Feature-based Vehicle Classification in Wide-angle Synthetic Aperture Radar

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By
Kerry Edward Dungan, M.S.
Graduate Program in Electrical and Computer Engineering

The Ohio State University
2010

Dissertation Committee:
Lee C. Potter, Adviser
Mikhail Belkin
Randolph L. Moses
© Copyright by
Kerry Edward Dungan
2010
ABSTRACT

Recent developments in synthetic aperture radar (SAR) have enabled persistent surveillance of a large scene using down-looking airborne radar in a circular path. An important application of this new technology is identifying vehicles. Numerous prior studies have focused on narrow aperture SAR to identify typically large military vehicles. This study demonstrates that the wide apertures available to circular synthetic aperture radar provide enough information to identify the relatively small signatures of civilian vehicles.

Some of the challenges associated with identifying civilian vehicles include small radar cross sections, similar dimensions, occlusions, unknown pose, and scalable computation. To demonstrate solutions to these challenges, a rapid attributed scattering center extraction method, visualization tools, two experimental databases, and two partial set based classifiers are developed. The new classifiers were adapted and extended from existing pattern recognition algorithms to address the SAR vehicle identification problem; the extensions may be useful to the pattern recognition community in general. In particular, a Bayesian interpretation of the pyramid match kernel is provided.

To demonstrate the algorithm, X-band scattering is simulated from ten civilian vehicles that are placed throughout a large scene, varying elevation angles in the 35 to 59 degree range. The algorithms achieved 98% or better classification performance.
Similar performance is demonstrated for a seven class task using airborne radar measurements. Given preformed imagery, scattering centers are extracted, coded, and classified in real-time.
This dissertation is dedicated to my wife Kathleen and my two beautiful daughters
Audrey and Claire.
ACKNOWLEDGMENTS

This work was supported in part by an allocation of computing time from the Ohio Supercomputer Center. This material is based upon work supported by the Air Force Research Laboratory under Award FA8650-07-D-1220 and by the Air Force Office of Scientific Research under Award FA9550-06-1-0324. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the U.S. Air Force. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.
VITA

1972 ........................................Born - Dayton, OH

1994 ........................................B.S. Electrical Engineering, The Ohio State University, Columbus, OH

1995-1996 ..................................Electrical Engineer, Harris Corporation, Palm Bay, FL

1997 ........................................M.S. Electrical and Computer Engineering, The Ohio State University, Columbus, OH

1997-2001 .................................Member of Technical Staff, Lucent Technologies, Columbus, OH

2001-2002 .................................Technical Account Manager, Synplicity, Inc., Durham, NC

2005 ........................................M.Ed. Mathematics Education, North Carolina State University, Raleigh, NC

2006-present ...............................Ph.D. Student, The Ohio State University, Columbus, OH

PUBLICATIONS


**FIELDS OF STUDY**

Major Field: Electrical and Computer Engineering
# TABLE OF CONTENTS

Abstract .......................... ii
Dedication ........................ iv
Acknowledgments .................. v
Vita ............................... vi
List of Figures .................... xi
List of Tables ..................... xvii

Chapters:

1. Introduction ........................ 1
   1.1 Point Pattern Matching .............. 3
       1.1.1 Correspondence Based Point Matching .... 4
       1.1.2 Non-correspondence Based Point Matching ... 5
       1.1.3 Methods that are Invariant to Transformations ... 7
   1.2 Evaluation of Algorithms ............. 8
   1.3 Summary of Contributions ............ 12
   1.4 Organization of Chapters ............ 13

2. Wide-angle Attributed Point Patterns ......... 15
   2.1 Building sets of attributed scattering centers .... 16
   2.2 Civilian Vehicle Circular SAR Databases .......... 20
       2.2.1 Experiment Design using CVDomes .......... 20
       2.2.2 Experiment Design using 2006 Collection ....... 27
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Attributed Scattering Centers</td>
<td>33</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Coordinate system conversions</td>
<td>34</td>
</tr>
<tr>
<td>3.3</td>
<td>Extracting a 3D Point Cloud from Single Pass Imagery</td>
<td>36</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Vehicle model for layover</td>
<td>37</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Extracting a Rectangle from a Set of Points</td>
<td>38</td>
</tr>
<tr>
<td>3.3.3</td>
<td>3D location of elevated points</td>
<td>39</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Empirical insets</td>
<td>42</td>
</tr>
<tr>
<td>3.4</td>
<td>Experimental Results</td>
<td>44</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Measured data at scene center</td>
<td>45</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Invariance to vehicle location</td>
<td>45</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Pickup truck example</td>
<td>48</td>
</tr>
<tr>
<td>3.5</td>
<td>Height Estimation Uncertainty</td>
<td>49</td>
</tr>
<tr>
<td>3.6</td>
<td>Discussion</td>
<td>50</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>52</td>
</tr>
<tr>
<td>4.2</td>
<td>Set Classification and Analysis</td>
<td>55</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Sets Distances using the Hausdorff Metric</td>
<td>56</td>
</tr>
<tr>
<td>4.2.2</td>
<td>MDS for Analysis and/or Training</td>
<td>59</td>
</tr>
<tr>
<td>4.2.3</td>
<td>LMDS for Visualization and/or Classification</td>
<td>61</td>
</tr>
<tr>
<td>4.3</td>
<td>Latent Fingerprint Application</td>
<td>62</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Classifying Latent Fingerprints</td>
<td>63</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Feature Vector and Covariance</td>
<td>64</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Latent Fingerprint Classification Results</td>
<td>65</td>
</tr>
<tr>
<td>4.3.4</td>
<td>Time complexity</td>
<td>66</td>
</tr>
<tr>
<td>4.3.5</td>
<td>Comparison to Prior Work</td>
<td>67</td>
</tr>
<tr>
<td>4.4</td>
<td>Vehicles in Circular Synthetic Aperture Radar</td>
<td>68</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Set Distance Parameters</td>
<td>69</td>
</tr>
<tr>
<td>4.4.2</td>
<td>CVDomes Classification Results</td>
<td>70</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Time Complexity</td>
<td>76</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Comparison to MSE Classifier</td>
<td>78</td>
</tr>
<tr>
<td>4.4.5</td>
<td>Single-pass 3D Example</td>
<td>80</td>
</tr>
<tr>
<td>4.4.6</td>
<td>Results for 2006 Collection</td>
<td>81</td>
</tr>
<tr>
<td>4.5</td>
<td>Discussion</td>
<td>83</td>
</tr>
</tbody>
</table>
5. Classifying Vehicles with Pyramid Match Hashing .................................................. 85

5.1 Introduction .................................................................................. 85
  5.1.1 Overview of Algorithm ......................................................... 86

5.2 Invariantly Mapped Database ....................................................... 88

5.3 The Pyramid Match ...................................................................... 91
  5.3.1 The Pyramid Histogram ....................................................... 92
  5.3.2 The Pyramid Match Kernel .................................................. 92
  5.3.3 Pyramid Match Hashing ........................................................ 93
  5.3.4 Approximating Bayes Optimal Weights in PMK ..................... 94

5.4 Classification Process ................................................................. 100
  5.4.1 Hash Code Length ............................................................. 102
  5.4.2 Fusion of Decisions .............................................................. 103

5.5 Experimental Results .................................................................. 105
  5.5.1 Classification Results using CVDomes ................................. 107
  5.5.2 Classification Results using the Circular SAR 2006 Collection 114

5.6 Discussion .................................................................................... 115

6. Conclusion ..................................................................................... 117

Appendices:

A. Appendix ..................................................................................... 119

  A.1 Cartesian to spherical coordinates ............................................. 119
  A.2 Variable aperture imaging ....................................................... 120
  A.3 Derivation of $\phi'$ for Chapter 3.4.2 ................................. 121

Bibliography .................................................................................... 122
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The image in (a) was generated from the Gotcha Public Release Dataset; the image has a 5-degree aperture centered broadside to the automobile. Detected peaks in (b) display an abstraction of the image with scattering centers plotted onto a pixel array.</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>The plot in (a) shows the extracted 2D scattering location for individual azimuths stacked into a cube. For visualization, the color of the dots corresponds to the center azimuth angle of the subaperture image in which a peak is detected. A vehicle has a spiral pattern in the cube caused by the leading edge of the vehicle at any given look angle. When flattened to a top view in (b), the point cloud reveals the vehicle’s 360-degree shape. The SAR even-bounce (d) and odd-bounce (e) images are composites of 5-degree aperture images for the full 360-degree extent around a vehicle. Blue even and green odd points are shown in (e).</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>For simulation, a vehicle $V$ is translated in the $xy$ plane, and the radar $R$ orbits the scene center $O$ with a constant elevation $\phi_0 = 45^\circ$.</td>
<td>22</td>
</tr>
<tr>
<td>2.4</td>
<td>The left two columns show 360 degrees of SAR imagery for three sample vehicles. For the first column the vehicles are centered in the scene as shown in (m), while the vehicles are offset in the second column as shown in (n). The third and forth columns show extracted scattering center locations that are attributed with an azimuth attribute. The arrows point in the azimuth direction.</td>
<td>24</td>
</tr>
</tbody>
</table>
2.5 If $v_5$ is offset from the center of a large circular SAR scene it will experience different elevations to the radar over the flight path. As a result, the same vehicle looks different when translated around the scene. We generated simulated vehicle imagery for each of the dots in the graph; imagery for six of the locations are displayed. The circled dots are locations used in the classification training set.

2.6 By extracting attributed scattering centers from the imagery shown in Figure 2.5, vehicles are abstracted to a set of attributed points. Among the attributes for a scattering center are 2D location and azimuth to the radar. Darker points in the plots start with a zero degree azimuth in the $x$ direction; lighter points represent increasing azimuths as the radar orbits counterclockwise from the $x$-axis. The plot to the right shows a spiral pattern by representing the azimuth on a $z$-axis.

2.7 There are seven identifiable civilian passenger vehicles in 2006 data collection parking lot. The top left image shows the seven vehicles in one scene, while the remaining images show close-up views of each vehicle independently. We extracted attributed scattering centers of these seven vehicles for all eight orbits of the radar.

3.1 The same vehicle imaged at the scene center (a) and offset by 0.4 times the radius of the flight path (b) yields markedly different signatures.

3.2 Ray tracing (a) on a facet model (Ford Taurus wagon on a dielectric half-space) to illustrate even-bounce and odd-bounce reflections. (b) Geometry of layover.

3.3 The local imaging surface for a vehicle location is offset from the scene center and has a different normal vector. Layover occurs in the direction of the radar $R$.

3.4 Two coordinate systems $A$ and $B$ are related by an affine transformation. $O_A$ can represent the scene center, while $O_B$ can anchor the local image.

3.5 Example points (a) are mapped onto a rectangle. Either even-bounce or HH information is sufficient to find the base of a Hyundai Santa Fe (a midsize SUV) in (b) and (c). In the HH only case, the algorithm has difficulty finding the end of a Nissan Sentra (a small sedan) in (e), while it works well with even bounce in (d).
3.6 This construction is a top view of the $x'y'$ plane. The rectangle extraction algorithm finds the polygon $FGHK$ around the base of the vehicle. Additionally, the vehicle’s elevated features are inset from the base and are modeled by $F_EG_EH_EK_E$. ........................................... 41

3.7 For seven vehicles (a)-(d) and (m)-(o), a single pass in the 2006 Collection yields point clouds with the 2D scattering center locations shown in (e)-(h) and (p)-(r). Points outside of the base are presumed to be due to layover from elevated reflectors. Parts (i)-(l) and (s)-(u) provide plots of a 3D extraction of the presumed elevated reflectors, where the elevated features have darker voxels. ........................................... 46

3.8 The images in (a)-(d) are top views of SAR point clouds of a Toyota Camry that is offset from scene center by the denoted $(q, \theta_v)$ pair. A 3D extraction algorithm folds suspected elevated points over a rectangle extracted from vehicle base points to generate a 3D image of the vehicle in (e)-(h). Unfolding the 3D point cloud at patch center results in the 2D images in (i)-(l). In the 2D images, odd-bounce reflectors are in the darker shade. In the 3D images, elevated reflectors are in the darker shade ........................................... 47

3.9 Toyota Tacoma pickup truck: (a) 2D image peaks; (b) 3D point cloud after inversion of conjectured layover points. ................................. 48

4.1 Calculate the set distance between the query $Q$ and each class sample $H_i$; the nearest neighbor classification is the shortest distance. An optional chain of processing for analysis/visualization or alternative classifiers using MDS/LMDS is indicated with the dashed boxes. Offline processing is shown in green. ........................................... 54

4.2 The image in (a) is a latent fingerprint with overlaid minutiae, and (b) is a tenprint image with overlaid minutia. The plot in (c) shows the registration of the latent and tenprint location information, while (d) adds the orientation field information. ................................. 63

4.3 Higher quality latent imagery results in higher probability of identification. The black line shows that good latent images are identified in the top five or so tenprint images around 90% of the time. Bad images are shown in red with ugly in blue. ................................. 66
4.4 Typical attributed point patterns in the latent fingerprint experiment contained 20 points in the latent image and 100 points in the tenprint. Calculating the minimized LTS-HD takes approximately 5s.

