract edge location and edge orientation for a panchromatic image. (Reproduced from Ref. 6, © IEEE 2009.)

model templates and input images. Using edges to “sparsify” the images was originally suggested by Arathorn in his implementation of the MSC algorithm for panchromatic imagery. However, as the complexity of the imagery increased, we found that simple edge extraction did not result in a representation that was sufficiently sparse. To address this, we introduced the use of both edge location and edge orientation information as shown in Fig. 11.<sup>9</sup> Here, edge orientations have been quantized into one of four orientation angles:  $0^\circ$  = green,  $45^\circ$  = yellow,  $90^\circ$  = red, and  $135^\circ$  = white. To constitute a successful match, not only do the edge pixel locations in both template and image have to align, but the *orientation* of the edge pixels must also be the same. High inner-product values are obtained in the MSC algorithm only when both the edge pixel location and orientation match.

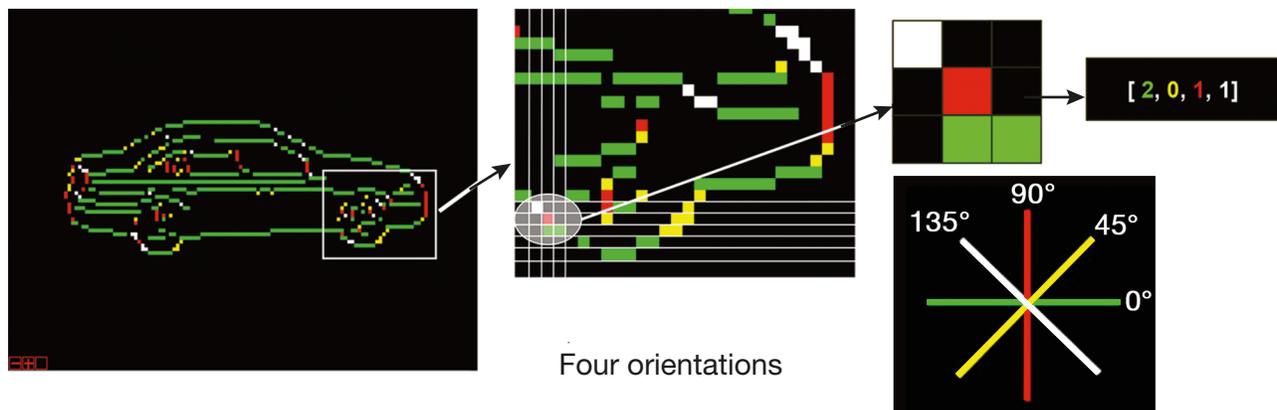
The use of oriented edges can be taken a step further by introducing edge feature vectors as shown in Fig. 12. An edge feature vector is constructed from the edge

orientation histogram computed in a local neighborhood. Because the feature vector captures information in the local neighborhood, it does not necessarily need to be computed for each pixel; spatial downsampling can be used to compensate for the growth in dimensionality caused by the feature vector representation. The benefit of using edge feature vectors is that they are robust against small-scale mismatches between the image and the model. Also, the addition of edge orientation information results in a more discriminatory correspondence value. The edge orientation feature vector has proven to be a powerful technique for images with adequate spatial resolution. However, as the number of edge pixels on the target/model decreases or as edge complexity increases, downsampling increases the risk of missing the target and the mismatch forgiveness inherent in this approach increases the likelihood of false detections.

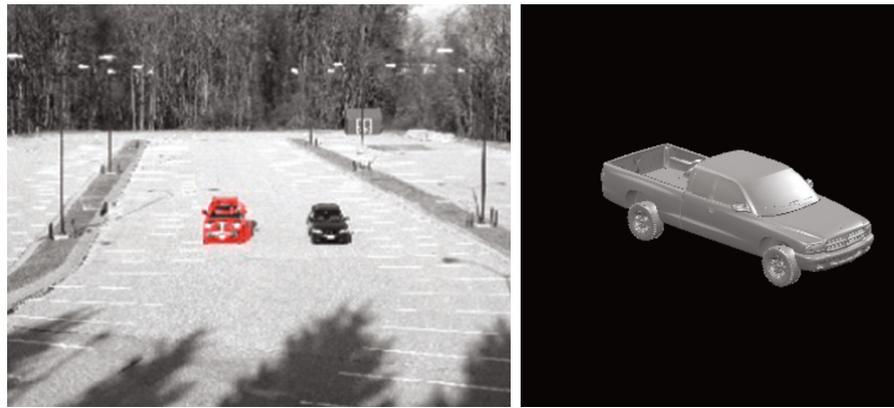
### EO Results

In this section, we present five examples of using the MSC algorithm on challenging panchromatic images. Each of the examples highlights different critical aspects of the MSC algorithm. The MS-MSC algorithm was run using edge feature vectors. In all of the examples, we present the final converged results by projecting the outline of the template, with the pose estimated by the MSC, in red on the input grayscale EO image. Ground truth about the true 3-D pose was not available for any of the examples presented; however, the MSC results were within a few degrees in azimuth and elevation and a few pixels in  $(x,y)$  of those determined manually. Additionally, for each example, information is provided about the convergence efficiency of the MSC as compared with performing an exhaustive search over the entire state space; this information is presented as a percentage of the state space searched.

The parameter search space, (min\_value:step\_size:max\_value), for each example was similar: azimuth



**Figure 12.** Example of calculating an edge orientation feature vector for a car model template. (Adapted from Ref. 6.)



**Figure 13.** MSC results using an input 3-D truck model and an image with challenging azimuth pose (truck on the left, car on the right). The 3-D model template of the target truck model input is also displayed. (Adapted from Ref. 6.)

### Example 1: Varying Pose

The MSC can detect and estimate the pose of the truck under challenging azimuth orientations as shown in Fig. 13. In this stressing case, both the truck (on the left) and the car were positioned to face the camera. Searching for the input truck model, the MSC takes 13 iterations (or 0.24% of the entire search state space) to converge to the correct vehicle and the correct pose. Figure 1 is another example of similar performance obtained for a different target orientation. The result matches our manually estimated pose for the truck in the image.



**Figure 14.** MSC results using an input 3-D car model and an image with complex clutter and lighting. (a) Original image. (b) MSC result projected in image. (Reproduced from Ref. 6, © IEEE 2009.)

### Example 2: Complex Clutter and Lighting

In Fig. 14 we present an example where the car was detected and its pose correctly determined in an image with complex lighting and clutter. This time the input is the 3-D car model. Even though the complicated edges obtained for this image decrease the sparsity of the superpositions at the various layers and intro-

duce local maxima where the MSC could incorrectly converge, the MSC was able to estimate the correct pose of the target in 12 iterations (0.64% of the entire search state space).

angle,  $0^\circ:5^\circ:359^\circ$ ; elevation angle,  $0^\circ:5^\circ:40^\circ$ ; in-plane rotation angle,  $-10^\circ:2^\circ:10^\circ$ ; and  $x$  and  $y$  translations, horizontal and vertical shifts at 2-pixel intervals over the entire image. Ten different scaling factors were competed. Scale range variations were kept modest; the assumption was that a fairly accurate estimate of scale is available *a priori* (especially true for SAR). For applications where this is not true, MSC may still be applied; however, additional care must be exercised so that the resulting superpositions do not become too dense and violate the sparseness precondition.

