NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

GENERALIZED HOUGH TRANSFORM FOR OBJECT CLASSIFICATION IN THE MARITIME DOMAIN

by

Pornrerk Rerkngamsanga

December 2015

Thesis Advisor:	Murali Tummala
Co-Advisor:	James Scrofani

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY	**2. REPORT DATE** December 2015	**3. REPORT TYPE AND DATES COVERED** Master's thesis	
4. TITLE AND SUBTITLE GENERALIZED HOUGH TRANSFORM FOR OBJECT CLASSIFICATION IN THE MARITIME DOMAIN			**5. FUNDING NUMBERS**
6. AUTHOR(S) Pornrerk Rerkngamsanga			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

A Generalized Hough Transform (GHT)-based classification scheme for an object-of-interest in maritime-domain images is proposed in this thesis. First, the object edge points are extracted and used to generate a representation of the object as a Hough coordinate table by using the GHT algorithm. The table is then reformatted to a contour map called a Hough features map. The coordinates of dominant peaks or Hough features on the map are extracted and fed into a feed-forward, back-propagation neural network for classification. In this research, the scheme is tested using perfect shapes of triangles, squares, circles, and stars and maritime-domain images of ships, aircraft, and clouds, and the classification results obtained are reported.

14. SUBJECT TERMS Generalized Hough Transform, object detection, object classification, Discrete Cosine Transform			**15. NUMBER OF PAGES** 79
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

GENERALIZED HOUGH TRANSFORM FOR OBJECT CLASSIFICATION IN THE MARITIME DOMAIN

Pornrerk Rerkngamsanga
Lieutenant, Royal Thai Navy
B.S., United States Naval Academy, 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 2015

Approved by: Murali Tummala
Thesis Advisor

James Scrofani
Co-Advisor

R. Clark Robertson
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A Generalized Hough Transform (GHT)-based classification scheme for an object-of-interest in maritime-domain images is proposed in this thesis. First, the object edge points are extracted and used to generate a representation of the object as a Hough coordinate table by using the GHT algorithm. The table is then reformatted to a contour map called a Hough features map. The coordinates of dominant peaks or Hough features on the map are extracted and fed into a feed-forward, back-propagation neural network for classification. In this research, the scheme is tested using perfect shapes of triangles, squares, circles, and stars and maritime-domain images of ships, aircraft, and clouds, and the classification results obtained are reported.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I want to express my heartfelt thanks for the consistent support of my advisor, Dr. Murali Tummala, and co-advisor, Dr. James Scrofani. I would also like to thank two classmates, Michail Pothitos and Marcos Chi Lim Siu, at the Naval Postgraduate School, who helped me prepare test images and gave me advice on how to use a neural network in MATLAB.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Computer vision for ocean imagery is a growing research area for detection of objects-of-interest, such as ships, aircraft, islands, clouds, and oil spills [1], [2], [3]. Information obtained from satellite and other aerial imagery can be used to improve maritime safety and security or maritime domain awareness.

Maritime domain awareness is vital for military operations, especially for maritime operations. The superiority in information warfare can improve the chances of victory. To achieve a large area of coverage and speedy communications, satellites, aircraft and drones are commonly used for surveillance and reconnaissance. These platforms gather information in an area of interest and relay it to control stations via electromagnetic transmissions. This information is then analyzed to build a common operating picture [4], which provides timely decision-making information to all levels of command.

To ensure the timeliness and reliability of the common operating picture, an automated object classification system is needed. In this thesis, an effective feature extraction technique and a machine learning based classification tool are used to develop an automated classification scheme for differentiating maritime-domain objects, such as ships, aircraft, and clouds, contained in aerial imagery.

A. THESIS OBJECTIVE

The main objective of this thesis is to develop an effective automated classification scheme based on the Generalized Hough Transform (GHT) and a feed-forward, back-propagation neural network classifier to allow computer systems to classify any object-of-interest contained in aerial images in an automated manner. The GHT algorithm is used for selection of features that represent the shape of an object. Then, a back-propagation neural network is used to classify the object based on the features.

B. RELATED WORK

Two major related works for this thesis are an automatic ship-types classification scheme [5] and a GHT based scheme for fingerprint localization and classification [6].

In [5], a Wiener adaptive filter was used to reduce noise in Synthetic Aperture Radar (SAR) images of ships prior to detection of ship edge points by using the Susan operator. Then, a Discrete Cosine Transform (DCT) was used for feature selection. In this thesis, a DCT filter is used to denoise maritime-domain images containing an object-of-interest before using a Sobel operator to detect the object edge points. .

The GHT algorithm has been shown to be an effective technique for creating R-tables used as models for fingerprinting [6]. Then, these tables are used for feature selection prior to classification. In this thesis, the GHT algorithm is used to create R-tables for all objects-of-interest, but each table is reformatted to a contour map or Hough features map before feature selection.

A back-propagation neural network with the Levenberg-Marquart training function was proven to be effective for ship classification [5]. In this thesis, a neural network with a scaled conjugate-gradient training function was used for object classification.

C. ORGANIZATION

This thesis consists of five chapters. The background of three underlying concepts—the GHT, the DCT, and the neural networks—is covered in Chapter II. A complete proposed GHT based classification scheme, the scheme development and its modular functionalities are presented in Chapter III. MATLAB implementations of the scheme and classification results are given in Chapter IV. Conclusions and recommendations for future work are provided in Chapter V. Three MATLAB scripts for Hough features map generation, feature extraction, and the feed-forward, back-propagation neural network used in the proposed scheme are provided in Appendix A, B, and C, respectively.

II. BACKGROUND

A discussion of the Generalized Hough Transform (GHT), the Discrete Cosine Transform (DCT), and a neural network that is used in this work is provided in this chapter. The GHT is an effective computer vision algorithm for the detection of arbitrary object shapes [7], [8]. It has been suggested that GHT processing is slow and requires large memory storage [9], but these drawbacks can be mitigated with the use of a DCT filtering based denoising technique [10] and by reducing the GHT R-table to a Hough features map.

The DCT is widely used in compression standards for video and image objects [11], [12]. Its two important properties—energy compaction and decorrelation—allow independent manipulation of the energy content of images and video frames in the frequency domain to reduce the size of data to be further processed [12]; therefore, the DCT is suitable for resolving the speed and memory storage problems of the GHT in this work.

Neural networks have been used to provide excellent solutions to real-world problems in many industries, such as banking, aerospace, and defense [13]. In defense applications, the networks have been used for object discrimination, feature extraction, and image identification [13]. Neural networks are equipped with statistical tools for function fitting, pattern recognition, clustering, and time-series analysis [13]. A common neural network architecture used for pattern recognition in signal processing and other applications related to this thesis is discussed later in this chapter.

A. GENERALIZED HOUGH TRANSFORM

The generalized Hough transform requires three steps: edge detection, Hough parameter computation, and R-table generation [7].

1. Edge Detection

In this research, an edge point is defined as the point at which the pixel value or the grayscale change exceeds the required threshold value. The threshold value is defined by using a number of possible parameters measurable in imagery, e.g., gradient magnitude, grayscale, or even the change in resolution itself. The value of the gradient magnitude is used to identify any edge points. By using processing speed as the selection criterion, we can select the optimal edge detection operator and derive the gradient magnitude.

The quality of the edge points, the gradient magnitude, and gradient direction at each point are the most important features that determine the computational quality of Hough parameters discussed in the next step of the GHT approach.

In this work, the Sobel operator is selected for the detection of significant edge points because of its low computational burden [14], [15]. Although the Sobel operator is sensitive to noise [14], the DCT low-pass filter is a promising solution to denoising an image prior to the edge detection [11]. Image denoising by using the DCT filtering is a part of this research work. The DCT is discussed later in this chapter.

2. Hough Parameter Computations

The GHT algorithm requires three Hough parameters: the gradient angle ϕ, the relative distance to a reference point r, and the base angle α to generate R-table as shown in Table 1 [7]. The gradient angle is defined as a gradient direction at each edge point of the object-of-interest. D. H. Ballard and C. M. Brown [7] suggested the use of the centroid of the object as a reference point; hence, the distance r is simply the Euclidian distance between the edge point and the centroid and is called the distance to centroid throughout this thesis. The base angle is the angle between the x-axis and r measured in a counter-clockwise direction. The geometry of Hough parameters is shown in Figure 1.

Table 1. R-table

ϕ_1	$(r_1, \propto_1), (r_2, \propto_2), (r_3, \propto_3), \ldots\ldots$
ϕ_2	$(r_{20}, \propto_{20}), (r_{21}, \propto_{21}), (r_{22}, \propto_{22}), \ldots$
……	……………………………
ϕ_{360}	$(r_{40}, \propto_{40}), (r_{41}, \propto_{41}), (r_{42}, \propto_{42}), \ldots$

Adapted from: [7] D. H. Ballard and C. M. Brown, "The Hough method for curve detection," in *Computer Vision*, New Jersey: Prentice-Hall, 1982, pp. 128–131.

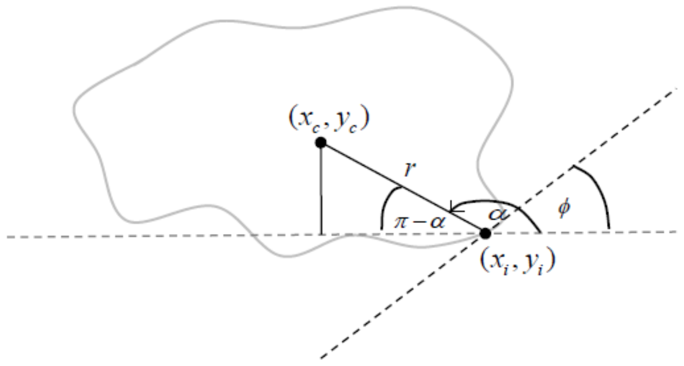


Figure 1. Geometry used to form the R-table

Adapted from: [8] "Generalized Hough Transform (GHT)," class notes for Digital Signal Processing, Dept. of Computer Science & Engineering, Indian Institute of Technology, Delhi.

3. R-table Generation

In [8], an R-table is defined as a table containing the Hough parameters ϕ and the corresponding pairs of r and α, (r, α), for each ϕ. The arrangement of elements contained in the R-table is depicted in Table 1. The R-table represents the shape of the object-of-interest. Theoretically, any object appearing in different images with the same orientation and scale should yield the same R-table. As a result, the pattern of the R-table can be used directly for the comparison or classification in this work. As a result, the pattern of R-tables of any objects from different images can be used as an object identifier to compare images and in the classification of an object(s)-of-interest.

In detecting a two-dimensional (2D) shape by using the GHT algorithm, the scaling factors and rotation angle θ of the shape with respect to its reference model must be known [7], [16]. The reference model is a selected shape of the object-of-interest with a specific scale and rotation angle.

Ideally, if the exact scaling factor and the rotation angle are known, the object-of-interest can be resized to its reference model size by a factor of $1/S$ and rotated backward by θ so that the same size of R-tables of the same object in different images can be generated prior to the classification. Feasible methods of estimation of S and θ are proposed by [17], [18] and [19], respectively. In reality, obtaining the exact scaling and rotation factors is not always possible. The estimation of these two parameters is also time-consuming. Because of the above problems, in this work we propose a new approach that simplifies the R-table for the object classification. A reformatted R-table concept is proposed and discussed in Chapter III.

B. DISCRETE COSINE TRANSFORM

In image processing, background noise is a major problem that impedes edge detection [11], [14]. Remote sensing imagery is commonly hampered by environmental noise such as brightness, rain and/or clouds and can be minimized by existing filtering techniques. In this work, the DCT filtering technique is leveraged for denoising an image. The DCT is a common tool for image compression. The most well known image compression application is the Joint Photographic Experts Group (JPEG) standard. In the first step of the JPEG compression process, the DCT is used to convert a raw image from pixel values (spatial domain) into DCT coefficients (frequency domain). Then, the low energy content (high frequency or abrupt changes of pixel values) is minimized by a quantization technique [12], [20].

In a similar fashion to the JPEG standard, the denoising filter is used to minimize the high frequency background noise in an image prior to edge detection; however, excessive filtering of high frequency content causes blurred edges in an object-of-interest in an image [21]. In this work, the size and shape of the denoising filter is investigated in order to prevent excessive filtering. To clearly understand the effects of the denoising filter, these two fundamental DCT concepts are reviewed in the following.