4.5 MDS enables visualization of the point patterns as vectors in $\mathbb{R}^2$. The plot in (a) shows v8 samples properly classified, while (b) shows v9 samples properly classified.

4.6 Eigenvalues for the Schoenberg augmented Euclidean distance matrices are displayed. (a) and (b) show representative examples, while (c) and (d) show the average trends for the dataset. There are several large positive eigenvalues and the few negative ones are small.

4.7 The probability of correct classification for several classifiers is displayed versus embedding dimension. Generally, performance peaks around $\mathbb{R}^4$ with similar performance for all four classifiers.

4.8 Typical attributed point patterns in the SAR experiment contained on the order of 200 points per set, and calculating the minimized LTS-HD, using a particle swarm optimization, required approximately 15 seconds.

4.9 After extracting 3D information for attributed scattering centers (a) displays all of the data for a Camry. Reducing the data set to bright peaks gives (b). Similarly, the reduced Maxima is shown in (c). The red rectangles represent estimated base outlines.

4.10 The 2D training samples of two dissimilar vehicles like a Camry (v1) and Tacoma (v10) are well separated by an RBF SVM and a linear SVM as shown in (a) and (b), respectively. However, similar vehicles like a Camry and a Maxima (v5) are more difficult to separate in (c) and (d). After extracting 3D information, the Camry and Maxima are more easily separated as as displayed in (e) and (f).

4.11 Classification performance, using a 45-degree aperture, is dependent upon viewing angle. Dips in the plot may coincide with corners of the vehicle or clutter from curbs. Some viewing angles such as the 250-300 degree range give higher performance.
5.1 The classification algorithm as two input branches, where the training data is processed offline and the query is processed when data is available. Query hash codes are compared against the database, and the nearest several candidates from the database are evaluated using the pyramid match kernel for classification. .......................... 86

5.2 An example attributed point pattern with an orientation (azimuth) attribute is displayed on the left. A point \( p_i \) is represented by a set \( H_i \) containing the point along with three nearest neighbors in a local frame of reference. ................................. 89

5.3 If an intersection is detected in the first level of a pyramid histogram, the two feature vectors exist within the smallest resolution bin, as shown on the left. In higher levels of the pyramid histogram, the hypothesis feature vector exists in set \( A \), while the query vector is in \( B \). 98

5.4 Hypothesis points are uniformly distributed in a resolution bin, while the query samples are normally distributed with respect to the hypothesis in the next level of the pyramid histogram. The expected distance at any level is determined by the average distance between corresponding pairs. ................................. 99

5.5 Each point of each training sample is mapped to a set of nearest neighbors in a local frame of reference. These sets are coded into \( M \)-bit hash keys. The points of a query sample are also coded to \( M \)-bit hash keys. The query codes are compared to database entries using Hamming distances. ................................. 101

5.6 A query sample (vehicle) contains 216 scattering centers; each are identified to a member of the training set. A majority vote based fusion method identifies the query. In (b), v6 is rotated and placed at the center is easily identified. In (c), (d), and (e), the vehicle is placed at the edge of the scene. Setting the Hamming NN parameter too small causes a near misclassification in (e). ................................. 108

5.7 To calculate classifier performance as parameters are varied, the 440 samples are accumulated in a confusion matrix. The trace over 440 give \( P_{id} \). This example shows a 98.6% performance for a 120-bit hash key to find 500 Hamming nearest neighbors. ................................. 109
5.8 As the hash key length increases, classification performance improves. While the sum and majority vote fusion rules provide similar results, the product and median rules lag in this application. 

5.9 More HNNs increase the chance of finding the largest PMK values. A longer hash code reduces the need for more HNNs to get the same performance.

5.10 Changing the local neighborhood size (a) and error covariance matrix affects performance. The variances are individually tuned for (b) location, (c) azimuth, (d) amplitude, and (e) polarization.

5.11 It is possible to achieve 100% classification on the 2006 data. The histogram in (a) shows an example of v5 correctly classified with a 40-bit hash key. Performance (b) peaks with a 30-bit hash key and time complexity increases with key length.

A.1 From the top view of a circular SAR collection, a vehicle $V$ is offset from the patch center $O$. Attaining a desired cross-range resolution on the vehicle requires the aperture $\delta'$, which in turn requires backprojection of the phase history over the aperture $\delta$. 

xvi


**LIST OF TABLES**

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Comparison of four classifiers</td>
<td>10</td>
</tr>
<tr>
<td>2.1</td>
<td>Vehicles identified by a number in experiments. For example, the Camry may be referred as 1 or v1.</td>
<td>21</td>
</tr>
<tr>
<td>2.2</td>
<td>Vehicle labels in 2006 Data</td>
<td>28</td>
</tr>
<tr>
<td>3.1</td>
<td>3D Processing Algorithm</td>
<td>33</td>
</tr>
<tr>
<td>3.2</td>
<td>Estimated height vs. actual height</td>
<td>43</td>
</tr>
<tr>
<td>3.3</td>
<td>Estimated versus actual widths and lengths (in meters)</td>
<td>44</td>
</tr>
<tr>
<td>4.1</td>
<td>These confusion matrices show actual vs. classified results for ten vehicles using a NN classifier; (a) has 6 training samples per class, while (b) uses only one training sample per class.</td>
<td>72</td>
</tr>
<tr>
<td>4.2</td>
<td>Confusion matrix showing actual vs. classified results for ten vehicles after applying MDS/LMDS with an RBF-SVM classifier.</td>
<td>77</td>
</tr>
<tr>
<td>4.3</td>
<td>When classifying the Maxima using 180 degrees of aspect, the MSE classifier performs poorly (a), while the minimized LTS-HD shows a relatively good performance (b).</td>
<td>79</td>
</tr>
</tbody>
</table>
CHAPTER 1: Introduction

This dissertation establishes that civilian vehicles are identifiable from wide-angle synthetic aperture radar (SAR) imagery using a compressed representation of attributed scattering centers. Some of the challenges associated with classifying civilian vehicles in SAR imagery include small signatures, similar vehicle dimensions, low resolution, unknown translation and pose, occlusions, and an explosion in the number library templates due to the anisotropy of radar signatures. Addressing these issues led to pattern recognition challenges of reduced memory usage, scalable computation, feature correspondences, partial matches, and invariance to translations and rotations. This dissertation successfully addresses each of these challenges and presents the first results for classification of civilian vehicles in SAR imagery using attributed scattering centers. In achieving the goal of vehicle classification, we developed a rapid approximate scattering center extraction method, visualization tools, two experimental databases, and two partial set based classifiers.

Synthetic aperture radar is an all-weather and all-lighting imaging system based on accumulated radar pulses that are overlapped in a particular scene. Using a persistent surveillance technique called circular SAR, a down-looking radar platform flies in a circular path around a scene. We generate images from the resulting phase history
using backprojection [1]. For visualization and classification purposes, within the circular aperture, we generate a series of overlapping subaperture images to characterize the anisotropic nature of SAR imagery.

From the series of sub-aperture SAR images, we compress the information into unordered sets of attributed scattering centers [2]. Our method implements a peak extractor to store location information as well as additional attributes such as amplitude, azimuth, and polarization. Plotting the attributed scattering center locations as they vary with azimuth captures the migration patterns of scattering features. In addition to compressing the imagery into the relatively small file sizes of attributed scattering centers, the unordered vector set representation fits into the point pattern matching thread of research [3].

After representing the radar imagery as sets of attributed scattering centers, we generated visualizations of the imagery as points in scatter plots. We developed a method to extract three-dimensional (3D) information by inverting layover for individual points. In the Chapter on partial Hausdorff distances (PHD), we developed a method to find distances between point patterns and visualize comparisons between point patterns in a Euclidean space.

To analyze and evaluate scattering center based visualization and classification of wide-angle SAR imagery, we developed two databases. The first database consists of extracted vehicles from the Gotcha public release dataset [4]. For the second database, we created a phase history for ten civilian vehicles using a physical optics electromagnetic scattering prediction code [5, 6]. The simulated dataset offered the flexibility to generate SAR imagery over a range of elevations as opposed to the fixed elevation of the airborne collection.
Finally, a large portion of the dissertation describes and demonstrates two new classification algorithms. Both algorithms are designed to classify point patterns with respect to a database of point patterns where patterns are considered equivalent over a class of transformations. In particular, we focus on the equivalence class of rigid transformations since we are interested in two-dimensional (2D) SAR images as they change over translations and rotations in a plane. We proceed by reviewing point pattern matching in more detail followed by a discussion on the evaluation and significance of the new algorithms.

1.1 Point Pattern Matching

Matching two sets of points in a 2D spatial sense is a process of transforming the sets with respect to each other to determine some information about their relation. The sets may or may not have the same cardinality, and we may be interested in either a full or a partial set match. We may have a correspondence (not necessarily one-to-one) between the two sets \textit{a priori}, and we would like to find the transformation that minimizes a distance between the two sets. However, if we do not have a correspondence, then we may want to find the best correspondence (or implied correspondence) while searching transformations simultaneously. Furthermore, we could map the points into a transformation invariant feature space and create a correspondence from the nearest neighbors or classify directly by some spectral or other method. Ultimately, for classification (as opposed to a registration problem), the correspondence may be a hidden or unnecessary step in finding a distance between sets.
Yin [3] divides point pattern matching into categories such as clustering, parameter decomposition, relaxation, bounded alignment, spectral graph analysis, genetic algorithms, and simulated annealing. Zhao et al. [7] describes graph methods such as spectral, tree, and relaxation. Goodrich et al. [8] lists cluster, absolute orientation, extracted information, and computational geometry as potential methods. Many other publications attack the point matching problem with seemingly independent methods. For instance, the iterated closest point method [9] spurred further works not discussed by the above researchers. In addition, techniques are often combined or embedded within another and it is difficult to categorize all of the methods in some general framework.

In any case, recent literature on point pattern matching typically focuses on the application of image registration. Some of these algorithms are well adapted to comparing attributed scattering centers as well. The algorithms fall loosely into a few major categories including correspondence, non-correspondence, and feature based methods. We briefly summarize examples of these three categories for point pattern matching in the following subsections.

1.1.1 Correspondence Based Point Matching

The correspondence methods match two point patterns, in which each point in a set $Q$ has a known correspondence to a point in a set $H$. Arun et al. [10] with improvements by Umeyama [11] presented methods for finding a closest match in 3D using least-squares. In fact, Arun improved upon prior work in the area [12,13]. Their goal was to find a transformation, $T$, to minimize

$$e^2 = \sum_{i=1}^{N} \|h_i - T(q_i)\|,$$  \hspace{1cm} (1.1)
where \( T(q_i) = Rq_i + t \) [10] for \( q_i \in Q \) and \( h_i \in H \). Notice that \( R \) is a rotation matrix, \( t \) is a translation vector. The sets \( Q \) and \( H \) are aligned with a rotation using principal component analysis and a translation to align the centroids.

One advantage of this approach is its ease of use. However, the early least squares methods only find the minimum distance for a given correspondence. If the correspondence is unknown, a higher level algorithm would be needed to search the various correspondences for a global minimum. During the past 15-20 years, point matching research has focused on non-correspondence based methods. These algorithms treat the correspondence as a dynamic relationship and use transformations yielding the shortest distance between point sets.

### 1.1.2 Non-correspondence Based Point Matching

In a seminal article, Besl and McKay [9] introduce the iterative closest point (ICP) algorithm to register two point patterns. A correspondence between patterns \( Q \) and \( H \) is initialized such that each point in \( Q \) corresponds to its nearest neighbor in \( H \). Then \( H \) is transformed to minimize the distance between \( Q \) and \( H \). At each iteration, a new correspondence is generated and the distance is minimized. This algorithm converges to a local minimum based on the initial transformation of \( Q \).

The ICP paper continues by providing a method for creating a set of transformation initializations in an attempt to find a global minimum.

The ICP algorithm’s popularity is evident by continued research in the area. Most recently, Liu [14] has presented refinements to the ICP algorithm culminating in the improved ICP (I-ICP). Furthermore, Liu cites over 20 additional references that report potentially useful modifications to the original ICP algorithm. The improvements
include methods to reduce the effects of occluded and outlier points, to create better initial transformation sets, and methods to find the nearest points when making correspondences.