Our first two examples show how the original MSC can accurately estimate the correct pose of a target in a panchromatic image with significant computational time savings compared with an exhaustive search. For these two examples, the MSC was run in a winner-take-all mode. Examples 3 and 4 show the additional benefits of the detection and GOF states of our MS-MSC algorithm. Finally, in the fifth example, we show the benefits of introducing a multiple model layer.

duce local maxima where the MSC could incorrectly converge, the MSC was able to estimate the correct pose of the target in 12 iterations (0.64% of the entire search state space).

### Example 3: Multiple Targets

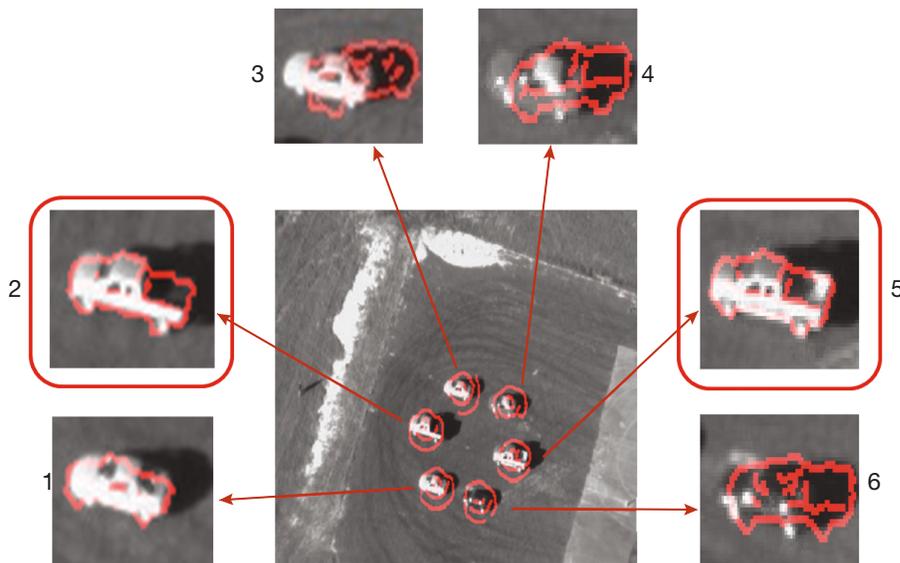
This example uses an image obtained from the Air Force Research Laboratory (AFRL) Sensor Data Management System public website video data sets (available at <https://www.sdms.afrl.af.mil>). The input image, shown in Fig. 15, contains six vehicles including two pickup trucks and four cars. We ran our MS-MSC on this image using our 3-D model of a pickup truck obtained from TurboSquid as the input model. This example demonstrates the ability of the MS-MSC to detect, recognize, and estimate the 3-D pose of more than one target in a single image.

The center image in Fig. 16 shows the result of the detection stage processing. The detection phase spatially locates all six vehicles in the image by the



**Figure 15.** AFRL image taken from an airborne platform showing six vehicles—two trucks and four cars—of various unknown makes and models.

second iteration of the MSC applied to the entire image. Each of these candidate locations identifies an initial detection that may match the pickup truck target model. The six surrounding subimages shown in Fig. 16 are the image chips that were extracted and passed to the parallel MSC processing stage. The final resultant match for each image chip is overlaid in red. The best two matches of the projected 3-D pickup truck model correspond to the two actual pickup trucks in



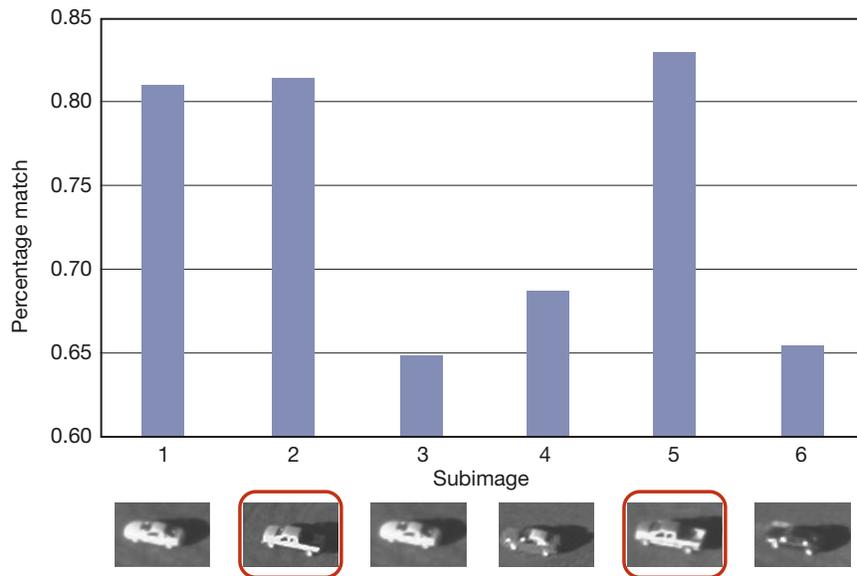
**Figure 16.** Results of applying the MS-MSC algorithm to the AFRL image in Fig. 15. The image in the center shows the initial candidate locations identified after the second iteration of the MSC; the circular red superposition contours illustrate the capability of the MSC to quickly identify the spatial locations of the objects even before the convergence of any of the other pose parameters (detection phase). Each of the six candidates identified were then processed in parallel, and the MSC was allowed to fully converge to the final pose estimates shown in the zoomed subimages.

the image (subimages 2 and 5). Note that for high-elevation viewing geometries, azimuth and in-plane rotation can be easily conflated. The pose estimated by the MSC provides a reasonable fit in this case as shown in Figs. 16 and 17.

The MSC attempts to modify the various pose parameters to force a fit of the model to the other vehicles present as well, as evidenced in Fig. 16. For the cars present, the vehicle shadows in the image are frequently being mismatched to the model. Explicit use of shadow information will be a future enhancement to our EO MS-MSC implementation; shadows are, however, being used currently in the SAR MSC implementation as discussed in a later section. Various GOF metrics are calculated for each of the six solutions and used to quantify the quality of the matches. Our metrics confirm that only the two trucks in the image are good matches to the input truck model. One of the metrics calculates the percentage of overlap between edges of the projected template and the edges of the candidate subimage. Figure 17 shows the value of this metric for each of the six candidates in the image. A percentage match higher than 80% was obtained for the two pickup trucks plus the car (subimage 1) in the bottom-left corner of the original full-scene input image. Using this metric alone to discriminate between good and bad matches would provide only one false alarm (subimage 1). By combining this metric with others into a GOF feature vector, it is possible to train a classifier to significantly reduce false alarms and poor matches.