1. The DCT Coefficients

The equations for the 2D-DCT and 2D inverse DCT (2D-IDCT) for a matrix of $P \times P$ are as follows [12]:

$$C(u,v)=\alpha(u)\alpha(v)\sum_{i=0}^{P-1}\sum_{j=0}^{P-1} f(q,p) cos\left(\frac{\pi(2i+1)u}{2P}\right) cos\left(\frac{\pi k(2j+1)v}{2P}\right) \tag{1}$$

where $u,v = 0,1,2,...,P-1$ and $\alpha(u)$ and $\alpha(v)$ are defined as

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{P}} & for \quad u = 0 \\ \sqrt{\frac{2}{P}} & for \quad u \neq 0 \end{cases} \tag{2}$$

and

$$f(i,j) = \sum_{u=0}^{P-1}\sum_{v=0}^{P-1} \alpha(u)\alpha(v)\ \mathrm{C}(u,v) cos\left(\frac{\pi(2i+1)u}{2P}\right) cos\left(\frac{\pi(2j+1)v}{2P}\right) \tag{3}$$

where $i, j = 0,1,2,...,P-1$.

Generally, a grayscale image or the intensity layer of a color image is preferred for computer vision applications. A typical grayscale image is simply a matrix of numbers from 0 to 255. By using Equation (1), spatial-domain image data (grayscale values) can be transformed into frequency-domain data (the DCT coefficients) where values and positions in the matrix reflect energy and frequency, respectively.

The spatial data is still needed for the calculation of the geometric parameters of the GHT. The IDCT is used to reproduce the spatial data after noise minimization. This new set of spatial data is then used for calculation of geometric parameters of the GHT.

2. Frequency Interpretation of DCT Coefficients

The frequency interpretation of DCT coefficients can be viewed as shown in Figure 2. The first element in a DCT coefficient matrix is called the DC coefficient (zero frequency) which accounts for the average brightness of an image [6], [14]. The remaining elements are AC coefficients. Low frequency content is contained in the upper left corner region of the coefficient matrix. The farther the locations of a matrix element in the DCT coefficient matrix is from the DC coefficient, the higher the frequency content [12], [20].

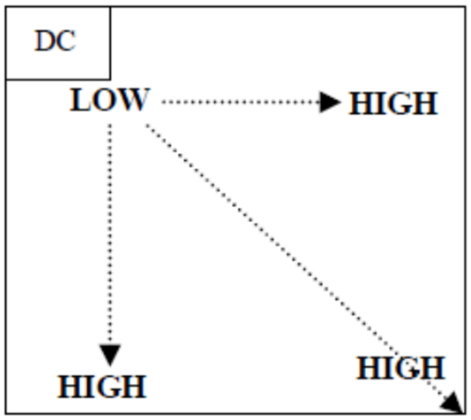


Figure 2. Frequency interpretation of DCT coefficient matrix

Adapted from: [12] S. A. Khayam. (2003, Oct. 10). The Discrete Cosine Transform. [Online]. Available: http://www.lokminglui.com/dct_tr802.pdf

S. A. Khayam [12] illustrated the distinction of energy distribution in the DCT coefficient matrix between an uncorrelated image and a correlated image as shown in Figure 3. Figure 3(a) is an uncorrelated image and Figure 3(c) is a correlated one. The energy distribution of the former is depicted in Figure 3(b), and that of the latter is in Figure 3(d). The uncorrelated image has more abrupt changes in pixel values, which means that there is more high frequency content than in the correlated one. Note that most of the energy of the two images is concentrated in the upper left corner, or the low frequency region, of the coefficient matrix. This energy distribution concept is used as a baseline for the implementation of the denoising filter described in Chapter IV.

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

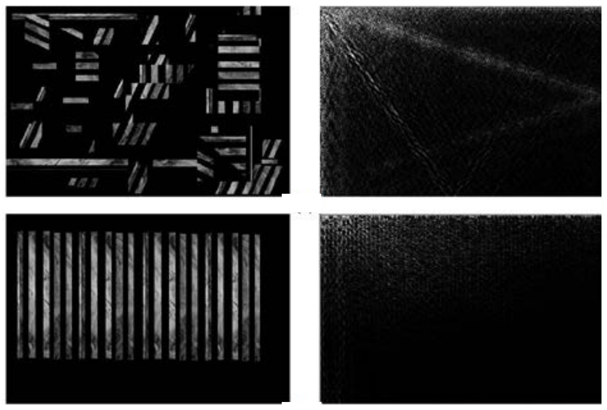


Figure 3. An example of energy distribution in an uncorrelated image (a) and a correlated image (c)

Source: [12] S. A. Khayam. (2003, Oct. 10). The discrete cosine transform. [Online]. Available: http://www.lokminglui.com/dct_tr802.pdf

C. NEURAL NETWORKS

Neural networks have gained favor in classification techniques used in remote sensing over traditional statistical approaches, such as the Euclidian, the maximum-likelihood, and Mahalanobis distance classifiers [22]. The networks have been engineered to mimic the functionalities of the human brain. Compared to traditional classifiers, neural networks are more accurate, faster, and better able to deal with data from multiple, disparate sensors.

In remote sensing, the feed-forward, back-propagation multi-layer perceptron (MLP) neural network is commonly used [22]. Its basic architecture is shown in Figure 4. The network consists of three layers: the input layer, the hidden layer, and the output layer. The nodes in the input layer represent elements of a feature vector. The hidden layer consists of nodes called hidden neurons, each of which is a weighting function. A higher number of hidden layers are known to enable the network to learn more complex patterns. The output layer represents the number of target types to be classified.

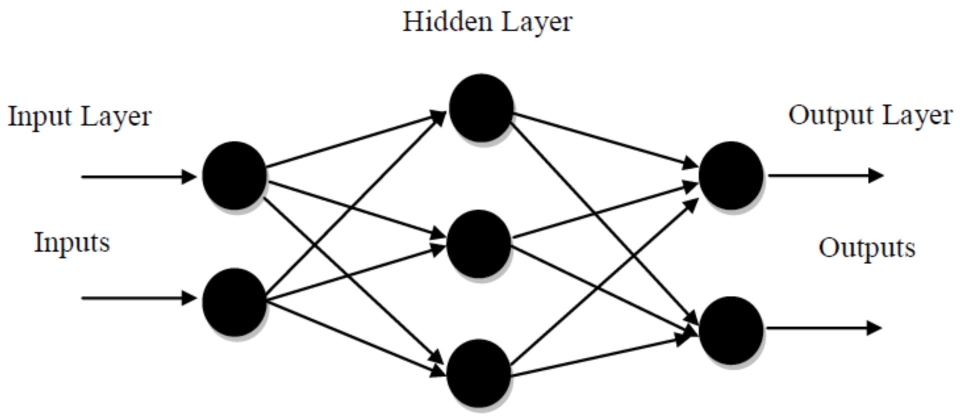


Figure 4. The MLP basic architecture

Source: [22] P. M. Atkinson and A. R. L. Tatnall, "Introduction to Neural networks in remote sensing," Int'l J. Remote Sensing, pp. 699–709, 2015.

Due to its ease of implementation and effective learning capability [22], the neural network with the MLP architecture is used as a classifier for the proposed classification scheme in this work.

The GHT, DCT, and neural network concepts that are used for building a GHT-based classification scheme were discussed above. The building blocks of the scheme and their functionalities are presented in the next chapter.

III. GHT-BASED CLASSIFICATION SCHEME

This chapter is organized into two sections. The proposed GHT-based classification scheme is presented and described in the first section. The scheme consists of five modules: the Denoising Filter, the Edge Extractor, the GHT, the Hough Features Extractor and the Neural Network. In the second section, the functionalities of each module are discussed based on four underlying concepts: the DCT, edge detection, the GHT algorithm, and the MLP. Implementation of the classification scheme and significant test results are elaborated upon in Chapter IV.

A. PROPOSED GHT-BASED CLASSIFICATION SCHEME

The schematic diagram of the proposed GHT-based classification scheme is shown in Figure 5.

1. Proposed Classification Scheme

The proposed classification scheme is based on the GHT algorithm discussed in the previous chapter. Recall that the first step of the GHT algorithm is to obtain edge points of the object-of-interest. To improve the quality of the object edges and to reduce processing time, the Denoising Filter module is used to reduce background noise. The Edge Detector computes a gradient magnitude of all image pixels and detects significant edge points of the object based on the gradient magnitudes.

The GHT module computes the following GHT parameters: the distance to the centroid, the base angle, and the gradient angle. The GHT module uses these parameters to build an R-table as shown in Table 1. The Hough Features Extractor module reformats the R-table. The reformatted R-table is called the Hough-features map in this work. The Hough features are extracted by identifying the significant peaks on the map. Finally, a neural network is used to classify the object-of-interest based on the Hough features.

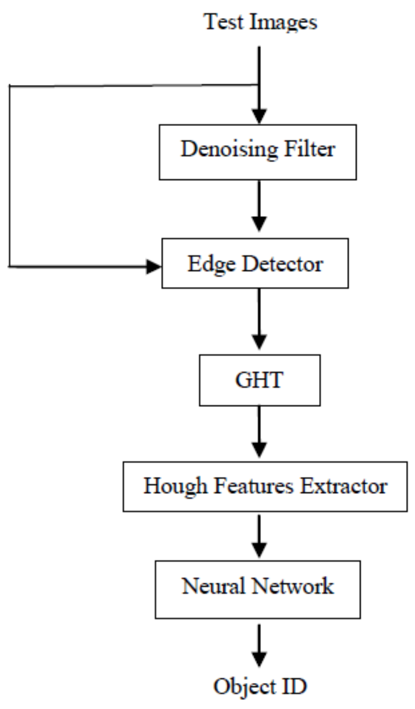


Figure 5. The proposed GHT-based classification scheme

2. Scheme Development

In order to develop the proposed classification scheme, related concepts are reviewed below.

a. Denoising Filter

The DCT coefficient thresholding techniques are commonly used for noise reduction in image and video compression standards, such as JPEG, MPEG, H.261, H263, and H.264 [11]. The DCT has two important properties: energy compaction and decorrelation [12]. Energy compaction is the property that only few coefficients (typically in the upper left corner of the DCT coefficient matrix) represent most of the energy in an image. The decorrelation property is the property that the transformed data or DCT coefficients become uncorrelated. In other words, the magnitude of each coefficient is independent of the neighboring coefficients, unlike the spatial values.

In Figure 6, the white pixels in the DCT coefficient matrix represent locations of the DCT coefficients where the energy of each DCT coefficient is above 1% of the maximum energy. Note that most of the energy of both ship images is concentrated in the low-frequency region. Consequently, the denoising filter is employed as the first module in the scheme to minimize the high-frequency noise by keeping only the high-energy data (low frequency DCT coefficients) for further processing.

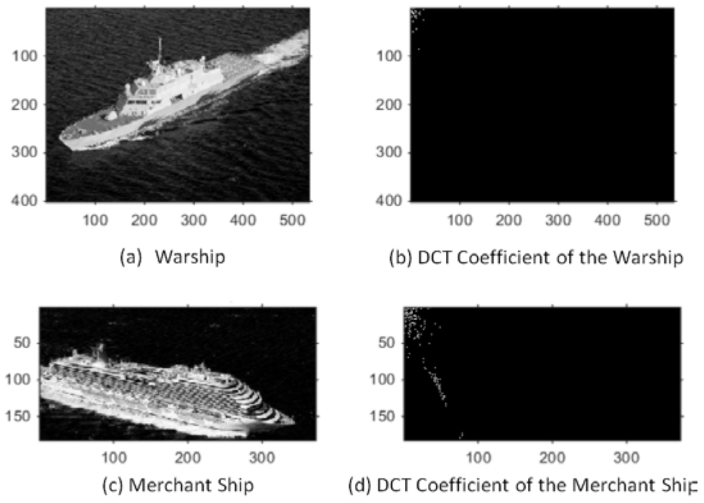


Figure 6. Investigation of energy distributions of the DCT coefficients of two ships with some background noise (i.e., waves and white caps)

b. Edge Detector

The Sobel operator, the Canny operator, and the Laplacian of Gaussian operator as described in [14], [15] are different edge detection operators. None of them is perfect in terms of noise sensitivity, processing time, or output parameters [14, Table 1, page 10]. For example, the Sobel operator is fast but sensitive to noise. The Canny operator is less sensitive to noise but time consuming. The Laplacian of the Gaussian operator gives correct edge points but malfunctions at the corners and curves.

In the proposed scheme, Edge Detector is needed to prepare edge point coordinates, which are required for GHT parameter computations. The Sobel operator is chosen for the detection of object edge points because of its low computational time [8], [9].

c. Generalized Hough Transform

The GHT algorithm has been successfully used for pattern recognition [23], detection [24], and classification [6]. Due to its high computational cost [25], a few variants of the GHT, such as the invariant generalized Hough transform [25], the randomized Hough transform [26], and the randomized, generalized Hough transform [9] have been developed to improve the detection time and storage capacity.