In addition to the ICP algorithm, the squared error measure could be adapted to many point set registration methods. For a recent example, Cordon et al. [15] present a scatter search method based on a least squares distance measure. They give experimental results showing that the algorithm runs significantly faster than the I-ICP. Other authors have proposed modifications to the way the least squares distance is measured. In one notable method, Agrawal et al. [16] penalize the squared-error measurement based on the number of unmatched points.

To eliminate the unknown correspondence between sets of vectors altogether, a noncorrespondence based registration and distance metric is desirable [3,17,18]. The Hausdorff measure is a well-known method for representing the distance between two point sets without having an a priori correspondence between the two sets. Huttenlocher et al. [17] is the first team to apply the classical Hausdorff measure [19] concept to matching spatial point sets starting with four definitions,

**Definition 1.1.1.** Directed Hausdorff (HD). Given two finite point sets \( P = \{p_1, \cdots, p_N_p\} \) and \( Q = \{q_1, \cdots, q_N_q\} \), the HD is defined as

\[
H(P, Q) = \max(h(P, Q), h(Q, P))
\]  

(1.2)

where

\[
h(P, Q) = \max_{p_i \in P} \min_{q_j \in Q} ||p_i - q_j||, \quad (1.3)
\]

and \( ||\cdot|| \) is a norm (typically Euclidean).

**Definition 1.1.2.** Directed Hausdorff Distance (DHD). The function \( h(P, Q) \) from the HD definition is called the Directed Hausdorff Distance.

**Definition 1.1.3.** Directed Partial Hausdorff Distance (DPHD). The DPHD is the \( K \)th smallest distance from \( P \) to \( Q \) denoted

\[
h_K(P, Q) = K_{p_i \in P} \min_{q_j \in Q} ||p_i - q_j||. \quad (1.4)
\]
Definition 1.1.4. Partial Hausdorff Distance (PHD). The PHD is defined as

\[ H_{LK}(P, Q) = \max(h_L(P, Q), h_K(Q, P)) \]  

where \( L \) and \( K \) allow for independently sized subsets of \( P \) and \( Q \).

Essentially, the DHD is the distance of the most isolated point between \( P \) and \( Q \). However, an outlier or occlusion could skew an otherwise close correspondence, which motivates using the \( K^{th} \) minimum as shown in the DPHD. Also, notice that \( h(P, Q) \) is not necessarily equal to \( h(Q, P) \). Depending on the application, using the directed or non-directed version of the measure may be desirable. In Chapter 4, we extend the PHD concept to classify sets of attributed point patterns.

1.1.3 Methods that are Invariant to Transformations

Rather than searching through transformations, feature-based methods represent imagery by properties that are invariant to transformations. For a simple example, one could find the centroid of an object and store the distance to the extremities as a feature vector; a classifier could train from these vectors. In radar, it is common to use global features, such as signal strength, in the constant false alarm rate (CFAR) detector [20]. More specific features such as vehicle length and width are more difficult to determine and could be difficult to determine if some attributed points are occluded in imagery [21].

Since interpoint distances are invariant to translation and orientation, a popular set of feature based methods represent the data in an adjacency matrix [22], which is a variation of graph-based methods [23]. In the adjacency matrix, each column represents some distance measure to every other point in the set. Several
algorithms [22, 24–26] explore point set comparisons using spectral properties of adjacency matrices. Unfortunately, the spectral approaches yield matching performances that break down in the presence of noise, clutter, and occlusion [25].

Instead of gaining translation and rotation invariance in an adjacency matrix, it is possible to align a set (or subset) of points to a local frame of reference determined by two points in a plane (in a 2 dimensional alignment) or some directional attribute [27–30]. As a result, the problem reduces to classifying unordered sets of vectors. Grauman [31] provides a nice review of unordered set matching along with a method called pyramid match hashing (PMH) for a fast database-driven approach to partial set classification. In Chapter 5, we extend the PMH to our SAR application and provide a Bayesian interpretation.

1.2 Evaluation of Algorithms

We created two databases to demonstrate the capabilities of the proposed processing and classification algorithms. For one experiment, the Gotcha public release data set [4] provided phase history for eight circular orbits of a parking lot containing seven civilian vehicles. We extracted imagery for the seven vehicles and classified vehicles from the last seven orbits, using the first orbit for training data. The experiment was first developed and applied to a PHD based classifier [32].

Second, using a physical optics electromagnetic scattering prediction code, we generated a phase history for ten civilian vehicles on an asphalt halfspace [5]; the data set is named CVDomes. The vehicle facet models represent a variety of small, mid, and large size civilian vehicles simulated from the late 1990s to early 2000s. By moving a vehicle to different locations in a large scene, variations in elevation
to the radar from 35 to 59 degrees cause vehicle signatures to differ primarily due to layover [1] changes. Thus, we rotated and/or translated the ten vehicles to 50 positions throughout the large scene, using 6 positions of each for training. The CVDomes experiment was originally tested using a PHD based classifier [33].

The mean squared error (MSE) classifier provides a benchmark for evaluating the proposed classification approaches. MSE provides a generalized likelihood ratio test for translated patterns in additive white Gaussian noise, and is therefore widely used; but, MSE is known to be brittle in the presence of occlusion, clutter, and other non-ideal conditions [34–36]. The MSE classifier includes a 2D FFT-based registration to effectively minimize the MSE over translations in a plane; however, non-ideal conditions cause the registration process, and subsequently, the classification, to fail.

Although image resolution limits the ability to distinguish point sets from two vehicles observed from a given viewing angle [37], the fusion of a diversity of vehicle aspects has been shown to greatly enhance discrimination [38–40]. For the small radar cross sections of civilian vehicles, the wide-angle imagery is particularly important to gain enough viewing aspects to make accurate decisions [32]. Since the MSE algorithm was originally developed for a narrow aperture application, we adapted the algorithm to classify wider aperture imagery by minimizing the MSE over pose rotations [39].

A fusion of aspects may improve performance of the MSE algorithm; however, clutter, articulation variations, and physical occlusions are problematic [40]. This is a motivation for using attributed scattering center-based methods to find a correspondence between the query and database entries. A limited correspondence can ignore clutter and occlusions while finding matching subsets within images. In addition, it is easy to attach attributes to the points such as polarization or azimuth to increase the
Table 1.1: Comparison of four classifiers

<table>
<thead>
<tr>
<th>method</th>
<th>trans.</th>
<th>pose</th>
<th>partial</th>
<th>occlusion</th>
<th>mem. (MB)</th>
<th>complex. (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>⚫</td>
<td></td>
<td></td>
<td></td>
<td>1420.0</td>
<td>366</td>
</tr>
<tr>
<td>MBF</td>
<td></td>
<td>⚫</td>
<td></td>
<td></td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>PHD</td>
<td>⚫</td>
<td></td>
<td>⚫</td>
<td>⚫</td>
<td>0.2</td>
<td>900</td>
</tr>
<tr>
<td>PMH</td>
<td>⚫</td>
<td>⚫</td>
<td>⚫</td>
<td>⚫</td>
<td>7.7</td>
<td>15</td>
</tr>
</tbody>
</table>

diversity of information. In contrast, the MSE algorithm integrates all of the pixels, clutter or not, during the registration and comparison processes.

Table 1.1 summarizes a comparison of classification algorithms. In addition to the MSE and PMH based classifiers, we consider a partial Hausdorff distance (PHD) classifier [32] and a model-based Bayesian feature (MBF) method [41]. The PHD algorithm minimizes a pseudometric between attributed point patterns over rigid transformations using a nonconvex optimization [3, 32, 33, 42]. The MBF assumes known translation and orientation, and uses the Hungarian algorithm to find a point correspondence for partial matching. MBF returns an approximate posterior relative probability, conditioned on the correspondence.

Considering Table 1.1 columns from left to right, the MSE, PHD, and PMH methods all provide for an unknown translation. For orientation, MSE requires either a separate pose estimate or a search over pose. The MBF method requires prior estimates of both translation and orientation as inputs. The MBF, PHD, and PMH can provide partial matching by design. The matching is partial and robust in that images are locally represented by a varying number of features, and individual features cannot significantly alter the similarity measure [31, 41]. MSE, on the other hand, inherently provides a global match score.
The MSE registration process is brittle under occlusions of aspect views, and the MBF model imposes conditional independence of missing features, in contrast to the simultaneous loss of features due to an occluded view. Thus, only the PHD and PMH are designed for point set matching in the presence of occluded viewpoints.

The complexity and memory columns of Table 1.1 provide comparison for the specific 10-class problem described in Section 2.2.1. The combined conditions of pose, elevation angle, and viewing angle cause the memory requirement of an MSE template-matching scheme to grow large with the size of the database [43]; the memory requirements listed in column 6 of Table 1.1 demonstrate the significant compression provided by the attributed point representation [44]. The point clouds used in MBF and PHD provide three orders of magnitude reduction in memory requirements. The local representation adopted in PMH not only yields invariance to unknown translation and orientation but also provides the computational advantage of hash codes. These advantages increase memory, versus PHD, but still represent a 200:1 compression versus the MSE image templates.

Complexity is reported simply as single-threaded query times on an Intel Core i7-920 processor. The PHD method required 900 seconds, with execution dominated by optimization over the unknown translation and pose. Likewise, MSE has a large execution time of 366 seconds. In contrast, the PMH method required only 15 seconds to classify a query.

The MSE method has complexity given by the product of the number of pixels and the library size. The library grows as the product of the number of classes and the number of samples needed to represent the azimuth and elevation variation of anisotropic millimeter wave scattering. The unknown translation and orientation...
preclude the use of tree-search and support vector preprocessing techniques for accelerating the match against the library of templates.

The complexity of the MBF method is dominated by the $O((m + n)^3)$ operations required to find the minimal correspondence, where $m$ and $n$ are the number of points in the compared samples. Assuming a 100 times increase in processing power since publication of [41], query times from [41] for a 60 entry database, are estimated at over 7 minutes. Moreover, the MBF query requires known translation and orientation; hence, the MBF approach presents a significant computational burden.

The PHD method uses a kd-tree to find the nearest neighbors between point patterns with time-complexity $O(mfn^{1-1/f})$, where $f$ is the dimension of the features. A nonconvex optimization multiplies the complexity by an estimated $10^5$ in search of a global minimum over unknown translation and orientation, thus driving up the computation time [33, 42].

The invariant PMH method is comparatively very fast. A measure of correspondence is calculated in $O(fK\log N_0)$ [45], where $N_0$ is a quantized feature range and $K$ is the size of invariantly mapped local sets. Additionally, due to a hash coded database and a fast Hamming nearest neighbor search, the time complexity can grow sublinearly in the size of the database.

1.3 Summary of Contributions

This dissertation provides the first demonstration of wide-angle radar classification, and we provide the first civilian vehicle classification results for two multiclass datasets. We represent the radar imagery using a fast attributed scattering center extraction method. The data is visualized in 2D, 3D, and with attributes.
By extending concepts from the image registration and pattern recognition communities, we show classification results using two new point pattern classification algorithms. The first is a partial Hausdorff based distance that is minimized over rigid transformations, where comparisons between point clouds can be visualized as single points in a Euclidean space for classification, visualization, and analysis. The second algorithm applies a pyramid match hashing based method to invariantly mapped local groups. A Bayesian interpretation is provided for the pyramid match kernel.

The two classification algorithms show strong performance discriminating the similar signatures of civilian vehicles in 22 cm resolution SAR imagery. Both algorithms are invariant to translations and pose rotations, and they are robust to partial matches and occlusions. The algorithms use a small amount of memory relative to image template based methods, and the PMH based algorithm in particular operates in real-time.

1.4 Organization of Chapters

This dissertation is comprised of two main topics for civilian vehicles in wide-angle SAR. Chapters 2 and 3 are concerned with data representation. Chapter 2 discusses the creation of databases of civilian vehicles based on extracted attributed scattering centers, while chapter 3 provides a method to further process the data by extracting limited three-dimensional information from a single pass of the radar platform. The second half of the dissertation provides two novel classification algorithms with application to the databases. Chapter 4 presents a classification algorithm that uses
a psuedometric based on a partial Hausdorff distance that is minimized over rigid
transformations in a plane, and Chapter 5 shows a classifier that uses hash-coded,
pose-invariant, local regions in a nearest neighbor, database-driven classifier.
CHAPTER 2: Wide-angle Attributed Point Patterns

The high-frequency scattering response of an object is well approximated as a sum of scattering centers [46], and it is possible to discriminate amongst various objects using an unordered set of feature points [32, 41, 47-50]. These attributed scattering centers are also called feature points or point clouds both for brevity and in reference to their intended use for pattern recognition [10, 32]. The scattering centers can include 2D or 3D location, azimuth, amplitude, polarization, and a variety of other attributes [2].