#### **Example 4: Multiple Targets, High-Clutter Image**

The image in this example also comes from the AFRL Sensor Data Management System site (Fig. 18) but is more challenging than the previous example because of the highly cluttered background and challenging lighting conditions. This image has three pickup trucks in it and was run against the truck model. In the detection phase (center image), we obtain 11 candidate detections, six of which are vehicles. The best match between the projected template and the candidate image chips was indeed the white pickup truck. This image helps to highlight some of the key



**Figure 17.** Percentage match between the projected truck model and image edges for each of the candidate subimages. The two actual trucks in the scene, subimages 2 and 5, have the highest percentage match GOF.

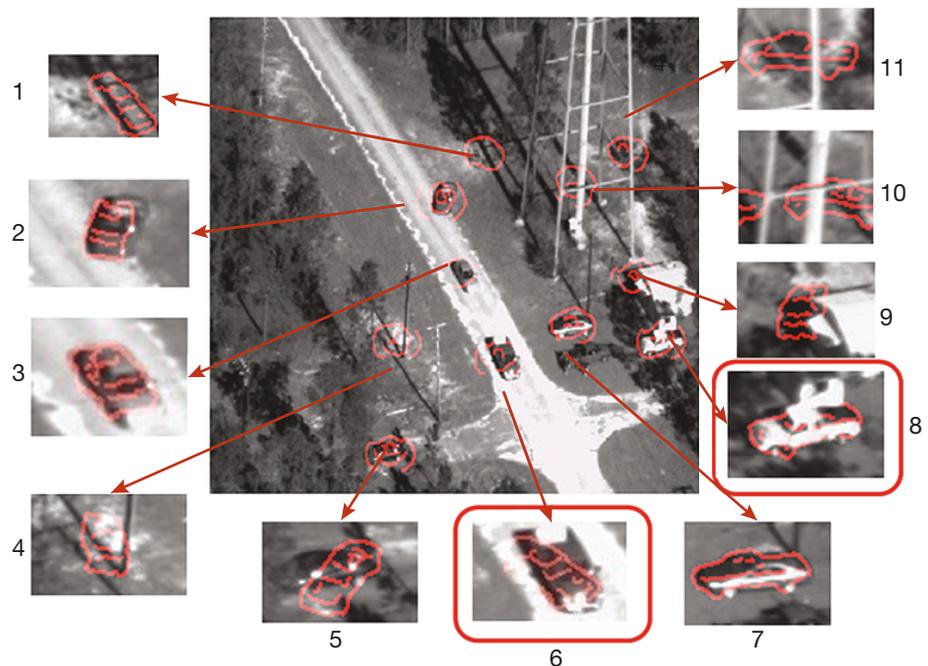
vehicles except the one truck in the right foreground; the lack of contrast between the truck and the background resulted in insufficient edge information being extracted. MSC will work if enough edges (with their corresponding orientations) can be extracted successfully. In this case, poor edge preprocessing for this truck and several of the other vehicles present resulted in reduced MSC results. Specular reflections noticeable in many of the subimages contribute to the poor initial edge extraction.

Additionally, it can be seen that we detected five false targets in the image that do not correspond to vehicles. These five false alarms contain linear features (e.g., water tower support structure) that in the first iterations of the detection stage

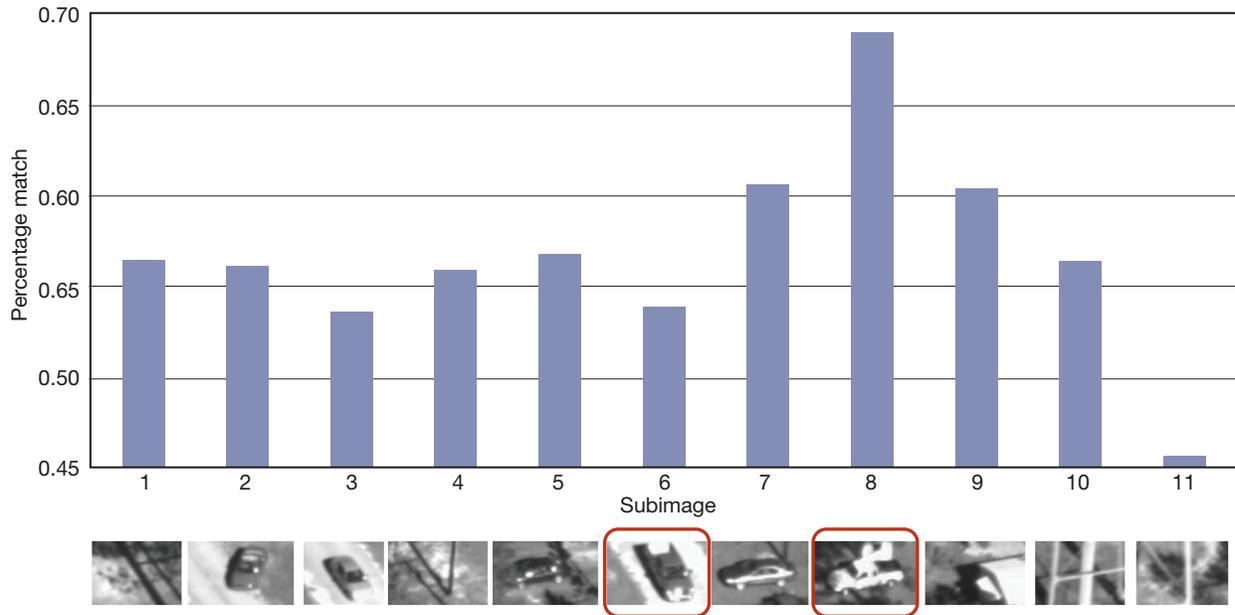
attributes of the MSC algorithm. The computation time was greatly reduced when compared with that required for an exhaustive search; each of the targets converged in less than 0.10% of the time required to exhaustively search the entire state space. The MSC also exhibited a degree of tolerance for target obscuration as evidenced by the detection of the car hidden behind trees in the lower-left corner of the image (subimage 5). Although the final fit is poor, the fact that this target was initially detected even when using the truck model was both surprising and encouraging.

This image also helps to highlight some of the weaknesses of MSC (and other similar template-matching algorithms) in challenging imagery. The clutter and varying lighting conditions show the importance of applying good preprocessing techniques prior to applying MSC. The detection phase found all of the

of the MS-MSC algorithm have a high correspondence with the linear features of the model. The correspondences of these false alarms are not as high as the



**Figure 18.** Results of applying the MS-MSC algorithm to the AFRL image. The superposition contours shown in the central image illustrate the MSC results obtained after just a few initial iterations prior to convergence (detection phase). Each candidate thus identified was then processed in parallel and allowed to converge to the final transformation poses shown in the zoomed subimages. The MSC algorithm was used to estimate the best 2-D projection of the model template to each of these initial candidate detects.