The use of the denoising filter reduces the number of bad edge points caused by the high-frequency background noise before the GHT parameter computations [11]. As a result, computational time is reduced; therefore, the GHT is a good algorithm for the proposed classification scheme due to its successful applications on arbitrary object shapes [6], [9], [25]. The GHT implements Hough parameter computations and R-table generation.

d. Hough Features Extractor

Before the extraction of Hough features, all R-tables for objects to be classified must be reformatted as mentioned in the previous chapter. The reformatted R-table, called a Hough features map in this thesis, contains Hough features or dominant peaks. The concept used for reformatting the R-table and feature extraction is described in the next section.

e. Neural Network

The neural network is an important classification tool to aid in the human decision making process in industry, business, and science [27]. Broadly, there are four advantages of neural networks: self-adaptive machine learning to adapt to various types of data, universal parameter estimation for a variety of functions with desirable accuracy, flexible modeling of nonlinear, complex relationships and effective classification for statistical analysis [27].

References [5] and [28] attained good outcomes when applying the neural network as the classification tool for arbitrary shapes of objects-of-interest; therefore, the neural network classifier is a desirable option for the proposed scheme.

B. FUNCTIONALITIES OF EACH MODULE

This section is intended to provide a functional description and mathematical operations of module of the scheme.

1. Denoising Filter

Three steps are performed by this module. First, a denoising filter extracts the first layer or the intensity layer of an input image. Second, the module performs a 2D-DCT on the layer to transform all pixel values or spatial data into DCT coefficients or frequency-domain data. In the last step, the module filters out the coefficients in the high frequency region and then inverse-transforms the filtered data into spatial data by using a 2D-IDCT.

2. Edge Detector

In this module, the Sobel operator is used to obtain gradient magnitudes for the detection of object edge points. There are two Sobel operator kernels, S_{bx} and S_{by}, as shown in Figure 7. Each kernel is used to convolve with the preprocessed image from the Denoising Filter to compute the horizontal and vertical change, G_x and G_y, at each image pixel, respectively [14]. Then, the gradient magnitudes $|G|$ are calculated as

$$|G| = \sqrt{G_x^{\,2} + G_y^{\,2}}\,. \tag{4}$$

The gradient magnitudes are normalized to the range 0 to 1. Thereafter, only pixels of magnitudes greater than a required threshold are considered significant edge points, which are given a value of 1. The remaining pixel values are set to 0. Finally, coordinates of the binary edge points (x_i, y_i) are obtained.

-1	0	+1
+2	0	+2
-1	0	+1

S_{bx}

+1	+2	+1
0	0	0
-1	-2	-1

S_{by}

Figure 7. Two kernels of Sobel operator

Source: [14] R. Maini, "Study and Comparison of Various Image Edge Detection Techniques," *Int'l J. Image Processing IJIP*, vol. 3, no. 1, pp. 1–12.

3. Generalized Hough Transform

The GHT module receives significant edge point coordinates from the Edge Detector. Based on the edge points, the GHT computes the object centroid, the distance to the centroid and the base angle in sequence. Finally, the gradient direction at each edge point is calculated so that an R-table can be generated as shown in Table 1.

a. Object Centroid

A conventional centroid of a 2D object (x_c, y_c) is simply an average of all edge points (x_i, y_i). Because the denoising filter only minimizes the background noise but does not completely eliminate it [29], the location of an object centroid is still affected by background noise. To enhance the accuracy of the centroid location in the presence of noise, a multi-region property measurement algorithm available in MATLAB is used.

b. Distance to Centroid

The distance between two points (x_c, y_c) and (x_i, y_i) on a Cartesian coordinate plane can be easily computed by using the Euclidian distance; therefore, the distance to the centroid r_i for the i^{th} edge point (x_i, y_i) is calculated as

$$r_i = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2} \tag{5}$$

c. Base Angle

From Figure 1, α is the angle between the *x*-axis and the distance-to- centroid in the counter-clockwise direction. The mathematical expression for the base angle in degrees is:

$$\alpha = \begin{cases} \tan^{-1}\left(\dfrac{y_c - y_i}{x_c - x_i}\right), & y_c - y_i \geq 0 \\ 360 + \tan^{-1}\left(\dfrac{y_c - y_i}{x_c - x_i}\right), & y_c - y_i \leq 0 \end{cases} \tag{6}$$

d. Gradient Angle

Changes in horizontal and vertical directions at each edge point are computed from the original image. Then the gradient angle is calculated using

$$\phi = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) \tag{7}$$

where Δx and Δy are changes in the horizontal and vertical directions, respectively.

e. R-table Generation

An R-table is created based on three parameters: ϕ, α, and r. Pairs of α and r at each object edge point are categorized by their corresponding ϕ. A general format for the table is shown in Table 1.

4. Hough Features Extractor

The Hough features extractor performs three major operations. The first operation is to reformat the R-table from the previous step into a square matrix of 360 × 360 whose row and column values correspond to the gradient angle ϕ and the base angle α, respectively. To do so, a zero matrix of 360 × 360 is established as a map template. Thereafter, the Hough Features Extractor sorts out by row all the (r,α) pairs in the R-table which contain the same α. Then it maps the total counts of α occurrences into the template at a coordinate (ϕ,α). For example, if $\alpha = 185$ occurs 50 times in the fifth row of the R-table, the value at a coordinate (5, 185) of the map is set to 50.

The second operation is to reduce the map dimensions. To resize the map, the Hough Features Extractor divides the mapped template into square blocks of $n \times n$, where n is any divisor of 360; averages all the elements in each block; and remaps the average into a new template of $N \times N$, where $N = 360/n$. The remapping process is shown in Figure 8.

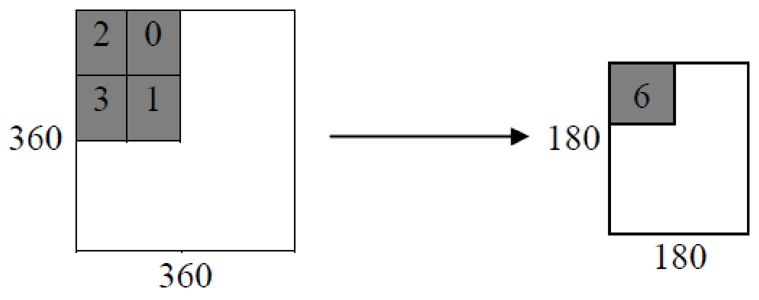


Figure 8. Remapping process when n = 2

The last operation is to extract coordinates of dominant peaks on the Hough features map and arrange them in a feature vector, which is the required input format for the neural network.

5. Neural Network

Prior to the classification, the Neural Network module has to be trained by a set of features of known class so that a good statistical model is developed and the network can predict outputs from inputs [22]. For the MLP, the network computes the error for each input feed and then back-propagates it for correction until the network achieves an acceptable state [22].

After obtaining the acceptable learning state, the Neural Network module is ready for classification. In the proposed scheme, a feature vector produced by the Hough Features Extractor module is fed into the trained network. The network compares the input features against the trained model to produce the class of the object [22].

The block diagram of the proposed GHT-based classification scheme, the scheme development, and the functionalities of each module in the scheme were discussed above. Implementation and classification performance results of the scheme are described in next chapter.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. IMPLEMENTATION AND RESULTS

This chapter is organized into two sections. In the first section, implementation of the five modules of the proposed classification scheme presented in Chapter III is described. Results of different classification cases are presented in the second section.

A. IMPLEMENTATION

This section is intended to provide implementation details of the proposed classification scheme depicted in Figure 5. The MATLAB version 2014b is the programming language used for the implementation. Some built-in functions in its image processing toolbox—`imread.m`, `size.m`, `gradient.m`, `edge.m`, `dct2.m`, `idct2.m`, `distance.m`, `atan2d.m`, `regionprops.m` and `findpeaks.m`—are utilized for the implementation of the first four modules. For the neural network classification, the pattern recognition tool in the Neural Network Toolbox is used.

In practice, the Denoising Filter and the Edge Detector are implemented together because the output of the Edge Detector, i.e., edge point coordinates, is dependent on the performance of the Denoising Filter. Likewise, the Hough Feature Extractor needs the output (R-table) from the GHT; therefore, the two are combined to simplify implementation.

The objective of this research is to design an automated classification system for aerial sensing imagery. Due to limited access to aerial sensing images, the author has acquired online aerial images of ships, aircraft, and clouds as an alternative for designing the system and demonstrating the concepts presented in Chapter III. In addition to these images, a set of miniature car images taken in a controlled environment is also used for proof-of-concept. The preparation of the car images is discussed later during the discussion of the implementation of the proposed edge detection algorithm and the proposed Hough features extraction algorithm.

1. Edge Detection

As mentioned at the beginning of this chapter, the Denoising Filter and the Edge Detector are implemented together in MATLAB codes. To obtain an effective size and shape for denoising filter, the energy distribution of the DCT coefficients of all images and the output edge points obtained after filtering are utilized.

a. Denoising Filter

As described in Chapter III, the energy distribution in the DCT coefficient matrix of each image can be used to determine the size and shape of the denoising filter. In this case, all aerial images are transformed into DCT coefficient matrices so that their energy distribution can be observed. An example of the energy distribution of three images is shown in Figure 9.

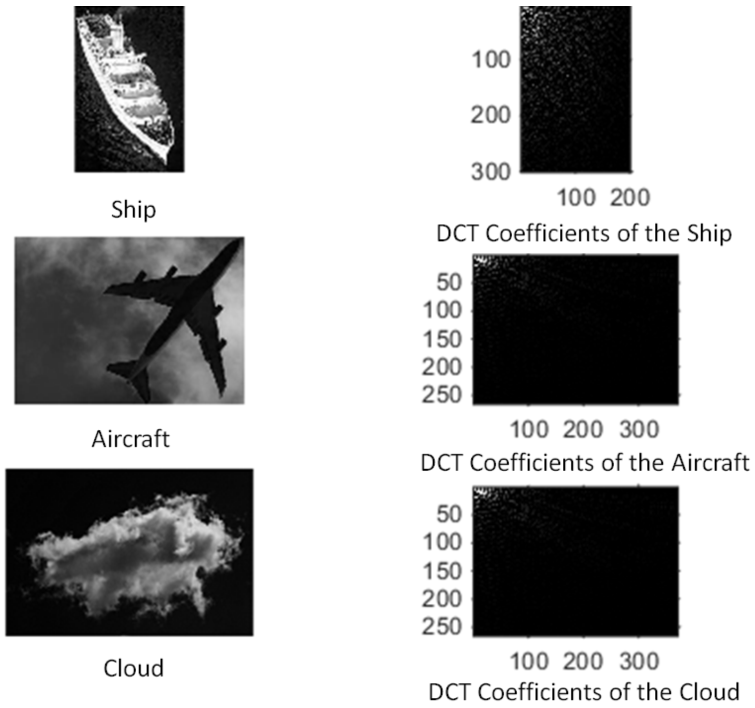


Figure 9. An example of the energy distribution of three objects in the frequency domain

After observing all of the collected images, the energy is most likely to be contained within a square window size between 50 × 50 and 200 × 200 depending on the image size and the scale of the object contained in the image. The denoising filter size of $L \times L$ is defined as a matrix that has the same dimension as the image and contains all zeros with the exception of ones inside the window as shown in Figure 10.

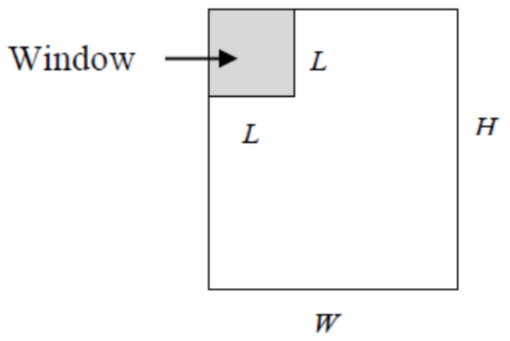


Figure 10. The denoising filter size of $L \times L$ for the image size of $W \times H$

Another investigation is to check the effect of a change in the image size or object scale. To do so, some of the images are resized up to five times their original size by using the `imresize.m` function in MATLAB and then transformed to obtain their DCT coefficients for energy observation. According to Figure 11, we see that the energy in the high frequency region of the DCT coefficient matrix obviously expands as the image size or the object scale increases.