Although radar bandwidth limits the ability to distinguish point sets from two vehicles observed from a given viewing angle [37], the fusion of a diversity of target aspects can greatly enhance discrimination [38-40]. In this dissertation, wide-angle SAR imagery is represented as sets of attributed scattering centers generated from a collection of subaperture images.

More specifically, a circular SAR generates a set of data called a phase history from a discrete set of radar pulses as the aircraft orbits a scene of interest. From the wide-angle phase history, a sequence of subaperture 2D images is generated using backprojection. In general, we assume a previously generated digital elevation map (DEM) and 2D image processor are available to process a given data circular SAR collection [51-53], thus producing “an orthographically correct image free of any geometric or defocus effects from wavefront curvature and also free of the effects of
Figure 2.1: The image in (a) was generated from the Gotcha Public Release Dataset; the image has a 5-degree aperture centered broadside to the automobile. Detected peaks in (b) display an abstraction of the image with scattering centers plotted onto a pixel array.

terrain-elevation-induced defocus [51].” Additionally, the data collection should be well motion compensated [54].

For use in applications, we have phase history from a 2006 Data collection [4], which is well-focused on a relatively flat terrain, and we have the CVDomes data set [5] that was simulated on a level ground plane. The following sections describe our process for approximating and representing attributed scattering centers that are found from local peaks in magnitude images.

### 2.1 Building sets of attributed scattering centers

To illustrate our application of attributed scattering centers, first consider Figure 2.1a, which shows an example 5 degree subaperture horizontal polarization image (HH) taken from the Gotcha Public Release Dataset [4]. The dataset consists of SAR
phase history data collected at X-band with a 640 MHz bandwidth, full azimuth coverage, and full polarization. Figure 2.1b shows extracted peak locations. (Throughout this dissertation, we use 256 pixels per 15 meters for the 2006 data and 256 pixels per 10 m for the CVDomes data.) These peak locations give a compressed representation of the target signature for the specific viewing angle. Here, we have extracted peaks by first processing the magnitude of the bandpass backprojection images [53] with a 2D Wiener filter as a simple smoothing filter to reduce the incidence of multiple local maxima around a scattering center location. Then we find all local maxima that are within a specified dB range that is referenced to the global maximum of the image set; in particular we used -25dB and above. In practice, one could select the threshold based on the average radar cross section of the clutter per pixel area. The local maxima are effectively found by moving a $3 \times 3$ window around the image and noting when the center pixel has greater magnitude than the other eight pixels (Matlab imregionalmax [55]).

More computationally intensive algorithms exist for extracting attributed scattering centers from SAR imagery [56–59], and they could be used in the proposed applications. Our simple peak extraction method, however, facilitates fast processing of a large set of images, requiring approximately 50 ms to extract attributed scattering centers from a $256 \times 256$ image.

In addition to spatial location $(x, y, z)$, other attributes are associated with each peak. The azimuth angle $\theta$ and elevation angle $\phi$ are defined from the vehicle location to center of the radar subaperture; these angles serve to encode the anisotropic nature of the target reflectivity [60,61]. A scattering magnitude $a$ is reported for each peak.
Finally, we adopt a binary attribute, $\nu$, to record from which of two polarimetric images the peak was extracted: even-bounce or odd-bounce. With a linear polarization basis, let $HH$, $VV$, and $HV$ denote the horizontal co-polarized image, the vertical co-polarized image and the cross-polarized image, respectively. The odd-bounce image is $HH + VV$, while the even-bounce image is $HH - VV$ [21,62–64]. If desired, a more complete description of the Stokes matrix can be adopted in lieu of the two attributes, $a$ and $\nu$ [64].

Figure 2.2a illustrates a composite of all detected peaks for a single vehicle from 90 subaperture images, with a 1 degree overlap. For example, the horizontal slice at 175 degrees azimuth is the set of peaks shown in Figure 2.1b. We refer to this composite collection of attributed peaks as a point cloud. Each point in Figure 2.2a is plotted using the $(x, y, \theta)$ information from an attributed peak, where the $x$ and $y$ axes are labeled in pixels. The elevation $\phi$, amplitude $a$, and polarization $\nu$ attributes are suppressed in this visualization; color encodes the azimuth angle, $\theta$. The spiral pattern in Figure 2.2a arises due to layover of the roofline returns, which are visible as an outer ring of points in the top view in Figure 2.2b.

The even and odd-bounce composite images are shown in Figure 2.2c and Figure 2.2d, respectively. When generating 2D SAR imagery of a vehicle on asphalt, it is typically possible to visualize an inner and outer ring of scattering. On level terrain, the inner ring is generated by radar echoes that experience a two-bounce path between the asphalt and the side of the vehicle; thus, the inner ring represents time-of-flight to the base outline of the vehicle [21]. In contrast, the outer ring arises from radar echoes reflected by elevated portions of the vehicle, such as the roof line. These elevated features project onto the image in a direction orthogonal to the radar
Figure 2.2: The plot in (a) shows the extracted 2D scattering location for individual azimuths stacked into a cube. For visualization, the color of the dots corresponds to the center azimuth angle of the subaperture image in which a peak is detected. A vehicle has a spiral pattern in the cube caused by the leading edge of the vehicle at any given look angle. When flattened to a top view in (b), the point cloud reveals the vehicle’s 360-degree shape. The SAR even-bounce (d) and odd-bounce (e) images are composites of 5-degree aperture images for the full 360-degree extent around a vehicle. Blue even and green odd points are shown in (e).

line of sight – an effect known as layover [1]. The polarization feature, $\nu$, encodes even versus odd-bounce echoes. The even bounce information coincides with the inner ring of scattering centers of Figure 2.2b, while the odd bounce information generally coincides with the outer ring. Figure 2.2e shows extracted peaks with blue for even and green for odd-bounce. The inner ring is suppressed but visible in the odd-bounce
image due to the relative strength of non-ideal dihedral base to the single-bounce points in the outer ring.

After extraction from imagery, a scattering center is represented by a vector

\[ \mathbf{p} = (x, y, z, a, \theta, \phi, \nu)^T, \]  

and a vehicle sample is represented by a set

\[ P = \{ \mathbf{p}_1, \cdots, \mathbf{p}_{N_p} \}, \]  

containing \( N_p \) attributed scattering centers in the collection subaperture images. Individual chapters may use \( Q \) or \( H \) in lieu \( P \), and not all applications use the \( z \) and \( \phi \) features. Additionally, the single-pass 3D section uses prime and unprimed versions of the vectors in \( P \) to distinguish local from global coordinates. The next section describes two databases of attributed scattering centers.

2.2 Civilian Vehicle Circular SAR Databases

In this section, we describe the design of two databases that are used throughout this dissertation. The first data set is synthetically generated radar imagery using ten vehicles of the CVDomes data set [5]. The second database contains measured airborne SAR scattering data for seven vehicles [4], referred as the 2006 collection.

2.2.1 Experiment Design using CVDomes

Using a physical optics electromagnetic scattering prediction code [5, 6], we generated a phase history for ten civilian vehicles in Table 2.1. The vehicle facet models represent a variety of small, mid, and large size civilian vehicles produced during the late 1990s to early 2000s. The facet models were placed on a dielectric half-space
Table 2.1: Vehicles identified by a number in experiments. For example, the Camry may be referred as 1 or v1.

\[ \epsilon = 5.9 - j0.1, \mu = 1.0008 \] to simulate an asphalt surface [65]. Automotive glass was modeled as a 0.635 cm layer \( \epsilon = 5.9 - j0.15, \mu = 1.0 \), and rubber was modeled as an absorber. Facets were otherwise assumed to be perfect electrical conductors.

The data were generated for 360 degrees in azimuth and for elevation angles from 30 to 60 degrees; phase history samples were taken for 5.35 GHz bandwidth centered at 9.6 GHz. The resulting set of data domes for ten vehicles is referred by the name CVDomes. From the CVDomes data set, we used 640 MHz of bandwidth centered at 9.6 GHz for 360 degrees in azimuth and 35 to 59 degrees in elevation.

Figure 2.3 shows the geometry for a circular SAR collection as related to a vehicle on the ground. The radar \( R \), along with its ground projection \( R_v \), orbit the origin \( O \) in a circular path. The angle of elevation \( \phi' \) from a vehicle to the radar varies throughout the orbit, except for a vehicle at the origin.

An interesting problem is to investigate the effects of placing a vehicle a ratio of the flight path radius from the scene center. For this problem, let us assume that
Figure 2.3: For simulation, a vehicle $V$ is translated in the $xy$ plane, and the radar $R$ orbits the scene center $O$ with a constant elevation $\phi_0 = 45^\circ$.

the terrain has a constant altitude that is the same as the scene center as shown in Figure 2.3, and a vehicle location $V$ can be represented by the vector

$$v = \|v\| [\cos \theta_v, \sin \theta_v, 0]^T,$$  \hspace{1cm} (2.3)

where $(\|v\|, \theta_v)$ is the vehicle location in polar coordinates on the $xy$ plane. Furthermore, the radar vector $r$ shall orbit with a constant elevation $\phi_0 = 45^\circ$ with respect to the scene center, where $r_v$ is the projection of $r$ onto the $xy$ plane. Thus $\|r_v\|$ is the flight path radius. Let the ratio of the vehicle offset to flight path radius be defined as

$$q = \frac{\|v\|}{\|r_v\|}. \hspace{1cm} (2.4)$$

As a result, using complex notation, the imaging elevation referenced to location $V$ simplifies (see Appendix A.3 for a derivation) to

$$\phi' = \cot^{-1} \left| e^{i\theta} - q e^{i\theta_v} \right|. \hspace{1cm} (2.5)$$
Hence at each radar azimuth $\theta$, we generate a subaperture image from a different elevation slice of the simulated phase history defined by $\phi'$. We index the simulated data by the $(q, \theta_v)$ pair.

Given that the simulation data contains only elevations from 30 to 60 degrees, $q$ is constrained. Applying (2.5) and $q = 0.4$ gives

$$\max_{\theta_v, \theta \in [0,360]} \phi' = 59.0 \text{ and } \min_{\theta_v, \theta \in [0,360]} \phi' = 35.5.$$  

Hence we use $0 \leq q \leq 0.4$ for simulations.

As described in Chapter 2.1, imagery is generated from the phase history data using backprojection over a 5 degree sub-apertures; we extracted 360 images around the SAR orbit with a 20 percent overlap. Figures 2.4 (a),(b),(e),(f),(i), and (j) show example images for v4, v8, and v9, where all 360 images are coherently added to give a 360 degree representation of a vehicle.

Depending upon the pose and location of the vehicle within the scene, scattering features may migrate or not persist, and the elevated features will exhibit different layover as a function of elevation angle. Thus the same vehicle in two different locations can appear significantly different, as shown in the first two columns of Figure 2.4. The first column shows imagery of the vehicle centered in the scene, while the second column shows the vehicle offset along the x-axis by 0.4 times the flight path radius with a 108° pose rotation. Figures 2.4 (m) and (n) show the position of the vehicles within the scene. The last two columns of Figure 2.4 provide quiver plots of the location and azimuth information that is derived from the corresponding imagery in the first two columns (the amplitude and bounce attributes are not displayed).

Another example of 360 degree vehicle imagery is shown in Figure 2.5. The top left image shows the vehicle (v5) centered in the scene, while the remaining images
Figure 2.4: The left two columns show 360 degrees of SAR imagery for three sample vehicles. For the first column the vehicles are centered in the scene as shown in (m), while the vehicles are offset in the second column as shown in (n). The third and forth columns show extracted scattering center locations that are attributed with an azimuth attribute. The arrows point in the azimuth direction.
Figure 2.5: If v5 is offset from the center of a large circular SAR scene it will experience different elevations to the radar over the flight path. As a result, the same vehicle looks different when translated around the scene. We generated simulated vehicle imagery for each of the dots in the graph; imagery for six of the locations are displayed. The circled dots are locations used in the classification training set.

show the vehicle offset by 0.4 times the flight path radius in five different locations in the scene. By extending this example, we designed a database characterizing vehicles located throughout the scene.

Based on the dots in the right half of Figure 2.5, we generated 50 sets of images for each of the ten vehicle classes in Table 2.1. There are ten 36-degree pose rotations at the center and ten locations on each of the four concentric rings about the center. Referring to Figure 2.3, each vehicle location \( v = ||v|| e^{j\theta_v} \), in polar coordinates, is represented by the pair \((q, \theta_v)\), where \( q \in \{0.0, 0.1, 0.2, 0.3, 0.4\} \) is the ratio

\[
q = ||v|| / ||r_v||
\]  

(2.7)

and \( ||r_v|| \) is the flight path radius. The resulting database contains 500 sets of images. Each set of images contains 360, five degree, subaperture images with 80 percent overlap. (To reduce complexity, the data set is usually decimated to 90 images with
Figure 2.6: By extracting attributed scattering centers from the imagery shown in Figure 2.5, vehicles are abstracted to a set of attributed points. Among the attributes for a scattering center are 2D location and azimuth to the radar. Darker points in the plots start with a zero degree azimuth in the $x$ direction; lighter points represent increasing azimuths as the radar orbits counterclockwise from the $x$-axis. The plot to the right shows a spiral pattern by representing the azimuth on a $z$-axis.