**Figure 19.** Percentage match between the projected truck model and image edges for each of the candidate subimages. The two actual trucks detected in the scene are subimages 6 and 8.

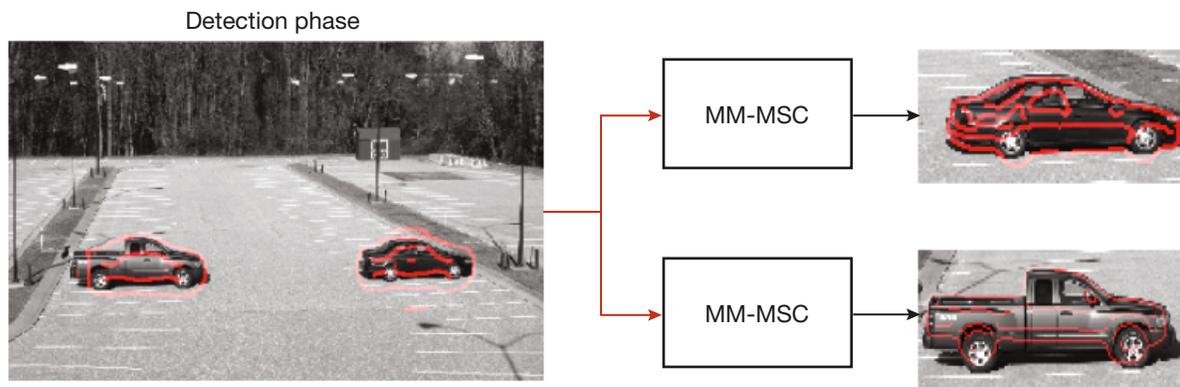
values obtained from the vehicles in the image. If the MS-MSC is run in a winner-take-all mode, these false alarms are eliminated and the algorithm converges to the white pickup truck in the lower right (subimage 8). Because the MS-MSC is retaining and processing all initial candidate detections in parallel, it is necessary to find other means to discard false alarms. Our approach uses the GOF metrics to accomplish this. Initial looks at the metrics calculated for this image indicate that they are capable of discriminating between good and bad matches. Figure 19 shows the percentage match between the edges of the projected truck model and each of the detected candidate image chips. A very high percentage match value is obtained only for the white truck in the image.

#### Example 5: Multiple Model MSC

The MSC implementation in this example incorporates both the detection phase processing and the multimodel layer to compete multiple models simultaneously. Figure 20 shows the results of using both the 3-D truck and car models as inputs to the multiple model layer in our MS-MSC algorithm operating on an image that contains the two types of vehicles. In each case, the MSC determined the appropriate model and pose of each vehicle in less than 14 iterations ( $<0.25\%$  of the entire search state space).

#### MSC FOR SAR IMAGES

Although the features used in the matching process are distinct, the MSC architecture and pose-related



**Figure 20.** MSC results with 3-D truck and car models used as inputs to the multimodel layer in the MS-MSC algorithm. Initial detection phase superposition contours are shown in the left-hand image, and final convergence of both model and pose are shown on the right. (Reproduced from Ref. 6, © IEEE 2009.)

transformation layers are equally applicable to SAR imagery.<sup>10</sup> The aspects unique to SAR are in the creation of SAR templates from the model and the image preprocessing techniques. These differences are discussed in the following sections. Note that we currently apply MSC to only the magnitude of the SAR data; the richer aspects of the full complex (magnitude and phase) SAR imagery are not yet fully exploited. Results highlighting the successful application of MSC to example MSTAR data<sup>11</sup> are shown in the *SAR Results* section. MSTAR is a public release SAR data set collected by Sandia National Laboratories and sponsored by the Defense Advanced Research Projects Agency and AFRL.

### SAR Model Template Generation

We generate a set of synthetic 2-D SAR image templates from a 3-D model using the radar simulation software package Xpatch. Xpatch takes as inputs a 3-D computer-aided design or facet model, a set of radar parameters, and an azimuth and elevation angle and generates a synthetic SAR image. Obtaining realistic SAR images from Xpatch requires that radar parameters be chosen to accurately model the SAR sensor of interest. We adopt Xpatch's azimuth angle conventions such that images are oriented with zero azimuth pointing to the left and increase in a counterclockwise direction. The set of 2-D projections obtained from Xpatch over the searchable range of azimuth and elevation become the input to the forward layer of the MSC algorithm. Like the EO model templates, the SAR model templates are generated once offline and stored for subsequent use by the MSC algorithm.

### SAR Template and Image Preprocessing

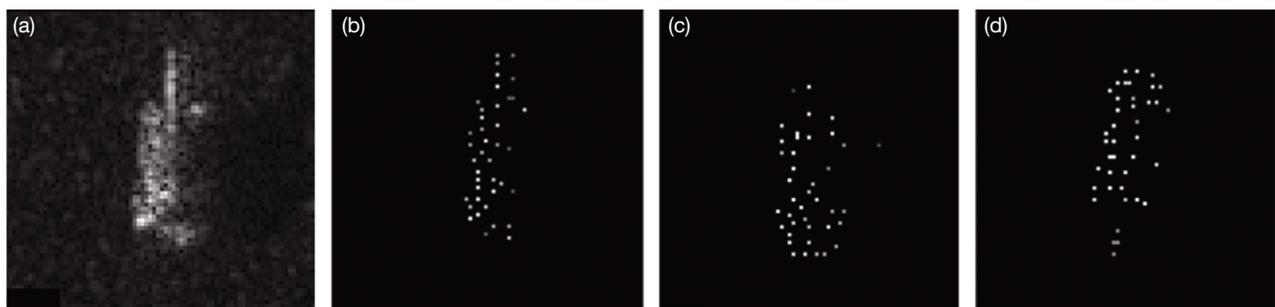
The MSC algorithm operates most efficiently when working on sparse data. In SAR imagery, the determination of the most effective image features required further investigation. In this section, we explore different preprocessing techniques to extract sparse features from the SAR imagery and show their effectiveness when used by