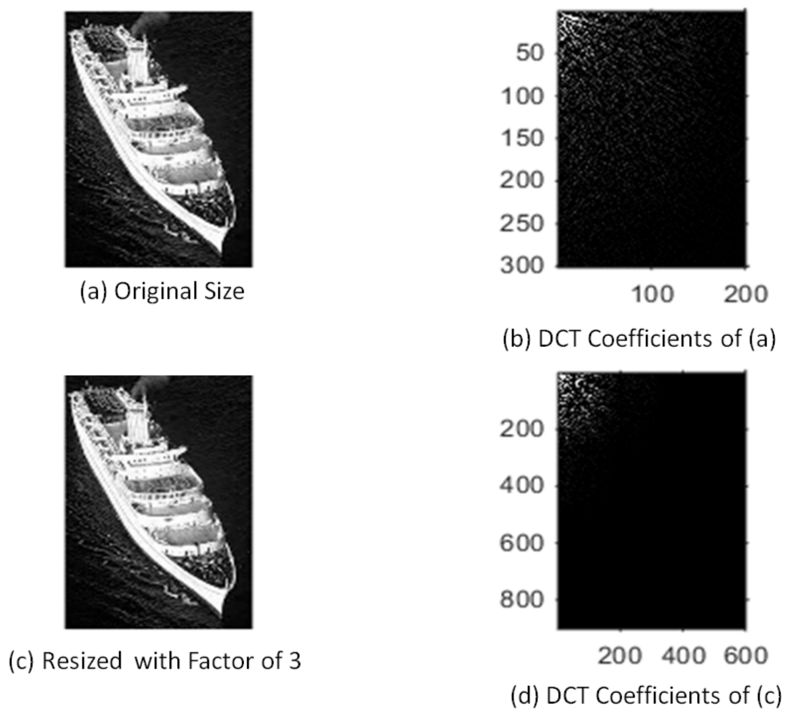


Figure 11. An example of a change in the energy distribution of resized images

To accommodate the variations of image size and object scales among the collected images, different denoising filter sizes with respect to the smaller dimension of an image have been used. The details of the denoising filter are discussed in the Edge Detector section.

b. Edge Detector

In this part, a built-in MATLAB function, `edge.m`, is used to detect and obtain the binary edge points of an object-of-interest based on the Sobel operator and the optimal thresholding algorithm. To verify that the use of DCT filtering improves the quality of edge points, the object edges detected in the test images after applying the denoising filter must be observed.

In real-world problems, a particular photographic device produces a fixed image size and only the scale of the object contained in the image changes due to the variations of distance or optical zoom. Consequently, some images of miniature cars taken by an iPhone-5 camera in a controlled environment are used for fair testing. The images have been prepared such that they can be assumed to have the same quality and background noise. Two significant assessments of the proposed edge detection algorithm are as follows.

(1) Denoising

This test is to assess the performance of the denoising filter for noise reduction and the quality of the object edges after filtering. Different filter sizes are applied to the car images. Then the `edge.m` function with the Sobel edge detection algorithm is used to detect the object edges in the filtered images. The proposed edge detection algorithm is shown in Figure 12.

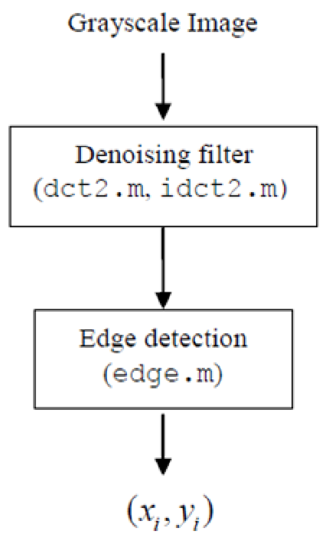


Figure 12. Flowchart of the proposed edge detection algorithm

The best visual edges of the car are obtained when the filter size is 120 × 120. According to Figure 13, we see that, in this particular example, the use of the denoising filter effectively minimizes the high frequency background noise. Although some of the car edge points are lost due to filtering, the edge points fairly delineate the shape of the car. Furthermore, the number of edge points is noticeably reduced after denoising. As a result, processing time required for generating the R-table is reduced.

(2) Scaling

This test is to verify that the denoising filter size of 120 × 120 works effectively on the same object with different scales. Two images of the same miniature car with different scales are denoised as shown in Figure 14. In this particular case, noise has been nearly eliminated and the object edges look very reasonable. It is observed that the larger the size of the object, the more the edge loss. This phenomenon makes sense because the bigger object has more edges, or high frequency content information.

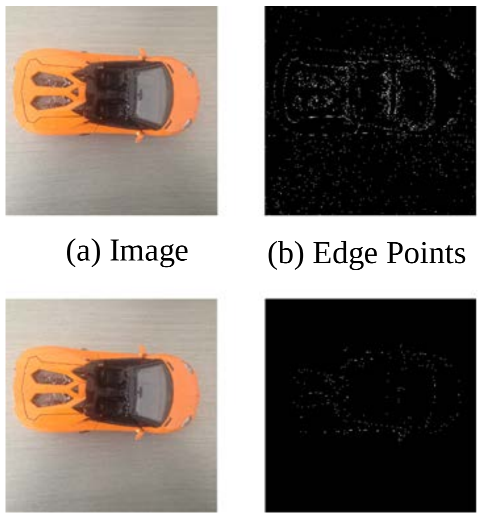
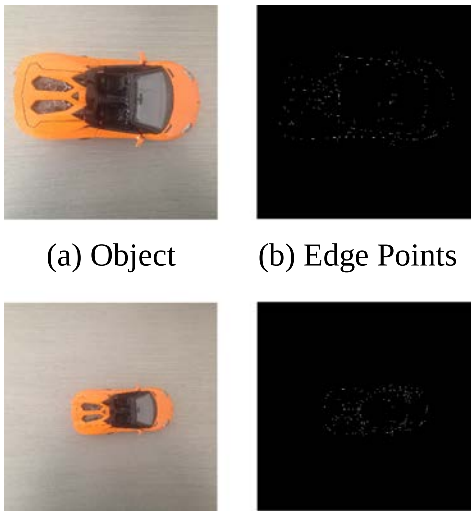


Figure 13. Edge detection of the same image: (b) without denoising filter and (d) with denoising filter size of 120 × 120



(a) Object (b) Edge Points

(c) Object (d) Edge Points

Figure 14. Edge detection of the same image with different scales by using denoising filter size of 120 × 120: (b) S =1 and (d) S =0.5

The filter size of 120 × 120 is not suitable for all the images due to the variations in size and quality; therefore, through experimentation a table of five empirical DCT filter sizes was obtained as shown in Table 2.

Table 2. Table of empirical denoising filter sizes with respect to the smaller dimension of images

Image Size (smaller dimension)	Filter Size
≤ 250	80 × 80
≤ 1000	100 × 100
≤ 2500	120 × 120
≤ 4000	140 × 140
Otherwise	160 × 160

2. Hough Features Extraction

In this section, the implementation of the GHT and Hough Features Extractor modules are discussed. MATLAB scripts from [30] have been modified to compute all necessary Hough parameters and build R-tables. The `ModelHough.m` [30] is used for the generation of R-tables. The GHT algorithm is shown in Figure 15. Thereafter, to reformat the R-table into the Hough-features map and extract the Hough features, two author-developed functions, `Hough_Features.m` and `SearchPeaks.m`, are used. MALAB scripts of these functions are provided in Appendix A and B, respectively.

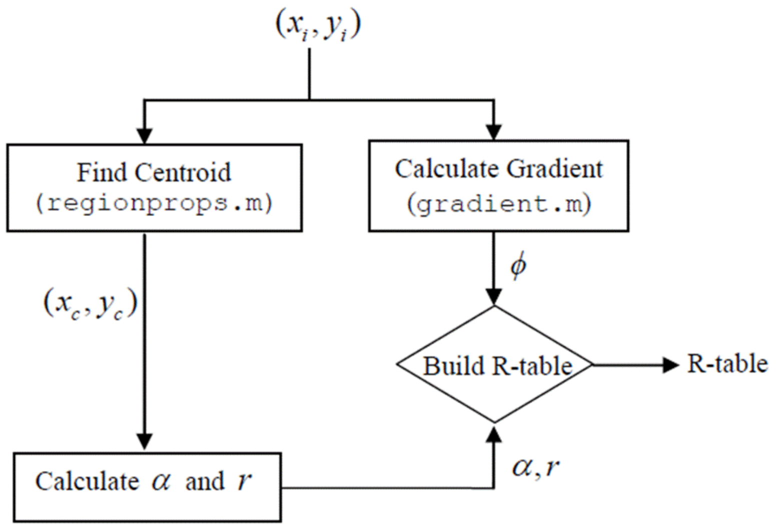


Figure 15. Flowchart of the GHT algorithm performed by `ModelHough.m`

a. GHT Algorithm

To generate the conventional R-table as in Table 1, three Hough parameters need to be computed: the distance to centroid r, the base angle α, and the gradient direction ϕ of each edge point. `ModelHough.m` leverages the `regionprops.m` function in MATLAB to find regional centroids of the binary object edges (x_{c_n}, y_{c_n}) and uses their average as the object centroid (x_c, y_c). The centroid is then used to calculate r and α based on Equations (5) and (6), respectively. In the meantime, the gradient directions are computed based on the pixel values of the original image by using `gradient.m`. Lastly, `ModelHough.m` arranges all parameters into the R-table format.

Building an accurate R-table is not an easy task because the computations of α and r are dependent on the location of the object centroid, which is sensitive to noise; therefore, `regionprops.m` is used to enhance the accuracy of the centroid location. Once the centroid location is deemed accurate, then the accurate distance to centroid can be computed by using `distance.m`. To compute α, the quadrant of the edge points with respect to the centroid must be taken into account. To solve the quadrant ambiguity and simplify the computation of α, `atan2d.m` is used.

After obtaining all Hough parameters, the R-table is formed. A three-dimensional (3D) matrix is needed for storing both parameters. To do so, a zero matrix of $360 \times T \times 2$ is established, where T is the number of detected edge points. The first layer of the matrix is used for storing r and the second layer for α.

b. Hough Features Extraction Algorithm

The `Hough_Features.m` function requires a block size *n* from a user to build the Hough features map as illustrated in Section B.4 of Chapter III. Thereafter, dominant peaks on the map are extracted and arranged into a row, vector format using `SearchPeaks.m`. The Hough features extraction process is depicted in Figure 16.

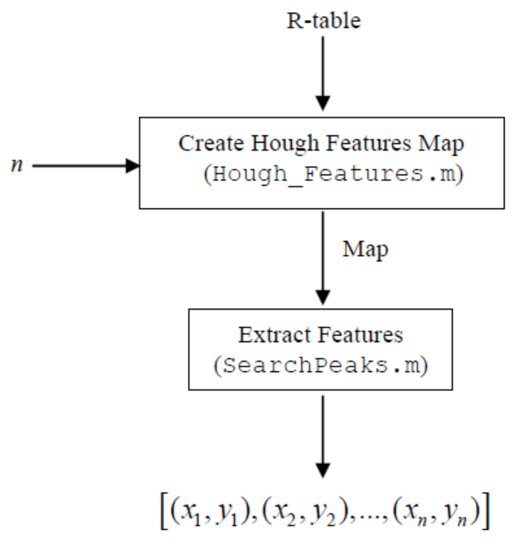
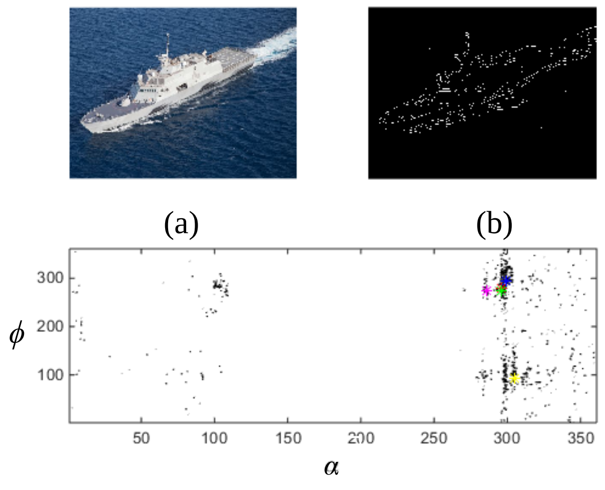

R-table
n
Create Hough Features Map
(Hough_Features.m)
Map
Extract Features
(SearchPeaks.m)
$[(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$

Figure 16. Flowchart of Hough-features extraction algorithm

With two inputs, R-table and *n*, the `Hough_Features.m` function performs three operations. The first operation is to reformat the R-table into the features map of 360 × 360 or *n* = 1 as shown in Figure 17. Note that the dominant peaks are marked by colored asterisks.



(c)

Figure 17. An initial Hough-features map of a warship : (a) original image; (b) detected edge points; and (c) the Hough-features map

The second operation is to reduce the dimension of the Hough-features map to $N \times N$, where $N = 360/n$. The resized Hough-features map, for n = 12, is shown in Figure 18.