20 percent overlap for the applications.) The six circled locations in Figure 2.5 are set aside as training samples. Thus, samples in the set

$$\{(0.0, 0°), (0.3, 0°), (0.3, 72°), (0.3, 144°), (0.3, 216°), (0.3, 288°)\} \quad (2.8)$$

are used for training, while the remaining 44 locations are used as classification samples.

For all images in each set, we implemented peak extractor of Chapter 2.1 to approximate scattering center locations. Figure 2.6 shows example views of $\{x, y, \theta\}$ for the images of Figure 2.5. As azimuth increases counterclockwise from the $x$-axis, the azimuth attribute goes from dark to light points. The 3D plot shows the azimuth
Figure 2.7: There are seven identifiable civilian passenger vehicles in 2006 data collection parking lot. The top left image shows the seven vehicles in one scene, while the remaining images show close-up views of each vehicle independently. We extracted attributed scattering centers of these seven vehicles for all eight orbits of the radar.

on the $z$-axis, and the leading edge (to the radar) of the vehicle is apparent by the spiral pattern.

### 2.2.2 Experiment Design using 2006 Collection

A 2006 data collection [4] provided airborne circular SAR phase history for a parking lot scene containing seven documented civilian vehicles. The data set contains eight orbits of the radar. The top left of Figure 2.7 shows an image of the parking lot with numeric labels for the seven vehicles, while Table 2.2 cross-references the label numbers with vehicle model.

Using the same imaging technique from Section 2.2.1, we isolated the seven vehicles as shown by the seven remaining images of Figure 2.7. Each image is a sum of 90, five degree aperture, backprojection images with 20 percent overlap around the
Table 2.2: Vehicle labels in 2006 Data

<table>
<thead>
<tr>
<th>number</th>
<th>model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Prizm</td>
</tr>
<tr>
<td>2</td>
<td>Maxima</td>
</tr>
<tr>
<td>3</td>
<td>Sentra</td>
</tr>
<tr>
<td>4</td>
<td>Santa Fe</td>
</tr>
<tr>
<td>5</td>
<td>Camry</td>
</tr>
<tr>
<td>6</td>
<td>Taurus</td>
</tr>
<tr>
<td>7</td>
<td>Malibu</td>
</tr>
</tbody>
</table>

SAR orbit. To generate attributed point patterns, we extracted the same information as described in (2.1).

For the experiment, the attributed point patterns from the first orbit of the SAR are used as training data. Then, in the subsequent seven passes, the vehicles are identified based on the training data.
CHAPTER 3. 3D Imaging in Wide-angle Radar

The material for this chapter is based on the manuscript: Dungan, K.E. and Potter, L. C., “3-D imaging of vehicles in wide aperture radar”, IEEE Transactions on Aerospace Electronic Systems, to appear.

3.1 Introduction

Persistent radar surveillance is enabled by technological advances and emerging operational concepts. Advances in digital data rates and storage capacity allow high-resolution synthetic aperture radar (SAR) mapping of a large scene [4]. Circular flight paths provide persistent coverage and maximal diversity in azimuthal viewing angle [66,67]. Yet, the combination of a large scene and wide-angle viewing dictates that elevation angle varies significantly across the scene. Consequently, the radar signature of an object may vary significantly as a function of location within the scene. This effect is illustrated in Figure 3.1 for X-band signatures with 640 MHz bandwidth. Figure 3.1a is a composite of 5-degree aperture SAR images around the full 360-degree flight path, while Figure 3.1b shows the same image with the vehicle offset from the scene center by 0.4 times the flight path radius.

The variation in the image signature versus elevation angle is due to the layover effect [1] arising from the 2D representation of a 3D object. Figure 3.2 illustrates layover. Figure 3.2a shows a ray-tracing on a facet model of a vehicle, illustrating
Figure 3.1: The same vehicle imaged at the scene center (a) and offset by 0.4 times the radius of the flight path (b) yields markedly different signatures.

the even-bounce multipath returns from the vehicle side and the odd-bounce direct returns from the roofline. As illustrated in Figure 3.2b, a roofline reflector at point E (with respect to a nominal vehicle base location reference V), being elevated above the ground surface, experiences layover when imaged to the ground as a point P. The length of layover is quantified by

\[ l \approx h \tan \phi', \quad (3.1) \]

where \( \phi' \) is the elevation angle from P (or E) to the radar, \( h \) is the height of E above its ground projection \( B_E \), and the layover \( l \) is the distance from \( P \) to \( B_E \).

More generally, the vehicle can be sitting at a different altitude and terrain grade with respect to the scene center. Chapter 2 describes the imaging technique as back-projection to a DEM. As a result, we have a 3D unit normal near the base of the vehicle following a rough vehicle detection [20].

Figure 3.3 shows an offset imaging plane for a circular SAR radar \( R \) that is orbiting the scene center \( O \) with a nominal depression angle \( \phi_0 \). (Notice that the simplified Figure 3.2b is a side view of the \( x'y' \) plane in Figure 3.3.) A detection [20] on the \( x'y' \)
Figure 3.2: Ray tracing (a) on a facet model (Ford Taurus wagon on a dielectric half-space) to illustrate even-bounce and odd-bounce reflections. (b) Geometry of layover.

The image plane provides the location $V$ of a vehicle with respect to the scene center $O$ and a unit normal in the direction of $z'$. The $O'$ coordinate system, centered at $V$, is described in detail in Chapter 3.2. An elevated point $E$ experiences layover onto the imaging plane as point $P$. Since the radar $R$ is relatively far from $E$, we assume that the radar elevation angle $\phi'$ with respect to the $x'y'$ imaging plane is the same for any point on the local image surface. This assumption necessarily limits the application such that $\phi'$ is never close to 90 degrees since the layover becomes arbitrarily large for a radar wavefront that is nearly normal to the direction of object height. (i.e. $l \to \infty$ as $\phi' \to 90^\circ$).

Due to limited azimuthal persistence of reflectors, a wide-angle aperture offers very little 3D resolution. For example, a 10-degree beamwidth at X-band with 640 MHz bandwidth (23 cm downrange resolution) provides a resolution in the third dimension of only approximately 20 m [68]. However, using two coherent passes of the radar, it
Figure 3.3: The local imaging surface for a vehicle location is offset from the scene center and has a different normal vector. Layover occurs in the direction of the radar $R$.

is possible to get 3D information using interferometric synthetic aperture radar (IF-SAR) [1, 69–71]. With more passes of a circular SAR, an application called elevation circular SAR has been shown to provide 3D information [66].

As shown in Figure 3.1, signatures vary significantly across a scene that is large relative to the flight path radius due to change in elevation angle. In this paper, we propose an image processing procedure to gain partial invariance to elevation angle and produce a 3D signature representation requiring approximately 20 kB storage per vehicle for object recognition.

Table 3.1 provides a summary for the process of generating 3-D imagery of vehicles from a circular SAR collection; steps 1 and 2 are assumed inputs to the algorithm. The remaining algorithm steps are discussed in the sections of the paper as follows. Chapter 3.2 describes how to represent a circular SAR data collection using a set of attributed scattering centers referenced to both global and local coordinate systems. Next, in Chapter 3.3 using the information in the set of scattering centers and a simple vehicle model, we present a method to extract 3D information from a single
Table 3.1: 3D Processing Algorithm

<table>
<thead>
<tr>
<th>step</th>
<th>description</th>
<th>ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Backprojection imaging to DEM</td>
<td>ref. [51]</td>
</tr>
<tr>
<td>2</td>
<td>Vehicle detection</td>
<td>ref. [20]</td>
</tr>
<tr>
<td>3</td>
<td>Extract attributed scattering centers</td>
<td>sec. 3.2</td>
</tr>
<tr>
<td>4</td>
<td>Solve for vehicle base rectangle</td>
<td>sec. 3.3.2</td>
</tr>
<tr>
<td>5</td>
<td>Invert layover of suspected elevated points</td>
<td>sec. 3.3.3</td>
</tr>
<tr>
<td>6</td>
<td>Create visualizations</td>
<td>sec. 3.4</td>
</tr>
</tbody>
</table>

pass of the SAR platform. Finally, in Chapter 3.4, using the data from the 2006 collection [4] of Chapter 2.2.2 and the CVDomes [5] simulation of Chapter 2.2.1, we provide experimental results showing a variety of 3D images and demonstrating invariance to vehicle location in the scene.

3.2 Attributed Scattering Centers

Chapters 2.1 and 2.2 describe a process for extracting sets of attributed scattering centers for two databases. Each scattering center is described by a feature vector (2.1); in this chapter the unprimed features reference the global coordinate system at the scene center, while the primed coordinates are referenced to the local imaging plane.

A target sitting on the DEM surface presents some planar shift \((\Delta x', \Delta y')\) and a pose rotation, \(\Delta \theta'\); the pose is a rotation in the \(x'y'\) plane. As the target location varies from scene center, the elevation and azimuth angles from a target to the radar vary. In order to gain partial invariance to the radar viewing angles, in the next
Section we propose a procedure to extract from the two-dimensional image a three-dimensional position for each reflector using layover and polarization from 2D images. The 3D representation is the key for obtaining invariance to change in elevation angle versus vehicle position in the large scene. For layover points identified with the aid of the polarization attribute, we replace a 2D peak location with a 3D position \((x', y', z')\). Thus, each point \(P\) in a target point cloud has a vector referenced to the local imaging plane of the form

\[
\mathbf{p'} = (x', y', z', \iota', \theta', \phi', \nu)^T.
\]  

3.2.1 Coordinate system conversions

Our approach is facilitated by choice of two coordinate systems. For a global coordinate system \((x, y, z)\) we select the scene center (i.e., phase reference point) as the origin, \(\hat{x}\) in the east direction, \(\hat{y}\) in the north direction, and \(\hat{z}\) in the up direction. Consider the detection of a possible vehicle at location \(V\). Define local coordinates \((x', y', z')\) where \(\hat{z}'\) is the unit-length outer normal to the local DEM surface (i.e., \(z'\) is in the \(+\hat{z}\) half-space), and the local origin \(O'\) is located at point \(V\).
Fig. 3.4 describes an affine transformation between two orthonormal coordinate systems, \( A \) and \( B \). Let us define the affine transformation on homogeneous coordinates \([72]\) with
\[
A^B T \overset{\text{def}}{=} \begin{bmatrix} B & A \R & B O_A \\ 0^T & 1 \end{bmatrix}
\]
(3.3)
such that a vector \( A^p \) with reference \( O_A \) is transformed to a vector \( B^p \) with reference \( O_B \) by
\[
\begin{bmatrix} B^p \\ 1 \end{bmatrix} = A^B T \begin{bmatrix} A^p \\ 1 \end{bmatrix}.
\]
(3.4)
The pre-superscript \( A \) and \( B \) designate the reference for the point being transformed.

The rotation matrix from basis \( A \) to basis \( B \) is defined by
\[
A^B R \overset{\text{def}}{=} \begin{bmatrix} \hat{i}_A \cdot \hat{i}_B & \hat{j}_A \cdot \hat{i}_B & \hat{k}_A \cdot \hat{i}_B \\ \hat{i}_A \cdot \hat{j}_B & \hat{j}_A \cdot \hat{j}_B & \hat{k}_A \cdot \hat{j}_B \\ \hat{i}_A \cdot \hat{k}_B & \hat{j}_A \cdot \hat{k}_B & \hat{k}_A \cdot \hat{k}_B \end{bmatrix},
\]
(3.5)
while the point \( O_A \) referenced from \( O_B \) is stored in \( B O_A \).

Let the basis for \( A \) be the standard Euclidean basis such that
\[
[i_A, j_A, k_A] = I \in \mathbb{R}^{3 \times 3}
\]
(3.6)
with \( \hat{x} = i_A, \hat{y} = j_A \), and \( \hat{z} = k_A \) in our problem. From the DEM, we have the unit normal \( k_B \) for the local imaging plane along with vehicle location \( A^O_B \), where \( A^O_B = -B^O_A \). Then we define \( i_B \) as the normalized projection of \( i_A \) onto the plane normal to \( k_B \); hence
\[
i_B = \frac{(I - k_B k_B^T) i_A}{\| (I - k_B k_B^T) i_A \|}.
\]
(3.7)
Subsequently, we can complete the orthonormal basis for \( O_B \) using Gram-Schmidt, where
\[
j_B = \frac{j_A - i_B^T j_A i_B - k_B^T j_A k_B}{\| j_A - i_B^T j_A i_B - k_B^T j_A k_B \|}.
\]
(3.8)
Here we define the $A$ coordinate system as $O(xyz)$ for the scene center and the $B$ coordinate system as $O'(x'y'z')$ at point $V$ of the local imaging plane with $\hat{x}'$ in the direction of $i_B$, $\hat{y}'$ in the direction of $j_B$, and $\hat{z}'$ in the direction of $k_B$.