#### ALGORITHM 1. SCATTERING POINT EXTRACTION

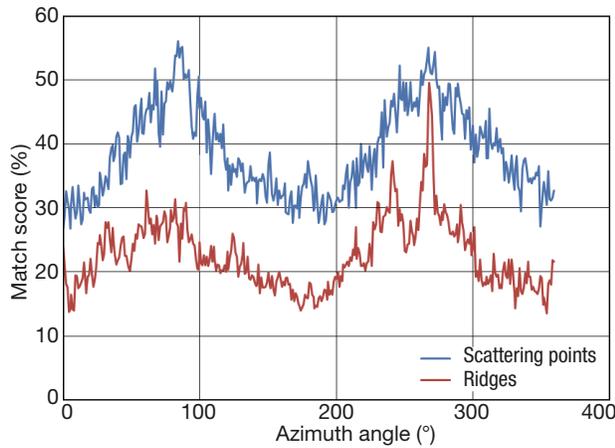
1. Identify pixels that represent local SAR magnitude maxima in a moving neighborhood window of  $L \times L$  pixels.  $L$  is kept small so that closely spaced scattering centers can be found; a value of 3 was used in our analysis.
2. Keep only the top  $k$ th percentage of the identified maximum pixels (usual ranges of  $k$  are 1–10%).
3. Dilate each scattering center pixel to provide some uncertainty in spatial location for subsequent matching.
4. Normalize scattering image so its Frobenius norm equals 1.

themselves or in combination in our SAR MSC implementation. The features extracted for SAR imagery include scattering point magnitude and location, ridge position and orientation, and shadows. All the features are extracted from the magnitude of the complex-valued SAR images, where the magnitude represents the reflectivity of scatterers in the image.<sup>12</sup>

Figure 21a shows an MSTAR SAR image chip of a T-72 tank oriented at an azimuth of 268°. By using the scattering point extraction steps outlined in Algorithm 1, the scattering point centers for the MSTAR chip were derived as shown in Fig. 21b. The scattering points extracted from two Xpatch-generated templates for the T-72 model oriented at azimuth angles of 268° and 84°, respectively, are shown in Figs. 21c and 21d. Although the MSTAR target and the Xpatch-generated template scattering centers shown in Figs. 21b and 21c, respectively, arise from the same azimuth orientation, the scattering centers do not align perfectly. In fact, our initial attempts at applying MSC to the MSTAR chip in Fig. 21a converged to the wrong solution of 84° azimuth (Fig. 21d) when using only scattering center features. This led us to investigate the discriminatory power of scattering centers. Figure 22 plots the percentage match



**Figure 21.** Comparison of extracted scattering centers. (a) Original MSTAR SAR T-72 tank target at azimuth 268°, (b) extracted scattering centers from MSTAR image, (c) scattering centers extracted from Xpatch template generated from T-72 model at azimuth 268°, and (d) scattering centers from Xpatch template at azimuth 84°. (Adapted from Ref. 10.)



**Figure 22.** Results of an exhaustive search to match the MSTAR image shown in Fig. 21a, which has an azimuth of 268°, to Xpatch-generated templates with azimuths varying over a full 360°. Scattering point (blue) and ridge (red) match scores are shown. The result shows that ridges are more discriminatory and have a maximum at the correct solution for the azimuth angle (268°). (Adapted from Ref. 10.)

score (i.e., normalized dot product) of scattering centers (blue) for the MSTAR target in Fig. 21, with Xpatch templates varying over a full 360° in azimuth. These results are obtained by an exhaustive brute force search over both spatial ( $x,y$ ) position and azimuth orientation. Given the scattering center feature representation, the optimal solution as found by this exhaustive search analysis is, in fact, at an azimuth of 84°, an offset of  $\sim 180^\circ$  from the correct orientation. This confirms that the SAR MSC converged to the best solution available (albeit an erroneous orientation match). The secondary peak at 265° in Fig. 22 (blue line) more closely corresponds to the actual azimuth angle of 268°. This result shows that scattering centers alone do not provide sufficient discrimination of targets.

The need for additional feature attributes led us to incorporate ridges. Ridge extraction finds local maxima in at least one spatial direction. Ridges are determined by computing the Hessian matrix at each pixel, which provides information about the strength and orientation (the steps are detailed in Algorithm 2). An example of ridge extraction is shown in Fig. 23. The original MSTAR image is provided in Fig. 23a (same as

#### ALGORITHM 2. RIDGE EXTRACTION

1. Smooth SAR image with a 2-D Gaussian filter.
2. Compute the Hessian matrix for every pixel:

$$H = \begin{bmatrix} H_{xx} & H_{xy} \\ H_{yx} & H_{yy} \end{bmatrix},$$

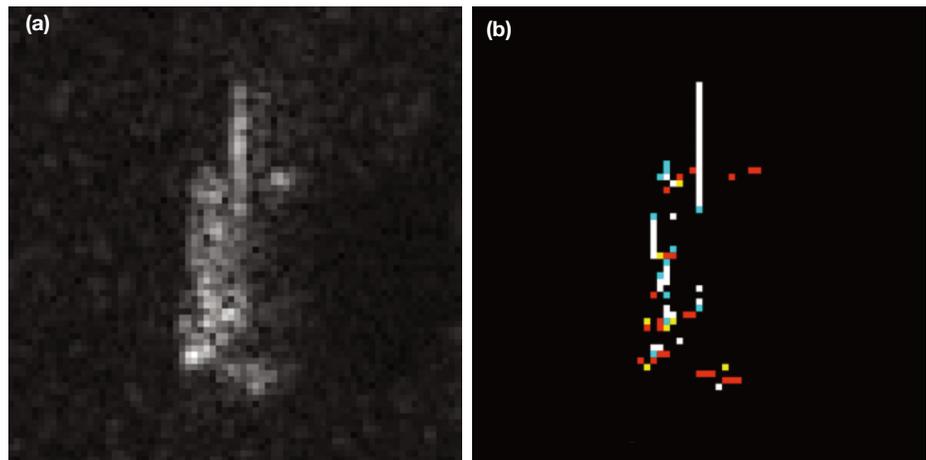
where  $H_{xx}$  is the second partial derivative in the  $x$  direction and  $H_{xy}$  is the mixed partial second derivative in the  $x$  (column) and  $y$  (row) directions.

3. Calculate the ridge strength

$$S(x,y) = H_{xx} + H_{yy} - \sqrt{(H_{xx} - H_{yy})^2 + 4(H_{xy})^2}.$$

4. Calculate orientation of each ridge  $\theta = \tan^{-1}(v_x, v_y)$ , where  $(v_x, v_y)^T$  is the minor eigenvector of the Hessian matrix.
5. Keep the  $k$ th percentile of pixels with the highest ridge strength (usual ranges of  $k$  are 1–10%).
6. Ridges can be quantized into orientation bins.

the left image in Fig. 21), and Fig. 23b shows the location and orientation of the extracted ridges. Red indicates an orientation of 0°; cyan, 45°; white, 90°; and yellow,  $-45^\circ$  as measured counterclockwise from 0° in the horizontal direction. The results of the exhaustive search analysis using only ridge magnitudes are also provided in Fig. 22 (red line). The ridge features are much more discriminatory and identify the correct solution for the azimuth angle at 268°. As the graph shows, ridges are much less susceptible than scattering centers to a 180° azimuth mismatch for this image. The weaker secondary peak around 240° in Fig. 22 is a result of the ridges on the turret misassociating with the side of the tank body.