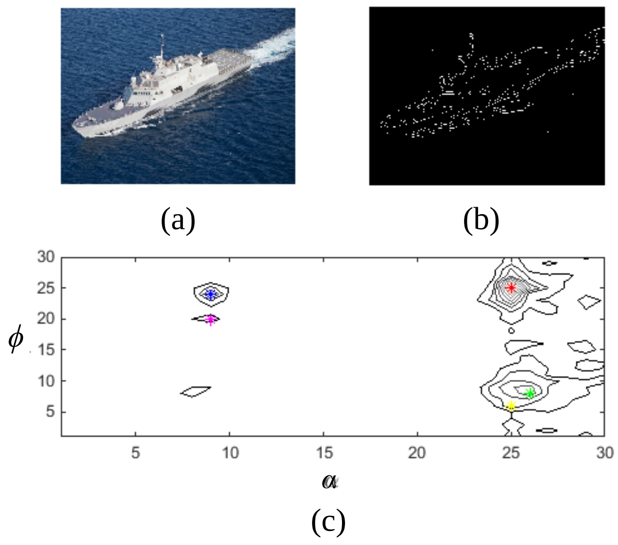


Figure 18. A reduced Hough-features map of a warship with n = 12: (a) original image; (b) detected edge points; and (c) the Hough-features map

The last operation is to extract the dominant peaks on the map. To detect the peaks, `SearchPeaks.m` is developed based on `findpeaks.m` in MATLAB. The dominant peaks can be located by using `findpeaks.m` to search for local maxima in each row and column of the map. The dominant peaks are the local maxima detected in both searches.

This peak detection algorithm occasionally produces false peak(s). For example, in Figure 18 the green peak is the true one, but the yellow is not; therefore, the algorithm must be modified to resolve the problem. A possible solution is to average all peaks that fall within the same cluster.

c. Testing of the Hough Feature Extraction Algorithm

The following are four test cases for the Hough-feature extraction algorithm.

(1) Two Similar Objects

This test case is to compare Hough-features maps of two similar objects. Images of two miniature cars (white and orange) are taken with the same camera and have the same background as shown in Figures 19(a) and (d). The edge detection results are shown in Figures 19(b) and (e). The maps of both cars are quite similar as seen in Figures 19(c) and (f); however, the peak locations are different. This implies that the peak features are distinct and can be used for classification of similar objects.

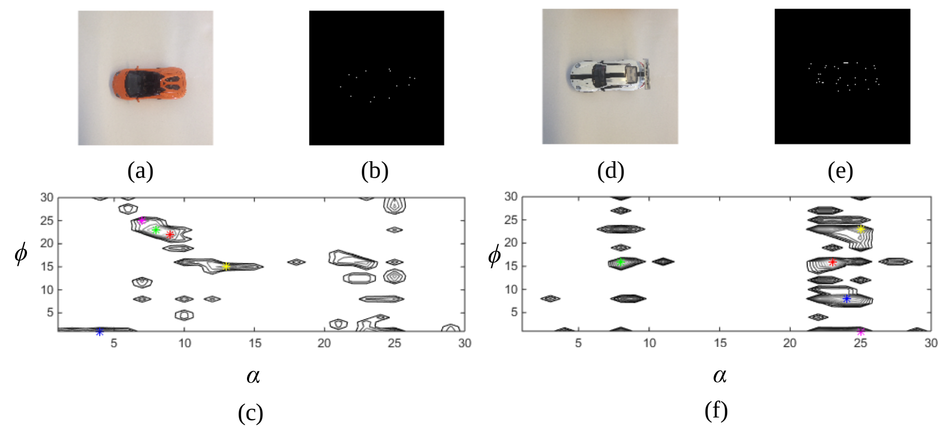
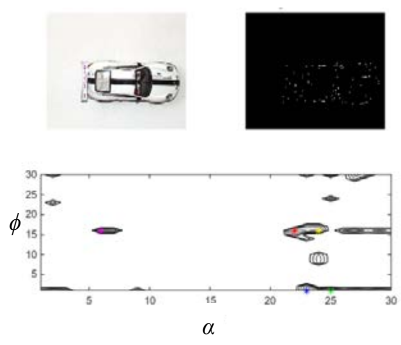


Figure 19. Hough-features map and peaks of similar shaped objects (orange and white cars)

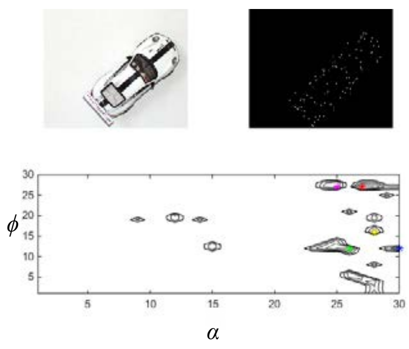
(2) Rotation-invariance

This case is to investigate whether the Hough features are rotation-invariant. To do so, Hough features maps of the same object with five different rotation angles of 0, 45, 90, 135, and 180 degrees as shown in Figure 20 are used for comparison. For each case, the figure includes the image of the object, its edge points and the Hough features map.

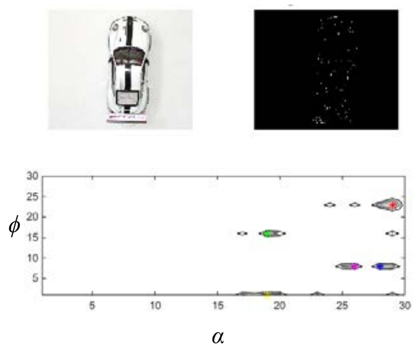
As seen in Figure 20, no distinct or common pattern emerges among the five maps. Most peaks tend to shift when the object is rotated. This indicates that the peak features are not rotation-invariant.



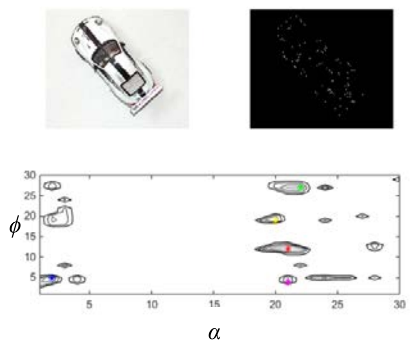
(a) Rotation Angle of 0°



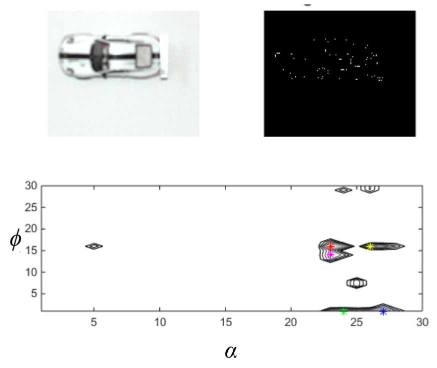
(b) Rotation Angle of 45°



(c) Rotation Angle of 90°



(d) Rotation Angle of 135°

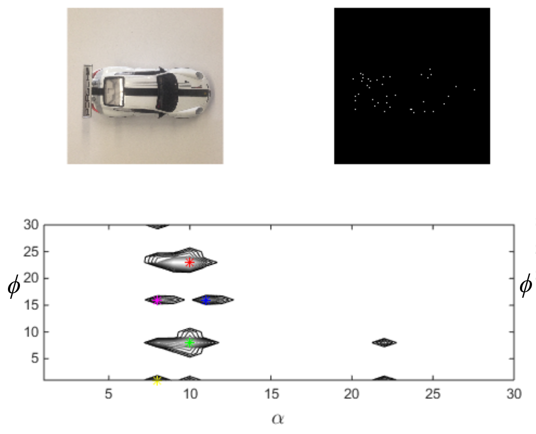


(e) Rotation Angle of 180°

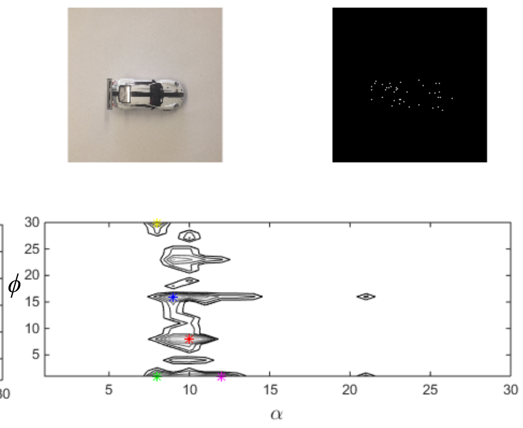
Figure 20. Hough-features maps of five different orientations of the same object

(3) Scale-invariance

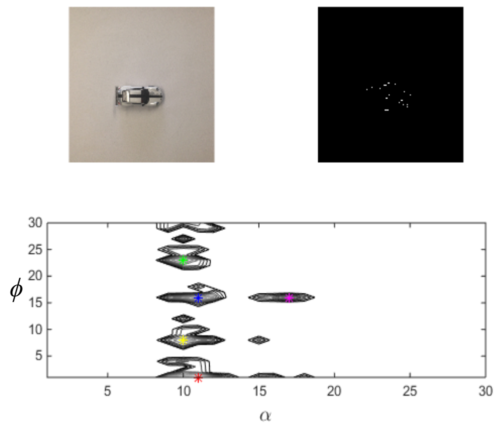
This case is to investigate whether the peak features are scale-invariant. The maps of three images of the same object with scales of $S = 1$, 5/8, and 3/8 as shown in Figure 21 are used for comparison.



(a) Original ($S = 1$)



(b) $S = 5/8$



(c) $S = 3/8$

Figure 21. Hough-features maps of three different scales of the same object

From Figure 21, three maps of the scaled objects look very similar and most of their dominant peaks are in the same location or almost collocated. Consequently, the Hough features are likely scale-invariant.

(4) Hough-Features Map of Four Different Objects

This case is to ensure that different objects with the similar orientation do not have the same Hough features. Four objects used for the test are shown in Figure 22. For each case, the figure includes the image of the object, its edge points and the Hough-features map. Although the four images vary in size and background, the object edges are visually well defined as seen in Figure 22; therefore, their features can be used for fair comparison.

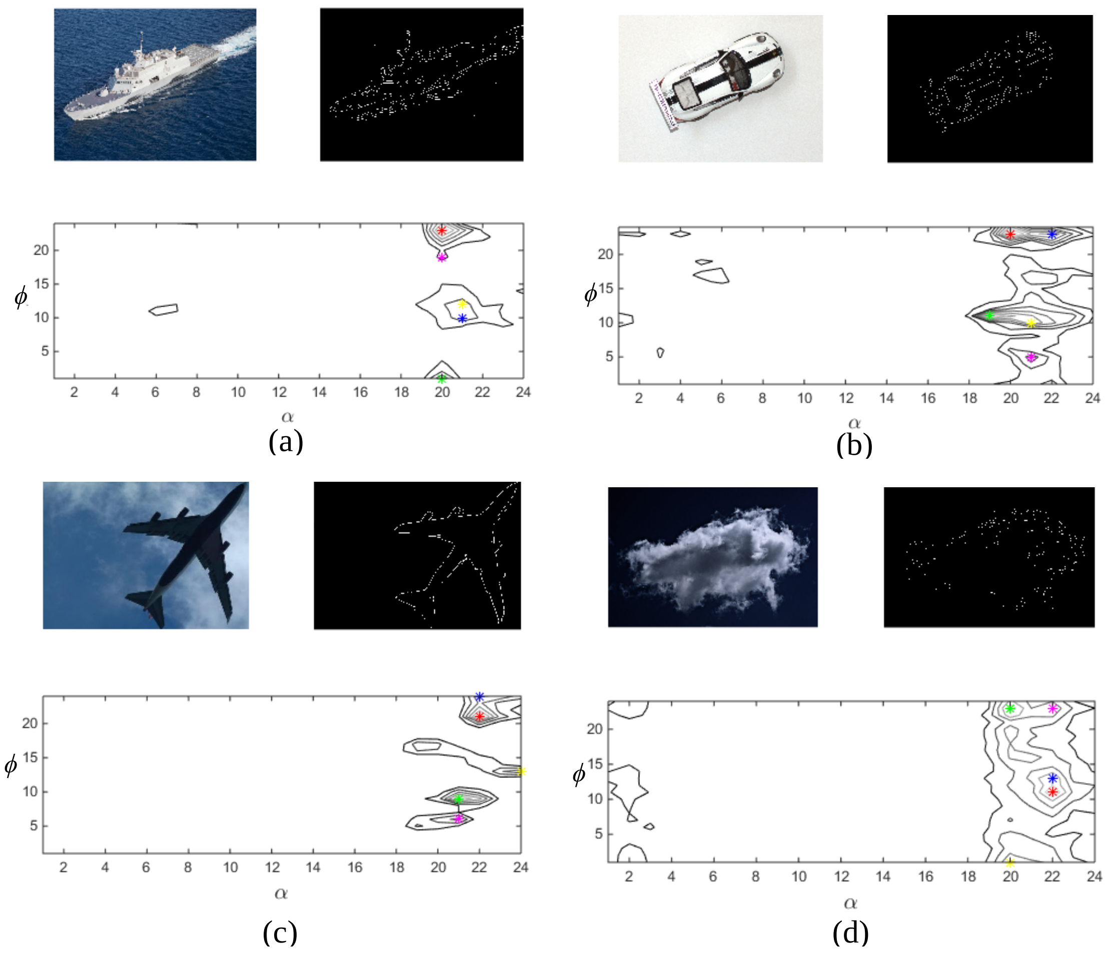


Figure 22. The Hough-features maps of four different objects with similar orientations

From Figure 22, the Hough features of each object are concentrated on the right hand side of the Hough-features map, but none of them have more than three peaks in common. For instance, the ship has two peaks in common with those of the car, three peaks in proximity of those of the plane, and two peaks in common with and one in proximity of the cloud.