Capital letters in the figures of this chapter represent points in space, while bold lower case letters represent vectors to those points. Thus a vector to a radar position $R$ is represented as the vector $\mathbf{r}$ when referenced to the scene center and $\mathbf{r}'$ when referenced to the local imaging plane. The conversion from $\mathbf{r}$ to $\mathbf{r}'$ is accomplished using the transformation in (3.4).

The radar can also be expressed in spherical coordinates in either system (see Appendix A.1 for our application). In global coordinates, $\mathbf{r} = (\rho, \theta, \phi)$, where azimuth angle $\theta$ is from $\hat{x}$ towards $\hat{y}$ and elevation angle $\phi$ is up from the $xy$ plane. Likewise, in local coordinates $\mathbf{r}' = (\rho', \theta', \phi')$. For vehicle location $V$ displaced from the scene center $O$, the azimuth and elevation angles experienced at $V$ and $O$ can differ significantly. The result can be a large difference in radar signatures, as illustrated in Figs. 3.1a and 3.1b. Similarly, the azimuth from the vehicle to the radar stored in $\theta$ or $\theta'$ varies throughout the scene and orbit of the radar. Appendix A.2 describes the selection of radar aperture points to maintain constant cross-range resolution.

### 3.3 Extracting a 3D Point Cloud from Single Pass Imagery

In this section we present an automated fast image processing procedure to extract a 3D point cloud from single-pass 2D imagery. In order to extract 3D location from a 2D SAR image, we must postulate a simple model for vehicles, because the limited azimuthal persistence of reflectors results in very limited three-dimensional resolution from a wide-angle aperture [68]. The model captures scattering behavior across the
elevation angles present in a large scene, e.g., from approximately 30 to 60 degrees elevation.

3.3.1 Vehicle model for layover

First, from Figure 3.2a we observe that the two-bounce reflections from the ground and vehicle sides create a strong virtual dihedral return, which is persistent over a large range of elevation angles [73]. The apparent point of reflection is along the dihedral crease formed by the intersecting planes of the ground surface and vehicle side panel. Thus, from a top view, the dihedral returns from the vehicle base form an approximate rectangle. The dihedral return is emphasized in the even-bounce polarimetric image. For a Hyundai Sante Fe in the 2006 Collection, the even-bounce image is shown in Figure 2.2c. This 360-degree image is formed as the coherent sum of 5-degree subaperture images with no overlap. We display the top 20 dB with respect to the brightest pixel. In the top view of the point cloud (Figure 2.2b), these even-bounce terms appear as an inner ring of points.

Second, the direct returns at broadside come primarily from the roofline; the small radius of curvature between the side and top of the passenger cabin results in very limited migration of the reflection point over the elevation angles from 30 to 60 degrees. The direct returns are emphasized in the odd-bounce polarimetric image $HH + VV$. For the Hyundai Sante Fe in the 2006 Collection, the odd-bounce image is shown in Figure 2.2d. In the top view of the point cube (Figure 2.2b), these odd-bounce terms appear as an outer ring of points, due to layover. An extracted peak is mapped to the base rectangle along the associated azimuthal viewing angle, $\theta'$,
yielding the layover distance. For elevation angle φ', 3D height z' is then cot φ' times the layover by solving for the height in (3.1).

### 3.3.2 Extracting a Rectangle from a Set of Points

To determine the 3D position, we begin by constructing a rectangle from the even-bounce points in the point cube. Our method for extracting the rectangle begins with a set of N points and a parameterized rectangle using length (l) and width (w) under a rigid transformation defined by the 2D translation (t_x', t_y') and rotation γ. The vector \( \mathbf{\Lambda} = (l, w, t_x', t_y', \gamma) \in \mathbb{R}^5 \) defines a rectangle, where \( \gamma \in [0, 2\pi] \). Next, each of the N points has some Euclidean distance to the rectangle that is stored in a vector \( \mathbf{s} = (\xi_1, \xi_2, \ldots, \xi_N) \in \mathbb{R}^N \). Figure 3.5a shows example points projected onto a rectangle. The ideal sequence, where all points lie on the rectangle, is a vector \( \mathbf{c} = (0, 0, \ldots, 0) \in \mathbb{R}^N \).

To mitigate the effect of clutter points, we map \( \mathbf{s} \) and \( \mathbf{c} \) to inverse exponential sequences in \( l^p \), where

\[
\mathbf{s} \in \mathbb{R}^N \rightarrow \mathbf{s}' = (e^{-a\xi_1}, \ldots, e^{-a\xi_N}, 0, \ldots) \in l^p; \tag{3.9}
\]

\[
\mathbf{c} \in \mathbb{R}^N \rightarrow \mathbf{c}' = (1, 1, \ldots, 1, 0, \ldots) \in l^p. \tag{3.10}
\]

The new sequence \( \mathbf{s}' \) gives a strong weight to points closer to the rectangle while marginalizing the influence of points away from the rectangle. The parameter \( a > 0 \) tunes the amount of suppression for points as a function of distance from the rectangle. The even-bounce base points in Figure 3.5b are within approximately 5 pixels of a notional rectangle. If \( a \) were set too large, then the desired points would be suppressed, while setting \( a \) too small would not suppress clutter points. In our
numerical experience, the rectangle is stably recovered for a range of \( a \) values, and we choose \( a = 0.1 \).

To find the best fit rectangle, we optimize the rectangle parameters to minimize the distance between \( s' \) and \( c' \) in \( l^p \); more specifically,

\[
\Lambda = \arg \min_{l,w,t_x,t_y,\gamma} d(s', c').
\]  

(3.11)

In calculating the distance, \( d \), we use \( p = 1 \) for a simple calculation of

\[
d(s', c') = \sum_{k=1}^{\infty} |c'_k - s'_k|.
\]  

(3.12)

Our nonconvex minimization technique uses a particle swarm optimization [3,32,74]. The particles are values of \( \Lambda \) that are initialized randomly around the likely location of the vehicle base. The length and width are set in a region typical of passenger vehicles. The rectangle translation parameters are set around the mean of \( x' \) and \( y' \) locations of the point cloud, while the rectangle rotation is initialized in the direction of the dominant principal component using principal component analysis.

Figure 3.5b and Figure 3.5d show examples computed from even-bounce images, and the rectangles closely match the vehicle bases. Figure 3.5c and Figure 3.5e demonstrate the more difficult problem of finding the ends of a vehicle using only \( HH \) information. The rectangle extraction performs well with even-bounce information on all seven vehicles in the 2006 Collection [4]. However, when using only \( HH \) data, the algorithm missed the end of the vehicle in two of the seven cases.

### 3.3.3 3D location of elevated points

In this section, we describe the extraction of 3D location using the estimated rectangle and an empirically derived inset that accounts for differences between an idealized rectangular solid and actual vehicles.
Figure 3.5: Example points (a) are mapped onto a rectangle. Either even-bounce or HH information is sufficient to find the base of a Hyundai Santa Fe (a midsize SUV) in (b) and (c). In the HH only case, the algorithm has difficulty finding the end of a Nissan Sentra (a small sedan) in (e), while it works well with even bounce in (d).

Figure 3.6 shows a simple model for inverting the layover of an elevated point $E$, typically an odd-bounce roofline reflector. The segment $B_E$ to $R_V$ is a top view of the construction in Figure 3.2b, and Figure 3.6 shows the imaging plane from Figure 3.3 viewed in the $\hat{z}'$ direction. Given a full set of attributed scattering centers like the ones displayed in Figure 2.2b, we convert the points to the $O'$ frame of reference centered at $V$. The multi-bounce returns traced in Figure 3.2a give rise to the rectangle labeled $FGHK$ in Figure 3.6. This rectangle is extracted using the method of Chapter 3.3.2. However, the $(x',y')$ projection, $B_E$, of an elevated reflector, $B$, lies interior to the $FGHK$ rectangle. The rectangle $F_EG_EH_EK_E$ provides a very simple, but effective, model of this inset effect. This physical inset between
Figure 3.6: This construction is a top view of the $x'y'$ plane. The rectangle extraction algorithm finds the polygon $FGHK$ around the base of the vehicle. Additionally, the vehicle’s elevated features are inset from the base and are modeled by $F_EG_EH_EK_E$.

reflecting surfaces for even bounce and layover returns is seen in Figure 3.2a and in the photographs of Figure 3.7. The insets are $X_1$ and $X_2$ for the vehicle sides and ends, respectively; they are empirically derived as shown later in Chapter 3.3.4.

To mitigate clutter, not all points are considered as possible layover points. Because vehicles in the class under consideration are less than 3 meters high, the maximum layover is less than $3 \cot 60^\circ$. Thus, a point $P$ is a candidate layover point if it lies external to the rectangle and in a region $0.1$ to $3 \cot 60^\circ$ meters from the rectangle.

An unknown elevated point $E$ above an unknown $B_E$ in the $O'$ frame of reference is imaged as point $P$ in the direction of the radar $R$, where $R_V$ is $R$ projected onto the $x'y'$ plane. A line from $R_V$ through point $P$ intersects rectangle $FGHK$ at point $B$ and rectangle $F_EG_EH_EK_E$ at $B_E$. At this stage, we have determined the inset rectangle $F_EG_EH_EK_E$ and a point $P$ with $x', y', z' = 0, \theta', \phi', \iota$, and $\nu$ attributes.

Now, it is possible to solve for $E$. Let $\hat{u}$ denote the unit vector in the $-r_v'$ direction, i.e.,

$$\hat{u} = -\left(\cos \theta', \sin \theta', 0\right)^T. \quad (3.13)$$
Also, let $\mathbf{p}' = (x', y', 0)^T$. Then, to find the point $B_E$ beneath the elevated reflector, we have that layover is in the $\hat{u}$ direction. That is,

$$b'_E = \mathbf{p}' + \beta \hat{u}$$

(3.14)

for some $\beta \in \mathbb{R}$. Since we have determined the location of the inset rectangle, we have vectors to its corners. If, for instance, $P$ is on the $F_EG_E$ side of the rectangle, then

$$b'_E = f'_E + \gamma (g'_E - f'_E)$$

(3.15)

for some $\gamma \in \mathbb{R}$, which yields two linear equations in the unknowns $(\beta, \gamma)$. After solving for $b'_E$, the 3D location of the point $E$ is given by

$$e' = b'_E + \hat{z}' \| \mathbf{p}' - b'_E \| \cot \phi'.$$

(3.16)

Finally, we repeat the layover reversal process for each point in the region around the rectangle $FGHK$, thus changing the spatial attributes of the scattering centers from $P$ to $E$.

### 3.3.4 Empirical insets

The inset rectangle $F_EG_EH_EQ_E$ provides a simple improvement in inverting layover and is determined empirically for the class of vehicles under consideration. The 2006 Collection [4] contains a parking lot with seven civilian vehicles that we imaged with 5-degree subapertures and 80 percent overlap of subapertures. The elevation angle $\phi'$ is approximately 45 degrees throughout the parking lot, which is close to the scene center. Point clouds were extracted as described above. We applied the rectangle extraction algorithm in Chapter 3.3.2 to estimate the base outline, $FGHK$, of each vehicle, and then estimated 3D locations of elevated reflectors as in Chapter 3.3.3. The measured height $\hat{h}$ was calculated taking the average of the highest
Table 3.2: Estimated height vs. actual height

<table>
<thead>
<tr>
<th>veh.</th>
<th>$h$</th>
<th>$\hat{h}$</th>
<th>$\tilde{h}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.36</td>
<td>1.38</td>
<td>-0.02</td>
</tr>
<tr>
<td>2</td>
<td>1.41</td>
<td>1.40</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>1.44</td>
<td>1.52</td>
<td>-0.08</td>
</tr>
<tr>
<td>4</td>
<td>1.67</td>
<td>1.55</td>
<td>0.12</td>
</tr>
<tr>
<td>5</td>
<td>1.41</td>
<td>1.43</td>
<td>-0.02</td>
</tr>
<tr>
<td>6</td>
<td>1.47</td>
<td>1.48</td>
<td>-0.01</td>
</tr>
<tr>
<td>7</td>
<td>1.43</td>
<td>1.43</td>
<td>0.00</td>
</tr>
</tbody>
</table>

$\mu_{\tilde{h}} = 0.00$

$\sigma_{\tilde{h}} = 0.06$

20 elevated points on the target. We computed $X_1 = 0.32 \text{ m}$ in order to minimize the mean squared difference between estimated heights and actual vehicle heights taken from Edmunds [75]. Table 3.2 lists the results. The error between the actual and measured vehicle heights is written as $\tilde{h}$ showing the estimate is unbiased with a standard deviation of 6 cm.