**Figure 23.** Extraction of ridge information from a SAR Image. (a) SAR image. (b) Ridge position and orientation. Red, 0°; cyan, 45°; white, 90°; yellow,  $-45^\circ$ . [Reproduced from Ref. 10, © Society of Photo-Optical Instrumentation Engineers (SPIE).]

In this implementation of the MSC for SAR imagery, we are not yet taking advantage of the ridge orientation information; only ridge location is used. However, future work will incorporate this information to provide further discrimination capability, as was done with the MSC for panchromatic imagery.

The shadow extraction algorithm attempts to find the darkest areas of the image. We use a simple extraction technique where shadow regions are identified by applying two low-pass filters and a threshold. The dark regions of the image are matched against shadow templates generated in Xpatch. More-advanced techniques, such as those found in Refs. 13 and 14, would enhance our capabilities to correctly identify and segment shadow shapes and improve the match between the Xpatch generated templates. However, even a coarse computation of the shadow region added valuable information for classifying targets.

### SAR Results

The results obtained after applying the MSC algorithm to MSTAR<sup>11</sup> data are presented in this section. Data were collected on several different target types, including a T-72 tank, a BMP2 infantry fighting vehicle, and a BTR-70 armored personnel carrier. In addition, large-area natural clutter scenes were collected. Complex imagery—i.e., both magnitude and phase—is provided, but at present, only magnitude information is used in the SAR MSC algorithm.

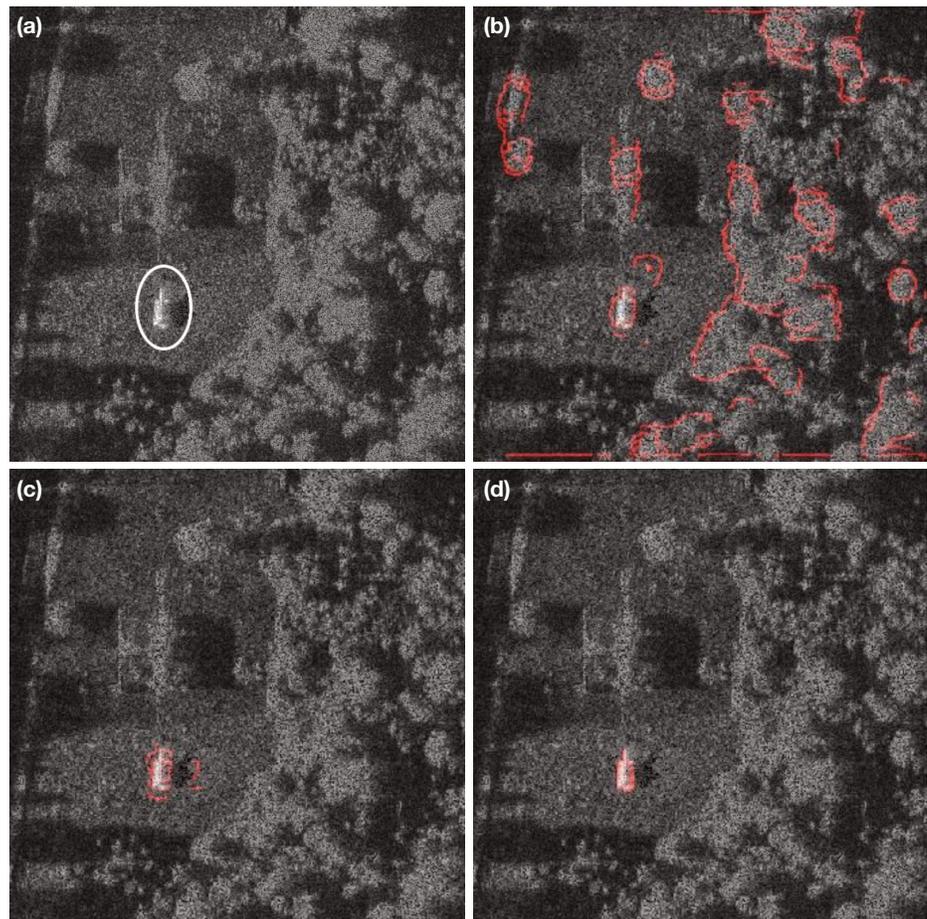
In each example presented here, a T-72 3-D tank model is used to find a real SAR MSTAR T-72 tank embedded in backgrounds with varying clutter. The detection and multiple model stages, introduced earlier, have not yet been implemented for SAR. The SAR implementation of the MSC still operates in a winner-take-all mode; that is, the algorithm searches to find the single best match in the image. A full 360° in azimuth is searched using a 2°-interval step size for all

examples provided. The application of MSC to SAR is less mature than the work presented for panchromatic images. However, the preliminary results presented in this section illustrate the benefits of using MSC with SAR imagery.

#### Example 1: Scattering Point and Ridge Matching in Natural Clutter

In this example, our test scene consists of an MSTAR T-72 tank target chip embedded in a natural clutter background. Both scattering point and ridge features were used in the matching criteria, and the target model input was an Xpatch T-72 tank. Each feature is computed separately to create the correspondence in each layer in the form of multidimensional dot products. The results from each dot product are combined and used to update the weight vector.

The target in the image is oriented at an elevation angle of 15.32° and an azimuth angle of 268°. Figure 24 shows the original input image (with the target high-