3. Neural Network Classification

There are many options for using the Neural Network toolbox for the classification in MATLAB. The most convenient one is to use a Graphical User Interface, which does not allow a user to change certain options, such as training and performance functions, and options for the percentage of training, validation, and testing images are predefined. For example, the user can select only a percentage that is a multiple of five.

In this research, the neural network is implemented by using a MATLAB script, which allows the user to change parameters. The neural network script generated by the Neural Pattern Recognition application is modified and named `NN_Trial.m`. The file is used for all the classification cases in the next section.

`NN_Trial.m` uses the MLP structure with the scaled conjugate-gradient backpropagation training algorithm. The percentage of training, validation, and testing images are set to 90, 5, and 5, respectively. The only variable in `NN_Trail.m` is the number of hidden neurons, which can be adjusted to improve classification performance.

The MLP structure in the MATLAB Neural Network toolbox is shown in Figure 23. A matrix of the Hough features is fed into the input node. The hidden layer performs the analysis based on the classification settings, e.g., the training function and the number of hidden neurons, to develop the best prediction model for a particular set of training features.

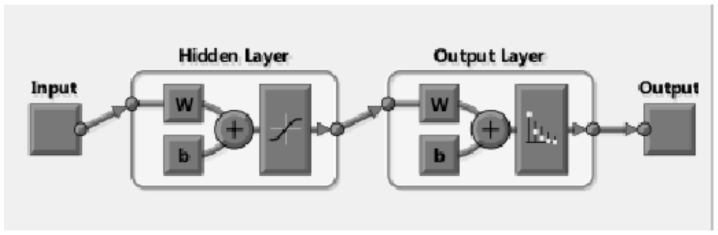


Figure 23. The MLP structure in MATLAB

Source: [13] M. H. Beale, M. T. Hagan, and H. D. Demuth. (n.d.). Neural network toolbox getting started guide. [Online]. Available: http://www.mathworks.com/help/pdf_doc/nnet/nnet_gs.pdf [Dec 8, 2015].

At completion of training, the program `NN_Trial.m` obtains a trained network with a prediction model. This trained network is then used to classify test features. The MATLAB script for NN_Trial.m is in Appendix C.

B. CLASSIFICATION RESULTS

MATLAB simulation results of the proposed classification scheme on two sets of images are presented in this section. The first set comprises 60 silhouettes of perfect shapes, and the second consists of 64 aerial images of maritime-domain objects, i.e., ships, aircraft, and clouds. Six binary classification cases, one three-class classification case, and one four-class classification case are investigated on the silhouettes. For maritime-domain images, classification results include three binary classification cases and one three-class classification case.

1. Classification of Perfect Shapes

The performance of the proposed GHT-based classification scheme on the perfectly shaped images of triangles, squares, circles, and stars is investigated in this section. These images are prepared by using the Windows paint program. The four objects-of-interest are different in size, orientation and location. There are 15 images of each shape; ten images are selected for training and the rest for testing. Some sample images are shown in Figure 24. Eight classification cases—six binary cases, a three-class case, and a four-class case—are investigated. For all of these eight classification cases, a denoising filter was not used because the perfect shape images have no background noise.

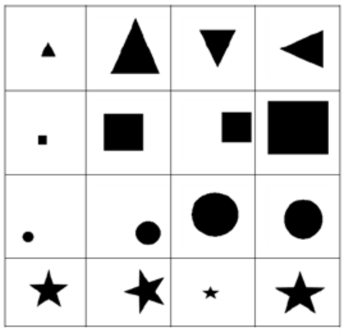


Figure 24. Some samples of four perfect shapes

a. Binary Classification

In this case, silhouettes of two shapes are selected for testing. There are six possible combinations: triangle versus square, triangle versus circle, triangle versus star, square versus circle, square versus star, and circle versus star. For each combination, the training and testing samples consist of ten images and five images of each shape, respectively.

Six dominant peaks extracted from the Hough features map size of 20 × 20 are used as input data and the hidden layer size of the neural network is 15. The classification performance on the first two combinations is 100%, while that of the remaining combinations is 90%.

b. Three-Class Classification

In this case, the training and testing samples consist of ten images and five images from each of three different shapes (triangles, squares, and circles), respectively. The images of the triangles, squares, and circles are assigned a class of 1, 2 and 3, respectively. The best performance of 86.7% is obtained by using six peaks extracted from the Hough features map size of 20 × 20 and the hidden layer size of 15 hidden neurons. The scheme can classify all triangles and squares correctly, but two of the circles are misclassified as a triangle and a square.

This ambiguity is possibly caused by some distortion introduced in some circles when drawn, and circles of very small size can be very similar to some triangles and squares.

c. Four-Class Classification

This classification case is analyzed like the previous case. The only difference is that images of stars are included in an equal amount and assigned a class of 4. The best confusion matrix is shown in Figure 25.

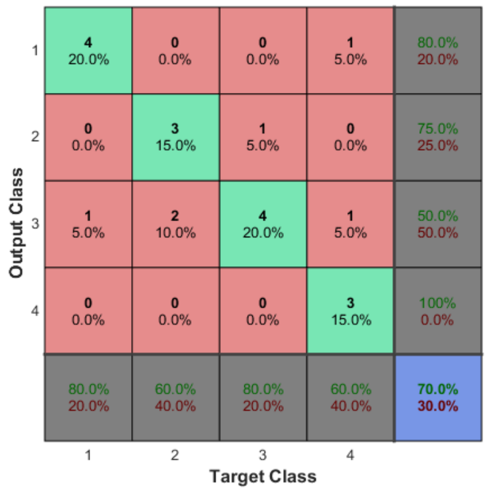


Figure 25. The best confusion matrix of four perfect shapes

According to Figure 25, we find that the most correctly classified shapes are triangles and circles with a classification performance of 80% (the first and third columns of the confusion matrix). There is ambiguity between squares and circles; therefore, only three out of five squares are correctly classified as indicated in the green cell of the second column.

Classification performances of all perfect shape cases are summarized as in Table 3. Note that as the number of objects to be classified increases, the classification performance decreases.

Table 3. Classification performances of all perfect shape cases

Cases	Performance (%)
Triangle versus Square	100
Triangle versus Circle	100
Triangle versus Star	90
Circle versus Square	90
Circle versus Star	90
Square versus Star	90
Triangle, Square, Circle	86.7
Four shapes	70

2. Classification of Maritime Domain Images

How distinct Hough features can be used by the neural network to differentiate typical objects in maritime domain images, such as ships, aircraft and clouds, is investigated in this section. Four cases are investigated: three binary classification cases and one three-class classification case.

Sixty-four images consisting of 23 ships, 25 aircraft, and 16 clouds are used for training. The test images comprise 12 ships, 12 aircraft, and 11 clouds. All images are chosen from the set of online images collected for the implementation of the scheme. Some samples of the images used in this classification are shown in Figure 26.

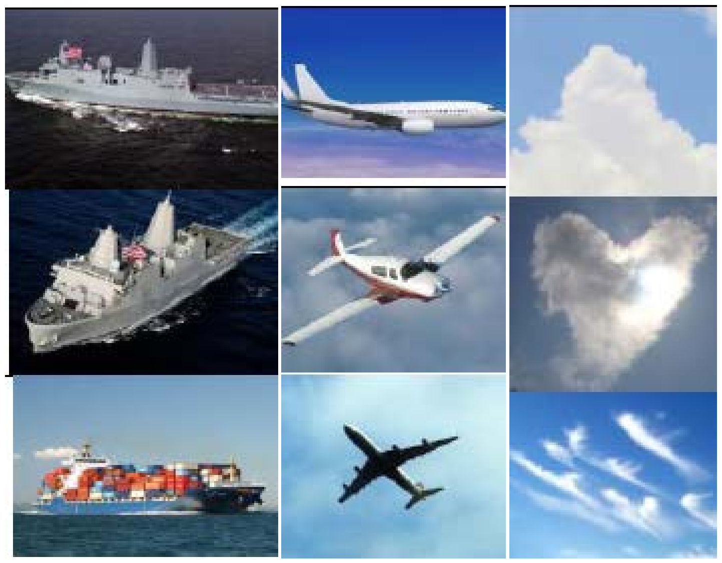


Figure 26. Some samples of maritime domain images

a. *Binary Classification*

Three combinations are tested: ships versus clouds, ships versus aircraft, and ships versus the other two. The highest classification results are obtained by using six dominant peaks extracted from the Hough-features map size of 20 × 20 and a hidden-layer size of 15 hidden neurons.

The scheme's classification results for the three cases are 87% and 79.2% and 82.9%, respectively. This indicates that aircraft are more similar to ships than clouds. Lastly, a binary classification case between ships and the others is tested. Aircraft and clouds are treated as a non-ship class. The performance to classify ships was found to be 82.9%.

Further tests are conducted to investigate the effects of denoising filter sizes, the number of features (peaks), the number of hidden neurons and the size of Hough-features map on classification performance and to determine the desirable settings for the proposed classification scheme.

To determine an optimal Hough-features map size, nine different map sizes (18 × 18, 20 × 20, 24 × 24, 30 × 30, 36 × 36, 45 × 45, 60 × 60, 90 × 90 and 180 × 180) were tested. The number of hidden neurons was fixed at 15. The classification results were averaged over 1000 trials. In each run, the `NN_Trial.m` function randomly initialized a new hidden layer so that the initial weighting function in the hidden layer was different from the previous run. This randomness enabled the neural network to generate a diverse set of prediction models. The classification performance results based on averaging results of 1000 trials for different map sizes are shown in Figure 27. For each map size, the scheme was tested with a different number of peaks up to a maximum of eight peaks.

According to Figure 27, the best average performance occurs when the size of the map is 20 × 20 and the number of peaks is six. Such a setting yields a classification performance of 77.8 %. Note that these results are an average and only useful for performance comparisons. The best prediction model can be decided based on the run that produces the highest classification percentage.

Using six peaks from the map size of 20 × 20 and varying the number of hidden neurons from 3 to 30, we found the highest average performance occurred when the number of hidden neurons was 15, as shown in Figure 28.

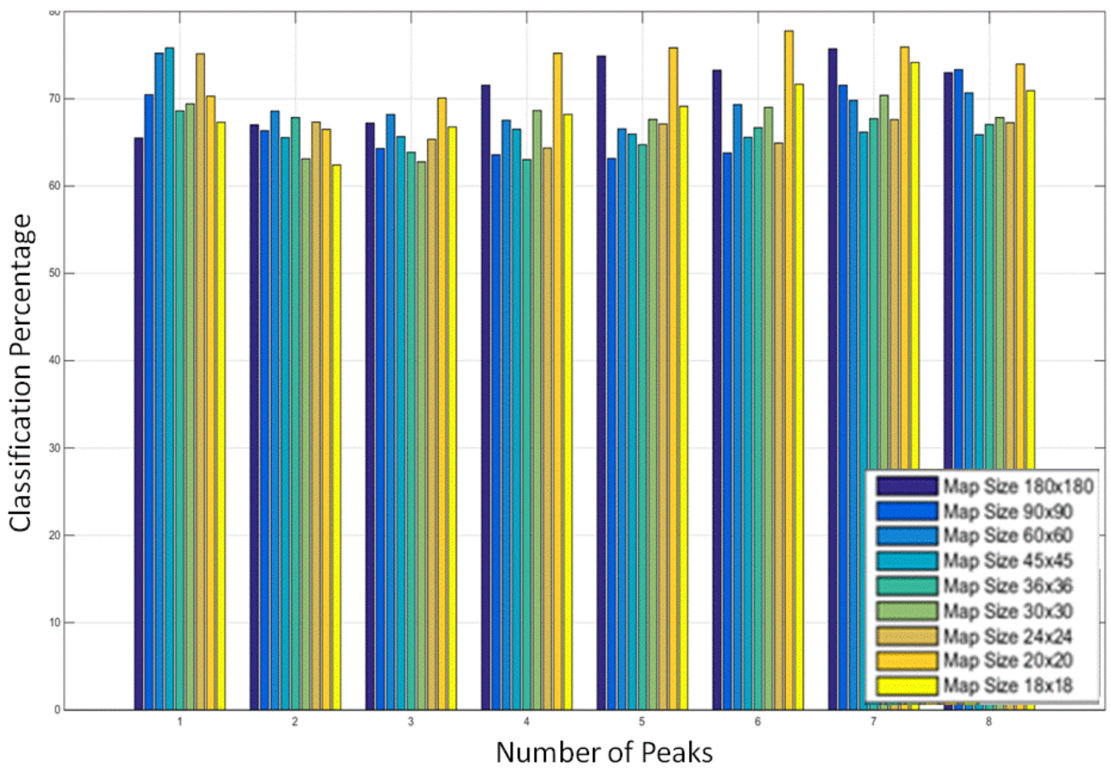


Figure 27. An average performance plot of the proposed classification scheme with 15 hidden neurons for different sizes of Hough-features map. Each map size is color coded as shown in the legend