Of interest is to also compare the actual vehicle dimensions to the measured base rectangle, $FGHK$, as shown in Table 3.3. Actual dimensions were taken to be the width and length of the smallest bounding box containing the vehicle [75]. The mean and standard deviations for errors are given at the bottom of the table. In the absence of actual height of the front and rear features, we choose $X_2$ by correcting for the mean length estimation error $\mu_l$ with an additional inset of 10 cm to compensate for hood and trunk insets from the edge of the vehicles. As a result $X_2 = 0.02 \text{ m}$. 

43
Table 3.3: Estimated versus actual widths and lengths (in meters)

<table>
<thead>
<tr>
<th>veh.</th>
<th>$w$</th>
<th>$\hat{w}$</th>
<th>$l$</th>
<th>$\hat{l}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.69</td>
<td>1.50</td>
<td>4.42</td>
<td>4.17</td>
</tr>
<tr>
<td>2</td>
<td>1.77</td>
<td>1.82</td>
<td>4.77</td>
<td>4.97</td>
</tr>
<tr>
<td>3</td>
<td>1.71</td>
<td>1.48</td>
<td>4.51</td>
<td>4.10</td>
</tr>
<tr>
<td>4</td>
<td>1.84</td>
<td>2.30</td>
<td>4.50</td>
<td>4.67</td>
</tr>
<tr>
<td>5</td>
<td>1.78</td>
<td>1.92</td>
<td>4.79</td>
<td>5.02</td>
</tr>
<tr>
<td>6</td>
<td>1.85</td>
<td>1.79</td>
<td>5.02</td>
<td>4.84</td>
</tr>
<tr>
<td>7</td>
<td>1.76</td>
<td>1.94</td>
<td>4.84</td>
<td>4.53</td>
</tr>
</tbody>
</table>

$\mu_\hat{w} = 0.05 \quad \mu_\hat{l} = -0.08$

$\sigma_\hat{w} = 0.24 \quad \sigma_\hat{l} = 0.27$

From these empirical observations, we note that the very simple rectangle model gives lengths and widths with small biases and with standard deviations approximately equal to the radar downrange resolution. Thus, the rectangular solid vehicle model, although extremely simple, allows reliable inference of 3D target structure across the entire class of seven vehicles. We believe the model and associated processing provide an excellent trade-off between model complexity and performance.

### 3.4 Experimental Results

We illustrate application of the proposed processing in two examples. First, we extract 3D information from the 2D attributed point sets of the 2006 Collection. Second, demonstrate the invariance provided to shifts in vehicle location across a large scene using the CVDomes data set.
3.4.1 Measured data at scene center

As described in Chapter 2.2.2, the 2006 Collection contains a parking lot with seven civilian vehicles which we imaged with 5-degree subapertures and 80 percent overlap of subapertures. The nominal elevation angle was approximately 45 degrees. Figure 3.7 illustrates the 2D to 3D conversion for the seven vehicles.

A visual inspection reveals that the six sedans have similar yet distinct signatures. The sport utility vehicle (SUV) in Figure 3.7l is clearly wider than the sedans. The SUV also has a prominent elevated feature corresponding to the rear of the cargo hold, as does the station wagon in Figure 3.7t.

The visualizations in Figure 3.7 and in Chapter 3.4.2 below give qualitative assessments of the proposed processing and augment the quantitative results in Tables 3.2 and 3.3. The utility of 3D point clouds is ultimately given a quantitative evaluation by use in a pattern classification algorithm; development of a pattern classification method is beyond the scope of this paper.

3.4.2 Invariance to vehicle location

To generate 2D imagery for an arbitrary vehicle location within a large scene, we used the Toyota Camry (v1) from CVDomess data set of Chapter 2.2.2.

Figure 3.8 provides four circular SAR point clouds generated for a Toyota Camry that is offset from the scene center by various \((q, \theta_v)\) pairs. The first three plots in (a)-(c) show how the Camry would appear when offset from the scene center. (Plot (d) is a reference view of the Camry imaged at the scene center.) The dark points represent the odd-bounce scattering centers. In (e)-(h), we extract 3D information using the technique developed in Chapter 3.3, and the elevated reflectors are dark
Figure 3.7: For seven vehicles (a)-(d) and (m)-(o), a single pass in the 2006 Collection yields point clouds with the 2D scattering center locations shown in (e)-(h) and (p)-(r). Points outside of the base are presumed to be due to layover from elevated reflectors. Parts (i)-(l) and (s)-(u) provide plots of a 3D extraction of the presumed elevated reflectors, where the elevated features have darker voxels.
Figure 3.8: The images in (a)-(d) are top views of SAR point clouds of a Toyota Camry that is offset from scene center by the denoted $(q, \theta_v)$ pair. A 3D extraction algorithm folds suspected elevated points over a rectangle extracted from vehicle base points to generate a 3D image of the vehicle in (e)-(h). Unfolding the 3D point cloud at patch center results in the 2D images in (i)-(l). In the 2D images, odd-bounce reflectors are in the darker shade. In the 3D images, elevated reflectors are in the darker shade.

Due to the assumptions of our experiment, $\phi'$ is a maximum when the azimuth $\theta$ is the same as $\theta_v$; thus each of Figures 3.8 (a)-(c) shows a bulge of the outer ring in the direction of $\theta_v$ when compared to Figure 3.8(d). The 3D extractions in Figures 3.8(e)-(h) look visually similar for all four views. The re-created 2D views
Figure 3.9: Toyota Tacoma pickup truck: (a) 2D image peaks; (b) 3D point cloud after inversion of conjectured layover points.

at scene center in Figures 3.8(i)-(l) are provided for comparison purposes. The views look similar; however, there are also differences. In particular, the interior points may have differences due to the limited persistence of scattering centers over changes in elevation. As the radar orbits around the vehicle, the imaging elevation varies with azimuth differently for each view.

3.4.3 Pickup truck example

For a final example, Figure 3.9 provides 2D and extracted 3D views of a Toyota Tacoma pickup truck. Images were formed from simulated far-field scattering for the Tacoma at scene center with elevation angle $\phi' = 45^\circ$. Compared to sedan and SUV type vehicles, a pickup truck has the added complexity of the raised dihedral base in the bed of the truck. The dense cluster of even-bounce points shown in the right side of Figure 3.9a are inside the base rectangle and are not considered as potential elevated points for the 3D extraction of Figure 3.9. However, as with the other vehicle types, we are able to extract 3D information for the roofline of the cab.
3.5 Height Estimation Uncertainty

The height of elevated reflectors was estimated in Chapter 3.3 using a single very simple physical model for all vehicles. Based on the seven vehicles in the 2006 Collection, we postulated the medial inset $X_1$. In this section, we characterize the effect on height estimates of model error $D_1$ in the selected inset value. The resulting height estimation error variance may be incorporated in a point matching algorithm that uses the proposed point cubes for target recognition. Here, we characterize the error of height estimates for elevated points as a function of $q$, the ratio of vehicle distance from scene center and flight path radius. To proceed, we assume that $D_1$ is a normally distributed random variable with the mean and variance reported in Table 3.2.

From (3.16), the estimated height of an elevated reflector is given by

$$e_z' = \|p - b_E\| \cot \phi'.$$

In the context of Chapter 3.4.2, let $\theta = \theta_V$ for maximum layover, and (2.5) simplifies to $\cot \phi' = (1 - q)$.

Note that for this analysis, $q \in (-1, 1)$. Allowing $q < 0$ moves the vehicle through scene center and away from the radar. Conversely, for $q > 0$, the vehicle is closer to the radar than to the scene center. As a result, we have

$$e_z' \approx (l + D_1)(1 - q).$$

where $l = \|p - b_e\|$ is the measured layover. Accordingly, the height estimation error, in meters, has mean

$$\mu = \mu_{D_1}(1 - q)$$

(3.19)
and variance

\[ \sigma^2 = \sigma_D^2 (1 - q)^2. \]  

(3.20)

Thus, from the empirical values in Table 3.2 for seven vehicles in the 2006 Collection, the estimated heights are unbiased for any \( q \) and have error variance

\[ \sigma^2 = 0.004(1 - q)^2. \]  

(3.21)

Intuitively, when the vehicle is close to the radar, \( q > 0 \) and the layover is larger with respect to the unknown base and inset, which reduces the height estimation error variance. On the other hand, when the vehicle is farther away from the radar, the variance is increased due to a small layover. Thus when the elevation angle is small, it is more difficult to estimate the reflector height. From (3.16) and the lateral symmetry of vehicles, one could obtain a refined estimate of vehicle height, if desired, using a weighted average of estimates from each side; the unequal variances provide the inverse weights.

### 3.6 Discussion

A set of attributed scattering centers is a compact and informative representation for storing 360 degrees of high-frequency SAR imagery. In this chapter we proposed a method for extracting 3D information from a single pass; layover in a 2D image was inverted with the aid of polarization. The 3D representations provided an invariance to the signature variation due to elevation angle changes across a large scene. The proposed processing was demonstrated on measured radar returns.

For single-pass 3D imaging, Chapter 3.3.2 describes the extraction of a rectangular base from a 360-degree view. For expanded capability, the feature extraction
algorithm may be modified to extract an “L” or a line segment when there is limited viewing angle on the target. For example, from a narrow aperture that contains a broadside view like in Figure 2.1b, we could search for a line segment that fits the even-bounce base.
CHAPTER 4: Classifying Vehicles with Hausdorff Distances

The material for this chapter is based on the manuscript: K. E. Dungan and L. C. Potter, “Classifying transformation-variant attributed point patterns,” Pattern Recognition, to appear.

4.1 Introduction

Suppose that a class sample is represented by a collection of feature vectors consisting of transformation-variant features along with other attributes. For example, a fingerprint can be represented by a set of minutiae, each containing two-dimensional (2D) spatial location along with type and orientation attributes [76]. The collection of feature vectors is referred to as an attributed point pattern or point set. The locations are subject to an unknown rigid transformation. When comparing two different samples, it may be necessary to register the location information before comparing the two point sets.

In this chapter we estimate the registration of sets using a Hausdorff distance based technique. Registration and set distance are related; the estimated registration between two sets yields the minimum distance between the two sets. Thus we refer to the distance between two sets under their estimated registration simply as the set distance (SD). Partial versions of the Hausdorff distance [17, 77, 78] are naturally robust to clutter and occlusions since they find a best subset match between two sets.
In previous work, Yin [3] demonstrated a method to register two 2D point patterns by minimizing a partial Hausdorff distance between the two patterns with a particle swarm optimization. The Yin article registered synthetically generated 2D point patterns perturbed by rigid transformations, random clutter, and random occlusions. We extend Yin’s contribution with four meaningful steps:

- In addition to 2D location, we included attributes that added more information to the point pattern. The associated distance between individual points can be characterized with a Mahalanobis distance using an appropriately selected error covariance matrix.

- In addition to registering point patterns, we use a version of the minimized partial Hausdorff distance as a pseudo-metric for a nearest neighbor (NN) classifier. Augmented distance matrices created from multiple samples are observed to be nearly positive definite, which reveals that the pseudo-metric well approximates a valid a distance measure.

- We demonstrate the first classification results for persistent radar surveillance; prior art has been restricted to narrow apertures. Further, we give the first published results for classification of civilian sedans, which present very small and very similar radar signatures, in comparison to military vehicles. To illustrate the generic applicability of attributed point patterns, we also present a small example for latent fingerprints.

- For applications with multiple training samples, we describe multidimensional scaling (MDS) and Landmark MDS (LMDS) [79] as tools for visualization, analysis, and alternative classifiers.
Figure 4.1: Calculate the set distance between the query $Q$ and each class sample $H_i$; the nearest neighbor classification is the shortest distance. An optional chain of processing for analysis/visualization or alternative classifiers using MDS/LMDS is indicated with the dashed boxes. Offline processing is shown in green.