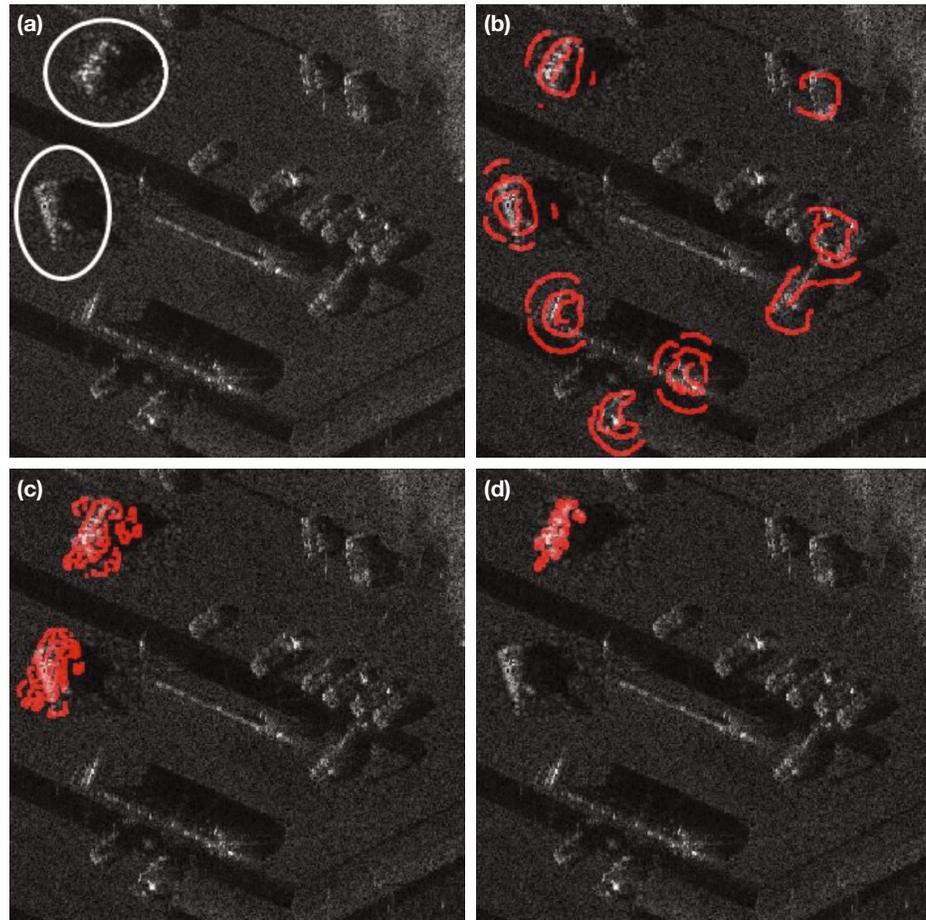
**Figure 24.** Iterations of MSC using scattering point and ridge features to identify the target in natural clutter. The red contours illustrate the forward superposition of the model into the image on each iteration. Convergence is achieved by the eighth iteration, as shown in panel d. (a) Embedded target. (b) Second iteration. (c) Third iteration. (d) Eighth iteration. [Reproduced from Ref. 10, © Society of Photo-Optical Instrumentation Engineers (SPIE).]

lighted) and three of the eight iterations of the MSC algorithm. Note that the correct target location is found after only two iterations. The remainder of the time (five additional iterations) is spent resolving the azimuth angle and fine tuning the position estimate. Convergence occurs at the eighth iteration after the MSC has searched through just 0.56% of the total state space (as defined by an exhaustive search). It correctly finds the target azimuth to be at  $268^\circ$ .

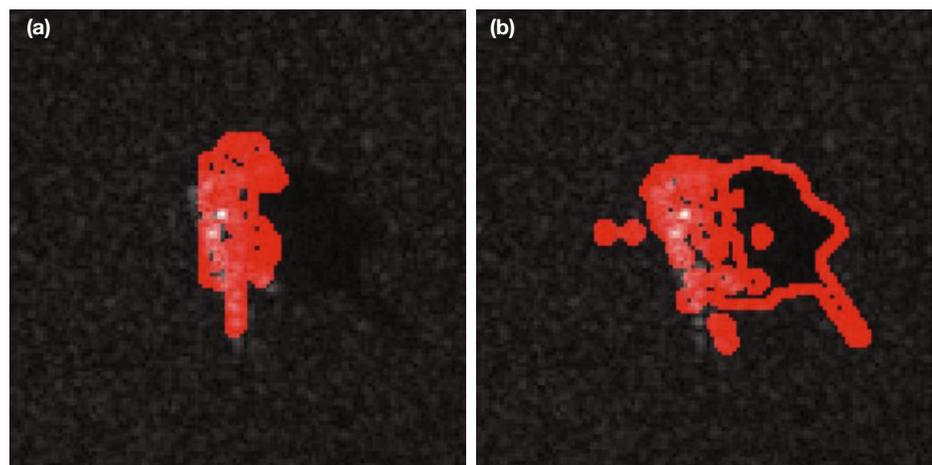
### Example 2: Scattering Point and Ridge Matching in Man-Made Clutter

Our second example embeds two different MSTAR T-72 target chips in a Sandia miniSAR image. The Sandia image was used to provide a more challenging background than was available with the MSTAR data set. It includes multiple vehicles in a parking lot. Both embedded target chips were taken at the same  $15.32^\circ$  elevation but at two different azimuths,  $50^\circ$  and  $109^\circ$ , respectively. MSC was run using scattering point and ridge features to find the target using the input T-72 tank model.

Figure 25 shows the result of the man-made background test. At the third iteration, the algorithm has focused on several vehicles present in the scene, including the two MSTAR tank targets. By the fifth iteration, the algorithm has eliminated all but the two targets. The 12th iteration shows the converged results on the upper-left target. The target was



**Figure 25.** Scattering point and ridge matching in man-made clutter. The red contours illustrate the forward superposition of the model into the image on each iteration. Convergence is achieved by the 12th iteration as shown in panel d. (a) Original image. (b) Third iteration. (c) Fifth iteration. (d) Twelfth iteration. [Reproduced from Ref. 10, © Society of Photo-Optical Instrumentation Engineers (SPIE).]



**Figure 26.** Comparison of converged results obtained for the MSC algorithm with (b) and without (a) shadow features. The addition of shadows provided a final match within  $1^\circ$  of the true azimuth. [Reproduced from Ref. 10, © Society of Photo-Optical Instrumentation Engineers (SPIE).]

matched with a reasonable result of azimuth equal to 56° after searching only 6.7% of the entire state space. In the future, the MS-MS algorithm will incorporate the ability to search for multiple targets in parallel within a single SAR image.

### Example 3: Shadow Matching

The final SAR example highlights the improvement gained by the addition of shadow features to the matching criteria. Figure 26a shows a case where ridges and scattering point features determined that the best target match, after 25 iterations, was at an azimuth value of 90°, a difference of 19° from the true azimuth value of 109°. The addition of shadow matching rotates the result to be closer to the actual tank orientation. After 28 iterations, a final match of 108° azimuth was found, as shown in Fig. 26b—the final match was only 1° off from the true azimuth angle.

## CONCLUSIONS

The MSC algorithm is a novel, biologically inspired technique to detect, identify, and estimate pose of target objects in imagery given a 3-D object model. The MSC emulates aspects of visual cognition by simultaneously projecting the model forward into the image and the image backward into the model. The MSC efficiently searches azimuth, elevation, scale, in-plane rotation, and  $(x,y)$  translational shifts. One of the most impressive aspects of the MSC is how little *a priori* knowledge is required—only a reasonably accurate 3-D target model is needed. Target segmentation, identification, and pose estimation are emergent attributes of the approach.

Our implementation has expanded the concept described in Ref. 2 to include a detection processing stage to quickly identify candidate target locations within the image and extract them for subsequent parallel processing. We also introduce a multiple model layer coupled with the detection phase to enable efficient processing of multiple targets of same or differing model types within a single image. Finally, we introduce a GOF stage that can be used to assess the estimated pose of the candidate targets obtained from the MSC algorithm.