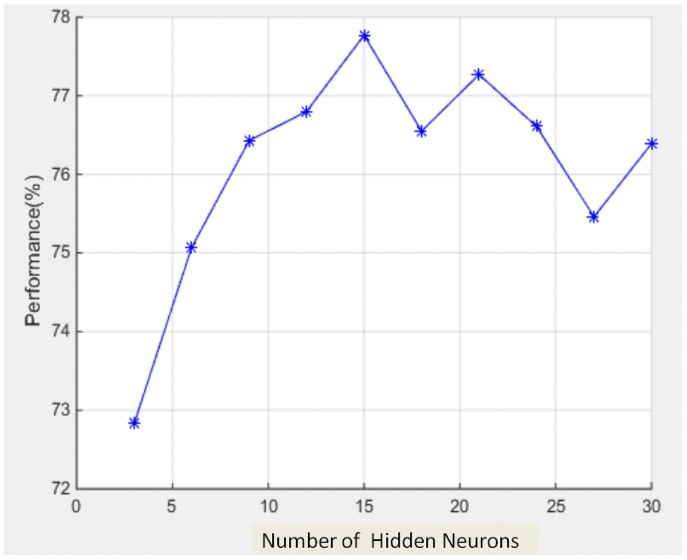


Figure 28. An average performance plot of the proposed scheme with different numbers of hidden neurons

To determine the effective sizes of denoising filters for the maritime domain images used in this work, five sets of denoising filters were tested. In this experiment, the number of neurons was set to 15. The number of peaks was six, and the map size was 20×20 . The filter sets were as follows:

Set 1: 40 × 40, 60 × 60, 80 × 80, 100 × 100 and 120 × 120

Set 2: 50 × 50, 70 × 70, 90 × 90, 110 × 110 and 130 × 130

Set 3: 60 × 60, 80 × 80, 100 × 100, 120 × 120 and 140 × 140

Set 4: 80 × 80, 100 × 100, 120 × 120, 140 × 140, and 160 × 160

Set 5: 100 × 100, 120 × 120, 140 × 140, 160 × 160, and 180 × 180.

The classification results for each filter set were averaged over 1000 trials. The average performance is depicted in Figure 29.

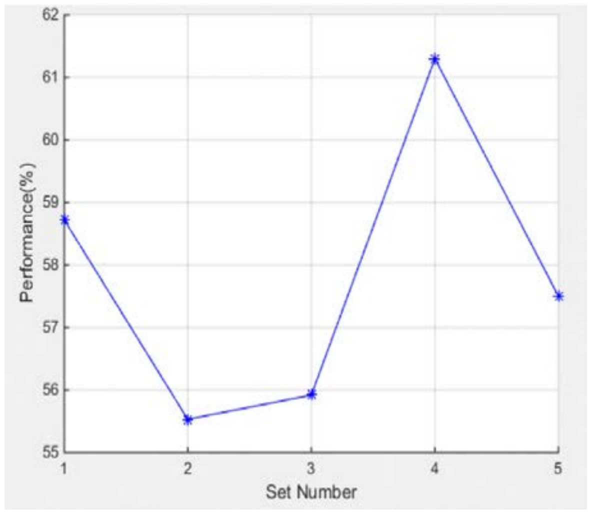


Figure 29. Scheme’s performance for different denoising filter sets

According to Figure 29, we see that the fourth set yielded the best average performance. This makes sense because the smaller the filter size, the more likely the loss of edge points, resulting in the poor quality of features. In contrast, the larger the filter size, the more background noise remains after filtering. The unwanted edge points caused by noise results in false peaks.

b. Three-Class Classification

This part is to investigate the capability of the classification scheme for classifying three objects: ships, clouds, and aircraft. The same 64 maritime images and the optimal settings of the classification scheme were used for testing. Ships, aircraft, and clouds were assigned classes 1, 2, and 3, respectively. The best confusion matrix for this classification case is shown in Figure 30.

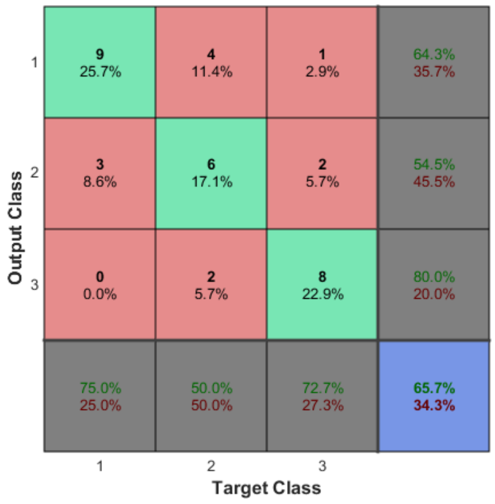


Figure 30. The best confusion matrix of the classification of ships, aircraft, and clouds

According to Figure 30, the correct classification percentages for ships, aircraft and clouds are 75 %, 50 % and 72.7 %, respectively. In the first column of the matrix, the results show that scheme misclassifies three ships as aircraft and is not confused by the clouds at all. Four aircraft are misclassified as ships as indicated in the first red cell of the second column. The high similarity of aircraft to ships and clouds results in an overall performance of 65.7 %.

The best performance of all maritime-domain image cases is summarized as in Table 4. Note that the performance of the classification scheme degrades as the number of objects to be classified increases.

Table 4. Classification performances of all maritime domain image cases

Cases	Performance (%)
Ships versus Clouds	87.0
Ships versus Aircraft	79.2
Ships versus Aircraft & Clouds	82.9
Ships, Aircraft, Clouds	65.7

In this chapter, implementation and partial testing of all modules of the proposed classification scheme were elaborated. The scheme was tested with two sets of images: perfect shapes and aerial maritime images. Some parameters, such as the number of neurons, number of peaks, and Hough features map size, were varied to determine effective setting for this classification scheme. The most effective setting found was the following: 15 neurons, six peaks, and a map size of 20×20; however, the scheme yielded poor classification results for higher-order classifications. In the next chapter, some feasible approaches to improve the scheme's performance are recommended in the future work section.

V. CONCLUSIONS

A Generalized Hough Transform-based classification scheme for maritime domain objects-of-interest was developed to improve maritime domain situational awareness. The classification scheme performs three important tasks: edge detection of the object contained in the imagery, feature extraction by using the GHT algorithm, and classification by using a neural network.

Edge detection of an object-of-interest in noisy aerial imagery was improved by using a denoising filter prior to the GHT based feature extraction. The GHT algorithm was used to form an R-table representing the object shape. To account for the scale and orientation variance of the shape, the conventional R-table was reformatted to a contour map, or a Hough-features map, where coordinates of dominant peaks, or Hough features, were extracted. These features are likely to be scale-invariant and exhibit an inconsistent pattern for an object with different orientations. This inconsistency could be overcome by the use of a neural network classifier.

A feed-forward, back-propagation neural network is an effective tool for building prediction models based on the Hough features for classifying two perfect shapes or two different object classes contained in maritime domain images; however, the classification performance of the neural network decreases as the number of classes of the object to be classified increases.

A. SIGNIFICANT CONTRIBUTIONS

The most significant contribution of this thesis is the proposed GHT-based feature selection method used to reformat the R-table into the Hough features map. A small set of features produced by this method are found to be effective for differentiating between two object classes. In maritime domain awareness, this method can be applied to improve the processing time and memory storage requirements of automatic classification systems onboard surveillance and reconnaissance platforms and at control stations.

The second contribution is the application of the DCT for denoising images to improve edge detection. Empirical filter sizes were determined based on the energy distribution of the DCT coefficients.

The third contribution is application of a neural network for classification. The network performed best when the number of hidden neurons was 15 and the percentage of training, validating and testing images was 90, five and five, respectively. This setting can be used as a baseline for classification.

B. RECOMMENDATIONS FOR FUTURE WORK

There are several possibilities for future work to improve the classification performance of the proposed scheme. In this thesis, due to variation in image sizes and background noise, an empirical DCT filter configuration as outlined in Table 2 was used for denoising images prior to edge detection. The DCT filtering technique was found to reduce background noise. A future effort can automate the DCT filtering process so that background noise can be minimized regardless of image size or background noise.

In this thesis, the Sobel operator was used for edge detection of an object of interest. Other edge detection approaches might be considered to replace the Sobel operator for better edge detection. This could increase feature accuracy.

The proposed GHT-based feature extraction does not account for variation in scale and rotation angle of an object-of-interest. In general, the GHT works better when scaling factor and rotation angle are accounted for. In future work, robust scaling factor and rotation angle estimators can be developed to de-scale and de-rotate each input image with respect to a reference model. This calibration could enhance the quality of the Hough features (dominant peaks) in the Hough-features map.

The feed-forward, back-propagation neural network with a single hidden layer was effective for most binary classification cases; however, its ability to build a prediction model decreases as the number of objects to be classified increases. This problem could be improved by using multiple hidden layers [31].

APPENDIX A. MATLAB SCRIPT FOR HOUGH FEATURES MAP GENERATION

The MATLAB script `Hough_Features.m` is used for generating a Hough-features map as described in Chapter IV. Comments are in green text. The code in Section A is developed by the author. The code in Section B is obtained from [30] and modified by the author.

A. `Hough_Features.m`

```
% This Hough_Feautes function takes two arguments: a color image (ImRGB) and a user
% input (n) to create a Hough features map so that Hough features can be
% further extracted by using SearchPeaks.m.
% The ModelHough code is borrowed from reference [30] and modified
% by the author for gerneration of the R-table.

function Y = Hough_Features(ImRGB,n)

% find R-Table (H)
  H = ModelHough(ImRGB);

 [r c] = size(H(:,:,2));

  % make all the distance not equal to zero = 1
  A = H(:,:,2);
  ind = find(A~=0);
  A(ind)=1;
  H(:,:,2)=A;

  % Create Hough features map (HF)
  HF = zeros(361,361);

  for i= 1:r
     for j = 1:c
        if H(i,j,2) == 1
          new_cols = H(i,j,1)+1; % base angle + 1
          if HF(i,new_cols) == 0
             HF(i,new_cols) = 1;
          else
             HF(i,new_cols) = HF(i,new_cols)+1;
          end
        end
```

```
      end
  end

 % Average values in block size of n x n to reduce map size.
 if n > 1
     r = 1;
     for q = 1:n:361-n
         M1(r,:) = sum(HF(q:q+n-1,:));
         r=r+1;
     end
     M2 = M1';
     k=0;
     for p = 1:n:361-n
         k=k+1;
         M3(k,:) = sum(M2(p:p+n-1,:));
     end
    Y = M3';
 else
    Y = HF;
 end
```

B. `ModelHough.m`

```
% Source: J. Pinquier. Practical: Generalized Hough Transform from
% http://www.irit.fr/~Julien.Pinquier/Docs/Hough_transform.html[30]
% Modified by: Pornrerk Rerkngamsanga, 10 DEC 2015
% This function takes two inputs: Im_BW and Gradient from SangaEdge.m

function [H step]=ModelHough(ImRGB)


% find edge points of the model
[Im_BW Gradi] = SangaEdge2(ImRGB);


% extract edge points
C = contourSanga(Im_BW,Gradi); %generates edge points (x,y), M-by-3 matrix


% Model initialization:
    % row = Gradi+90
    % column = number of the couple (alpha, distance)
    % 3rd dimension: 1 = alpha, 2 = distance
N = length(C(:,1));
H=  zeros(361,N,2); % initialization of Hough Table
Alpha = SangaAlpha(C,Im_BW);
```

```
Beta = SangaBeta(C);
[xc,yc]=barycenterSanga(Im_BW); % find the object centroid

% for each edge point
for index=1:N

    k=1;

    while H(Beta(index),k,2)~=0
      k=k+1;
    end

    % Fill out alpha values to the H-Table
    H(Beta(index),k,1)= Alpha(index);

    % compute ans fill out the distance values
    H(Beta(index),k,2)= distance(xc,yc,C(index,1),C(index,2));

end
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. MATLAB SCRIPT FOR HOUGH FEATURES EXTRACTION

The MATLAB script `SearchPeaks.m` is used for Hough-feature extraction as described in Chapter IV. Comments are in green text. This code is developed by the author by leveraging `findpeaks.m` function provided in MATLAB for peak localization.