In this article, we also introduce a new image preprocessing technique that uses edge feature vector with location and orientation information to enhance the performance of the MSC on panchromatic EO imagery. Additionally, different preprocessing techniques including scattering centers, ridges, and shadows were introduced that allowed the application of MSC to SAR imagery for the first time.

Several examples are presented that show the benefits of using MSC on both types of imagery. The MSC has demonstrated consistent performance over variations in target azimuth, scene lighting, and background clutter.

All of the examples show how the MSC can locate and correctly estimate the pose of the targets in only a small fraction of the computations needed for a conventional brute force approach—under 1% for the panchromatic and under 10% for the SAR examples presented. (The apparent performance reduction for SAR is a result of the reduced state space as both elevation and scale are largely known *a priori*.)

Although the benefits of the GOF stage were not completely implemented for this article, we subsequently successfully applied it to a Research Program in Applied Neuroscience program where the MS-MS algorithm was used to automatically detect and classify TV logos in random pictures downloaded from the Flickr website. An SVM classifier was trained using the GOF metrics calculated from a labeled training image previously processed by the MS-MS algorithm. The SVM classifier was able to successfully discriminate between good and bad matching results of the MSC with an accuracy of 89.5%.

Our future work includes the incorporation of collection-specific effects in our model templates for applications where support data on collection timing and geometry are known. Additional efforts in sparse representation of both the model and the image are underway to enable larger numbers of simultaneously competed models and better performance on highly cluttered images. We are also continuing our investigation of GOF-based classification to attach confidence scores to the resultant matches as well as identifying sensitivities to partial occlusions.

**ACKNOWLEDGMENTS:** The authors thank APL's Precision Engagement Business Area and Global Engagement Department—now the Force Projection Department—for sponsorship of this research. In particular, we thank Dr. Glenn Mitzel, Fred Riedel, and Mark LoPresto for their support of the MSC research. Imagery data were taken from the Sensor Data Management System video samples (AFRL), the MSTAR data set (Defense Advanced Research Projects Agency and the AFRL), and the Sandia National Laboratories website. We also thank Dr. Daniel DeMenthon for his critical insights and continuing useful discussions on the MSC algorithm and both Sean Martin and Dr. Michael Williams for their helpful review of this article.

## REFERENCES

- Arathorn, D., "Computation in the Higher Visual Cortices: Map-Seeking Circuit Theory and Application to Machine Vision," in *Proc. IEEE 33rd Appl. Imagery Pattern Recognition Workshop*, Washington, DC, pp. 73–78 (2004).
- Arathorn, D., *Map-Seeking Circuits in Visual Cognition, A Computational Mechanism for Biological and Machine Vision*, Stanford University Press, Stanford, CA (2002).
- Lowe, D., "Distinctive Image Features from Scale-Invariant Key-points," *Int. J. Comput. Vis.* 60(2), 91–110 (2004).
- Morel, J. M., and Yu, G., "ASIFT: A New Framework for Fully Affine Invariant Image Comparison," *SIAM J. Imaging. Sci.* 2(2), 438–469 (2009).

- <sup>5</sup>Martin, S., Rodriguez, P., and Murphy, P., "The Application of Simulated Annealing to a Map Seeking Circuit," in *Proc. IEEE Systems, Man, & Cybernetics Conf.*, San Antonio, TX, pp. 3830–3835 (2009).
- <sup>6</sup>Murphy, P. K., Rodriguez, P. A., and Martin, S. R., "Detection and Recognition of 3D Targets in Panchromatic Gray Scale Imagery Using a Biologically-Inspired Algorithm," in *Proc. IEEE AIPR Workshop*, Washington, DC, pp. 1–6 (2009).
- <sup>7</sup>Rote, G., "Computing the Minimum Hausdorff Distance Between Two Point Sets on a Line Under Translation," *Inf. Process. Lett.* **38**(3), 123–127 (1991).
- <sup>8</sup>Vogelstein, R. J., Harshbarger, S. D., McLoughlin, M. P., Beaty, J. D., Yantis, S., et al., "Research Program in Applied Neuroscience (RPAN)," *Johns Hopkins APL Tech. Dig.* **28**(3), 222–223 (2010).
- <sup>9</sup>Olson, C., and Huttenlocher, D., "Automatic Target Recognition by Matching Oriented Edge Pixels," *IEEE Trans. Image Process.* **6**(1), 103–113 (1997).
- <sup>10</sup>Peterson, C. K., Murphy, P. K., and Rodriguez, P. A., "Target Classification in Synthetic Aperture Radar Using Map-Seeking Circuit Technology," in *Algorithms for Synthetic Aperture Radar Imagery XVIII*, Proc. SPIE, Vol. 8051, Edmund G. Zelnio and Frederick D. Garber (eds.), SPIE, Bellingham, WA, pp. 805113–1–805113-10 (2011).
- <sup>11</sup>MSTAR Overview, <https://www.sdms.af.mil/index.php?collection=mstar> (accessed 2 Mar 2011).
- <sup>12</sup>Oliver, C., and Quegan, S., *Understanding Synthetic Aperture Radar Images*, Artech House, Inc., Boston, MA (1998).
- <sup>13</sup>Papson, S., and Narayanan, R., "Modeling of Target Shadows for SAR Image Classification," in *Proc. 35th Applied Imagery and Pattern Recognition Workshop (AIPR'06)*, Washington, DC, pp. 1–7 (2006).
- <sup>14</sup>Jahangir, M., Blacknell, D., Moate, C., and Hill, R., "Extracting Information from Shadows in SAR Imagery," in *Proc. IEEE International Conf. on Machine Vision*, Islamabad, Pakistan, pp. 107–112 (2008).

## The Authors

**Patricia K. Murphy** is a Principal Professional Staff member in the Weapon and Targeting Systems Group of the Force Projection Department. She was the principal investigator (PI) for the MSC EO target recognition work presented in this article. She initiated the investigation into the MSC approach and led the design and development of the MSC prototype software under an independent research and development (IR&D) project. **Pedro A. Rodriguez** is a Senior Professional Staff member in the Weapon and Targeting Systems Group of the Force Projection Department. He was in charge of developing the first version of the MSC algorithm that could handle 3-D models in MATLAB. He was also instrumental in redesigning the MSC architecture and the development of both the multistage MS-MSC and the multiple model MSC. **Cameron K. Peterson** is a Senior Professional Staff member in the Weapon and Targeting Systems Group of the Force Projection Department. She was the PI for an IR&D project that investigated the use of MSC to perform target recognition in SAR. Her work represents the first successful application of MSC to SAR. For further information on the work reported here, contact Patricia Murphy. Her e-mail address is [pat.murphy@jhuapl.edu](mailto:pat.murphy@jhuapl.edu).

The *Johns Hopkins APL Technical Digest* can be accessed electronically at [www.jhuapl.edu/techdigest](http://www.jhuapl.edu/techdigest).