```
% this code is an example of SearchPeak.m function that takes an input (Hough features map)
% and extracts co-ordinates of the three tallest peaks. More  peaks can be extracted by increasing % a number of if-statements.

function [Final_Pks_Select] = SearchPeaks(HF)
[s1 s2] = size(HF);
%find co-ordinates (x,y) of peaks in each row
A = reshape(HF',1,numel(HF));
[Val_Peaks_H Loc_H] = findpeaks(A);
Peaks_Loc_H = zeros(length(Loc_H),2);

for i=1:length(Loc_H)
   r = floor(Loc_H(i)/s1)+1;
   c = Loc_H(i)-s1*floor(Loc_H(i)/s1);
   if c == 0
      Peaks_Loc_H(i,:) = [r-1 s1];
   else
      Peaks_Loc_H(i,:) = [r c];
   end
end
Peaks_Loc_H = [Peaks_Loc_H(:,2) Peaks_Loc_H(:,1)];

%find co-ordinates (x,y) of peaks in each column
B = reshape(HF,1,numel(HF));
[Val_Peaks_V Loc_V] = findpeaks(B);
Peaks_Loc_V = zeros(length(Loc_V),2);
for i=1:length(Loc_V)
   r = floor(Loc_V(i)/s1)+1;
   c = Loc_V(i)-s1*floor(Loc_V(i)/s1);
   if c == 0
      Peaks_Loc_V(i,:) = [r-1 s1];
   else
      Peaks_Loc_V(i,:) = [r c];
```

```
    end
end

% extract only co-ordinates (x,y) and their value that occur in both search above
Common_Pks = [];
Val = [];
counter = 1;

for i = 1:length(Peaks_Loc_H(:,1))
    for j = 1:length(Peaks_Loc_V(:,1))
        if Peaks_Loc_H(i,:) == Peaks_Loc_V(j,:)
           Common_Pks(counter,:) = Peaks_Loc_V(j,:);
           Val(counter,:) = Val_Peaks_V(j);
           counter = counter+1;
        end
    end
end

% Select only the first three tallest peaks
Pks_Select = [];
First = find(Val == max(Val)); % first peak
First_Pks_Select = Common_Pks(First,:);

Dummy = Val;
Dummy(First)=0;
Second = find(Val == max(Dummy)); % second peak

if length(First)==0
      Final_Pks_Select = [0 0 0 0 0 0];
elseif length(First)== 1
   if length(Second)==0
      Final_Pks_Select = [First_Pks_Select 0 0 0 0];
   elseif length(Second)==1
      Second_Pks_Select = Common_Pks(Second,:);
      % find third peak
      Dummy(Second)=0;
      Third = find(Val == max(Dummy)); % thrid peak
      Third_Pks_Select = Common_Pks(Third,:);
      if length(Third)==0
         Pks_Select = [First_Pks_Select;Second_Pks_Select;Third_Pks_Select];
         Final_Pks_Select = [Pks_Select(1,:) Pks_Select(2,:) 0 0];
      else
         Pks_Select = [First_Pks_Select;Second_Pks_Select;Third_Pks_Select];
         Final_Pks_Select = [Pks_Select(1,:) Pks_Select(2,:)
Pks_Select(3,:)];
    end
  else
    Second_Pks_Select = Common_Pks(Second,:);
    Pks_Select = [First_Pks_Select;Second_Pks_Select];
    Final_Pks_Select = [Pks_Select(1,:) Pks_Select(2,:) Pks_Select(3,:)];
  end
elseif length(First)== 2
  if length(Second)==0
     Final_Pks_Select = [First_Pks_Select(1,:) First_Pks_Select(2,:) 0 0];
  else
     Second_Pks_Select = Common_Pks(Second,:);
     Pks_Select = [First_Pks_Select;Second_Pks_Select];
     Final_Pks_Select = [Pks_Select(1,:) Pks_Select(2,:) Pks_Select(3,:)];
  end
else
   Final_Pks_Select = [First_Pks_Select(1,:) First_Pks_Select(2,:) First_Pks_Select(3,:)];
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. MATLAB SCRIPT FOR NEURAL NETWORK

The MATLAB script `NN_Trial.m` is used for all classification cases presented in Chapter IV. Comments are in green text. This code is modified from a neural network script provided in *neural pattern recognition* application.

```
% Solve a Pattern Recognition Problem with a Neural Network
% Script generated by Neural Pattern Recognition app
% Created Sat Oct 17 14:51:39 PDT 2015 and modified to train and test data
% N times and to calculate the average percentage or performance of N runs.
%
% This script assumes these variables are defined:
%
%   train_x18_6Pks - train data.
%   train_label - target data.
clear
clc

%Start Training Process
load('train_x18_6Pks');
load('train_label');
x = train_x18_6Pks';
t = train_label';

N = 1000;
holder = [];
for i = 1:N
% Create a Pattern Recognition Network
hiddenLayerSize = 15;
net = patternnet(hiddenLayerSize);

% Setup Division of Data for train2ing, Validation, test2ing
net.divideParam.trainRatio = 90/100;
net.divideParam.valRatio = 5/100;
net.divideParam.testRatio = 5/100;

% train the Network
[net,tr] = train(net,x,t);

% test the Network
y = net(x);
```

```
e = gsubtract(t,y);
tind = vec2ind(t);
yind = vec2ind(y);
percentErrors = sum(tind ~= yind)/numel(tind);
performance = perform(net,t,y);
% Plots
% Uncomment these lines to enable various plots.
% figure, plotperform(tr);
% figure, plottrain2state(tr);
% figure, plotconfusion(t,y)
% figure, plotroc(t,y);
% figure, ploterrhist(e)

% Start Testing Process
% test_x18_6Pks - test data.
% test_label - target data.
load('test_x18_6Pks');
load('test_label');
x2 = test_x18_6Pks';
t2 = test_label';

y_test = net(x2);
tind2 = vec2ind(t2);
yind2 = vec2ind(y_test);
percentErrors2 = sum(tind2 ~= yind2)/numel(tind2)

% Plot confusion matrix when classification percentage is greater or equal to 80%
if percentErrors2 < 0.2
figure, plotconfusion(t2,y_test)
end
holder(i) = percentErrors2; % keep all test results produced by each trained net
end
%
Final_Percent = 100-(sum(holder)*100/N) % calculate the performance or average
percentage of a thousand runs
```

LIST OF REFERENCES

[1] L. Kaur and V. K. Sharma, “Object detection from the satellite images using divide and conquer model,” in *Proc. SSRG Int. J. Comput. Sci. and Eng.*, vol. 1, no. 10, pp. 13–18, 2014.

[2] E. Schwarz, D. Krause, M. Berg, H. Daedelow, and H. Maass, “Near Real Time Applications for Maritime Situational Awareness,” *Int. Arch. Photogramm. Remote Sensing and Spatial Inform. Sciences*, vol. XL-7/W3, pp. 999–1003, 2015.

[3] C. Brekke and A. H. Solberg, “Oil spill detection by satellite remote sensing,” *Int. J. Remote Sensing of Environment*, pp. 1–13, 2005.

[4] Maritime Security Policy, National Security Presidential Directive NSPD-41, George W. Bush Administration, Washington, DC, 2004, pp. 1–9.

[5] M. R. Noroozi, A. Ramezani, and M. Aghababaee, “Automatic ship types classification in silhouette images,” *Int. J. Eng. and Advanced Technol. IJEAT*, vol. 4, no. 1, pp. 52–56, Oct. 2014.

[6] J. Barrho, M. Adam, and U. Kiencke, “Finger localization and classification in images based on Generalized Hough Transform and probabilistic models,” in *Proc. Ninth Int. Conf. Control, Automation, Robotics and Vision*, 2006.

[7] D. H. Ballard and C. M. Brown, “The Hough method for curve detection,” in *Comput. Vision*, NJ: Prentice-Hall, 1982, pp. 128–131.

[8] “Generalized Hough Transform (GHT),” class notes for Digital Signal Processing, Dept. of Comput. Sci. & Eng., Indian Inst. of Technol., Delhi.

[9] P.-F. Fung, W.-S. Lee and I. King, “Randomized Generalized Hough Transform for 2-D grayscale object detection,” in *Proc. IEEE* 13*th Int. Conf. Pattern Recognition*, vol. 2, pp. 511–515, Aug. 1996.

[10] Z. Qian, W. Wang, and T. Qiao, “An edge detection method in DCT domain,” in *Procedia Eng.*, pp. 344–348, 2012.

[11] H. Joo, J. Park, J. Kim, and B.-U. Lee, “Image noise reduction in Discrete Cosine Transform domain,” *IEEK Trans. Smart Processing and Computing*, vol. 2, no. 1, pp. 20–26, Feb. 2013.

[12] S. A. Khayam. (2003, Oct. 10). The Discrete Cosine Transform. [Online]. Available: http://www.lokminglui.com/DCT_TR802.pdf

[13] M. H. Beale, M. T. Hagan, and H. D. Demuth. (n.d.). Neural network toolbox getting started guide. [Online]. Available: http://www.mathworks.com/help/pdf_doc/nnet/nnet_gs.pdf [Dec 8, 2015].

[14] R. Maini, “Study and comparison of various image edge detection techniques,” *Int. J. Image Process. IJIP*, vol. 3, no. 1, pp. 1–12.

[15] K. Kumar, “Comparative study on various edge detection techniques for 2-D image,” *Int. J. Comput. Appl. IJCA*, vol. 119, no. 22, pp. 6–10, 2015.

[16] S. Baluja, “Making templates rotationally invariant: an application to rotated digit recognition,” unpublished.

[17] H. Wang and D. Suter, “Robust adaptive-scale parametric model estimation for computer vision,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 11, pp. 1459–1474, Nov. 2004.

[18] J. Cai and H. Wang, “Adaptive scale based entropy-like estimator for robust fitting,” *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Process.*, pp. 5509–5513, 2013.

[19] W.-Y. Kim and Y.-S. Kim, “Robust rotation angle estimator,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, no. 8, pp. 768–773, Aug. 1999.

[20] J. Creighton, “The Discrete Cosine Transform and its application to image compression,” unpublished.

[21] R. A. A. Nobrega, J. A. Quintanilha, and C. G. O’Hara, “A noise-removal approach for LIDAR intensity images using anisotropic diffusion filtering to preserve object shape characteristics,” in *ASPRS 2007 Conf.*, 2007.

[22] P. M. Atkinson and A. R. L. Tatnall, “Introduction Neural networks in remote sensing,” *Int. J. Remote Sensing*, pp. 699–709, 2015.

[23] M.-J. Li and R.-W. Dai, “A personal handwritten Chinese character recognition algorithm based on the generalized Hough transform,” in *Proc. Third Int. Conf. Document Anal. and Recognition*, pp. 828–831, 1995.

[24] P. K. Gupta and R. Kanhirodan, “A DCT based filtering of biomedical images,” in *IEEE Int. Conf. Ind. Technol.*, pp. 2551–2556, 2006.

[25] I. C. Garcia and U. Zolzer, “Hough Transformed based ship segmentation using centerline extraction and feature angles,” in *Proc. Eighth Int. Conf. Signal Image Technol. and Internet Based Systems*, pp. 149–154, 2012.

[26] K. Nanaa, M. Rizon, M. N. A. Rahman, Y. Ibrahim and A. Z. A. Aziz, “Detecting mango fruits by using Randomized Hough Transform and backpropagation neural network,” in 18*th Int. Conf. Inform. Visualisation*, pp. 388–391, 2014.

[27] G. P. Zhang, “Neural networks for classification: a survey,” *IEEE Trans. Syst., Man, and Cybern.*, vol. 30, no. 4, pp. 451–462, Nov. 2000.

[28] R. Nekovei and Y. Sun, “Back-propagation network and its configuration for blood vessel detection in angiograms,” *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 64–72, 1995.

[29] R. Öktem, K. Egiazarian, V. V. Lukin, N. N. Ponomarenko, and O. V. Tsymbal4, “Locally adaptive DCT filtering for signal-dependent noise removal,” *EURASIP J. Adv. Signal Process*, pp. 1–10, 2007.

[30] J. Pinquier. Practical: Generalized Hough Transform [Online]. Available: http://www.irit.fr/~Julien.Pinquier/Docs/Hough_transform.html

[31] S. Karsoliya, “Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture,” *Int. J. Eng. and Technol.*, vol. 3, no. 6, pp. 714–717, 2012.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California