Advances in processing and high performance computing have made it possible to tractably solve optimizations necessary for the registration and classification in a multiclass problem [32, 33, 42]. Figure 4.1 describes the proposed classification algorithm. Given a database of training images, we extract a set of transformation-variant feature vectors (attributed points) for each image; the database is represented as a set of sets $H = \{H_1, H_2, \cdots, H_i, \cdots\}$. Then, given a query image, we extract a set $Q$ of attributed points and calculate a set distance $d_i$ from the query to each set in the database. Using a nearest neighbor test, the classification $C$ is determined from the minimum set distance between $Q$ and each $H_i$. Thus,

$$C = \arg\min_i d(Q, H_i).$$

(4.1)

Figure 4.1 shows several application specific parameters used in the set distance calculation. The distances between attributed point patterns are minimized under a transformation $T$. The measurement error covariance matrix $\Sigma$ is used to calculate a Mahalanobis distance between individual feature vectors. Based on the estimated level of clutter in a query, the parameter $K$ is set to facilitate the best subset match.
Finally, the particle swarm optimization (PSO) is run for a specified number of particles and iterations as determined by training.

Notice that green boxes in Figure 4.1 are performed offline, while dashed boxes are part of the optional MDS/LMDS chain. If multiple training samples are available, it is possible to generate a matrix of distances $d_{ij}$ between patterns in the union of classes. By applying multidimensional scaling (MDS), the samples are represented in a Euclidean space, $X_i \in \mathbb{R}^n$, for a visualization of class separation and an analysis of the pseudo-metric. Given the points in the Euclidean space, it is possible to train classifiers other than NN, such as a support vector machine (SVM) or a linear discriminant analysis (LDA) classifier. When a measured query sample $Q$ is available, we can map the sample into the Euclidean space using a landmark MDS algorithm (LMDS) [79] for visualization or classification.

The remaining sections are organized as follows. Chapter 4.2 describes Figure 4.1 with subsections detailing the set distance, MDS analysis/training, and LMDS visualization/classification. In Chapters 4.3 and 4.4 we use the proposed approach in two applications: latent fingerprint classification and circular synthetic aperture radar (SAR). Chapter 4.5 provides a summary and discussion of results.

4.2 Set Classification and Analysis

This section develops a version of the Hausdorff distance that is minimized over a set of transformations. Since the distances and samples are not generated in a familiar Euclidean Space, the MDS/LMDS sections provide a means for analysis and visualization.
4.2.1 Sets Distances using the Hausdorff Metric

The Hausdorff distance (HD) is a well-known method for representing the distance between two point sets without having a prior correspondence between the two sets. Huttenlocher et al. [17] applied the classical Hausdorff measure [19] concept to matching point sets. Essentially, the HD is the distance of the most isolated point between \( Q \) and \( H_i \). However, an outlier or occlusion could skew an otherwise close registration, in which case, the partial Hausdorff distance (PHD) [17], which is the \( K \)th minimum distance between points in the sets, may be used. We apply a more robust form of the PHD called the least trimmed square Hausdorff distance (LTS-HD), which takes the mean of the \( K \) minimum distances between point sets [80]. The directed LTS-HD may be written

\[
h_K(Q, H_i) = \frac{1}{K} \sum_{q_1 \ldots K \in Q} \min_{h_k \in H_i} ||q_j - h_k||.
\] (4.2)

The point sets \( Q \) and \( H_i \) are not necessarily registered prior to calculating the LTS-HD. Rucklidge [81] investigated minimizing the PHD over rigid transformations with scaling. We build upon this concept by using the LTS-HD and generalizing the underlying norm with the Mahalanobis distance [82]. The resulting set distance is defined by

\[
d_i = \min_T h_K(T(Q), H_i),
\] (4.3)

where \( T \) is the set transformation. In our application, \( T \) defines rigid transformations in a 2D plane; however, \( T \) is flexible to fit the desired application such as scaling, shifts in time, or 3D transformations.

The norm in (4.2) is calculated from the Mahalanobis distance

\[
||q - h|| = \sqrt{(q - h)^T \Sigma^{-1}(q - h)},
\] (4.4)
where $\Sigma$ is the measurement error covariance matrix for the vectors in a point set. The Mahalanobis distance facilitates the comparison of vectors, where the various dimensions have different scales, different error sources, or correlated errors. For example, in radar, some attributes may contain spatial locations of bright reflectors, while other attributes contain information about intensity, polarization, or direction of illumination. Use of an appropriate error covariance matrix increases class separability. In practice, it is typical to estimate the measurement error variances of each feature in the feature vectors to populate the diagonal of $\Sigma$; however, determining the off-diagonal covariance terms may improve results.

Mount et al. [83] suggest setting $K$ in (4.2) as a fraction of the cardinality of one of the sets; thus in our application,

$$K = \left\lfloor r \min\{|Q|, |H_i|\} \right\rfloor, \quad 0 < r \leq 1. \quad (4.5)$$

For example, a value of $r = 0.6$ would mean $d_i$ is minimized over a subset of 60% of the points in $Q$, supposing $|Q| < |H_i|$. Hence, the pattern match would be immune to a number of spurious or occluded points. In applications, the designer should estimate the average clutter ratio in query data and set $r$ a little lower.

To calculate the set distance of (4.3), we minimize the LTS-HD over the set of allowable transformations using a nonconvex technique called a particle swarm optimization (PSO). The PSO as a generic nonconvex optimization tool is explored in detail by Clerc and Kennedy [74], with emphasis on convergence to local versus global minima. In a precursor to our application, Yin [3] applies a PSO to find the global minimum PHD over rigid transformations with scaling between two 2D point sets.
To search the 2D affine space with scaling, the set transformation is defined by

\[ T(q) = sRq + x, \quad \forall q \in Q. \]  

(4.6)

Thus, the PSO searches for an angle in the rotation matrix \( R \), a scaling factor \( s \), and a translation vector \( x = (x_1, x_2)^T \) for an optimization over four variables. Due to the physics of our applications, we do not allow scaling.

The PSO algorithm requires initializing a number of particles (i.e. \( x \) and \( y \) translations and \( \theta \) rotation) and running the algorithm for a number of iterations; the initializations are randomly set within application-specific bounds. The particles then move as the PSO algorithm converges to a local minimum \([3, 74]\). As with any non-convex optimization problem, it is possible to converge upon a local minimum that is not the global minimum. In general, more particles and iterations implies a greater chance of finding the global minimum at the expense of processing those extra particles. Efficient selection of stopping criteria for PSO is an open problem explored by Kwok et al. \([84]\). However, a sufficient number of particles and iterations can be empirically set using training data.

When minimizing an LTS-HD over a set of transformations, it is possible to find the same point pattern in multiple places within the same point set. Hence, the resulting set distance could violate the metric properties of non-degeneracy and triangle inequality; the directed LTS-HD may fail under symmetry as well. However, as the diversity of information and point set sizes increase, it is less likely to find repeated patterns leading to metric violations. The next section describes multidimensional scaling as applied to set distances, and the eigenvalues that are generated offer evidence to the suitability of the set distance as a metric. Use of the pseudo metric is further justified by strong classification performance in practice.
In summary, the set distance (4.3) requires setting $T$, $K$, $\Sigma$, and the PSO parameters; all of these are deterministic for the desired application. The set distance design is robust to an unlimited number of applications, and for two of these, fingerprints and SAR, we demonstrate specific design decisions for the four parameters.

4.2.2 MDS for Analysis and/or Training

Because a sample is defined as an unordered set of transformation-variant vectors, it cannot be plotted as a single point in an inner product space. Furthermore, it is not possible to visualize a comparison of point patterns since they do not exist in a Euclidean space. However, given a collection of samples and a matrix of set distances between those samples, it is possible to approximately represent the collection of samples in a Euclidean space using multidimensional scaling (MDS). The representation of point patterns as a single point in a low-dimensional Euclidean space permits visualization of data and allows for the use of classifiers other than NN. Additionally, for analysis purposes, the eigenvalues generated during the MDS process provide evidence showing how close the pseudo-metric is to a true metric.

We apply MDS as described by Potter and Chiang [85]; given a set of samples, a Euclidean distance matrix (EDM) stores all pairwise distances between samples. If the distance measure follows the rules of a metric space, then the EDM is symmetric with zeros on the diagonal. By squaring the distances in the EDM, we define an $m$-by-$m$ matrix $D$. Schoenberg [86] proved the following theorem:

**Theorem 1.** Let $e \in \mathbb{R}^m$ be given by $e = [1, 1, \cdots, 1]^T$, and let $\Phi$ be the orthogonal projection matrix onto the subspace $M = \{x \in \mathbb{R}^m : x^T e = 0\}$, i.e.,

$$\Phi = I - \frac{1}{n}ee^T.$$  \hspace{1cm} (4.7)
A matrix $D$ has representation by pairwise distances of $m$ points in $\mathbb{R}^{m-1}$ if and only if $\Phi D \Phi^T$ is a positive-semidefinite matrix. Further, for the matrix $D$ to admit representation in the embedding space $\mathbb{R}^n$ for $n < m - 1$ requires $\text{rank}(\Phi D \Phi^T) \leq n$.

Finally, we solve the eigen decomposition $\Phi D \Phi^T = U \Lambda U^T$, and the rows of $U \Lambda^{1/2}$ are the Euclidean embedding of the point set into an $n$-dimensional space.

Suppose that we have multiple samples for two different classes $H_{i*}$ and $H_{j*}$ where “*” represents a wildcard for all of the samples within a given class. First, we construct a matrix of squared distances between each sample. For example, with four training samples from two classes, $\Psi_{ij} = \{H_{i1}, H_{i2}, H_{j1}, H_{j2}\}$, we generate a matrix of squared set distances with zeros on the diagonal

$$D_{ij} = \begin{bmatrix}
0 & d_{12}^2 & d_{13}^2 & d_{14}^2 \\
 d_{12}^2 & 0 & d_{23}^2 & d_{24}^2 \\
d_{13}^2 & d_{23}^2 & 0 & d_{34}^2 \\
d_{14}^2 & d_{24}^2 & d_{34}^2 & 0
\end{bmatrix}. \quad (4.8)$$

The subscripts of the matrix entries correspond to the position in $\Psi_{ij}$; for example,

$$d_{13} = \min_T h_K(T(H_{i1}), H_{j1}). \quad (4.9)$$

If $|\Psi_{ij}| = m$, then calculating $D_{ij}$ requires $m(m - 1)/2$ distance calculations. Upon eigen decomposition of

$$\Phi D_{ij} \Phi^T = U_{ij} \Lambda_{ij} U_{ij}^T, \quad (4.10)$$

we find the embedding

$$X_{ij} = \Lambda_{ij}^{1/2} U_{ij}^T \quad (4.11)$$

and design a classifier to choose between the classes $i$ and $j$. With the eigenvalues in $\Lambda_{ij}$ in descending order from maximum magnitude, truncating the first $n$ columns of $X_{ij}$ provides an embedding in $\mathbb{R}^n$. 
If our defined set distance (i.e. minimized LTS-HD) were a proper metric, then \( \Phi D \Phi^T \) would always be positive-semidefinite. However, noisy measurements, occasional violation of the triangle inequality, or occasional failure to find a global minimum in (4.3) will cause some negative eigenvalues. Qualitatively, a few small negative eigenvalues relative to several large positive eigenvalues lends evidence that the pseudo-metric may be useful in the application.

4.2.3 LMDS for Visualization and/or Classification

As described in Chapter 4.2.2, a collection of training samples \( H_i \in H \) can be embedded in a low-dimensional Euclidean space for analysis. Given a new query \( Q \), it is possible to add a row and column to the matrix \( D_{ij} \) with distance measurements to the new sample \( Q \) and apply MDS to embed the measurement,

\[
\{ H_{i_1}, H_{i_2}, \ldots, H_{j_1}, H_{j_2}, \ldots, Q \} \xrightarrow{\text{MDS}} \{ h_{i_1}, h_{i_2}, \ldots, h_{j_1}, h_{j_2}, \ldots, q \} \subset \mathbb{R}^n.
\]  

(4.12)  

(4.13)

Then, with the training samples in \( \{ h_{i_1}, h_{i_2}, \ldots, h_{j_1}, h_{j_2} \} \subset \mathbb{R}^n \), one could design a classifier and classify \( q \).

However, each new query would require a new embedding and new training. Furthermore, to visualize multiple queries in the same Euclidean embedding would require expanding the EDM with extra rows and columns for the additional queries along with the distances between the queries themselves. Fortunately, a technique called landmark MDS (LMDS) [79] provides a method to add query samples to a Euclidean space based on distances to training samples that were previously embedded using MDS. Thus all training and query samples can be visualized together, and a
classifier need be trained only once. Using a common embedding, however, may cause a suboptimal embedding for individual measurements.

To implement LMDS, Silva [79] describes a method to invert the eigen decomposition of the embedding and apply the result to the landmarks. Let \( X_t \) be the pseudo-inverse \( (\Lambda^{1/2}U_i^T)^+ \) truncated to the dominant \( n \) eigenvalues (i.e. the embedded dimension). Thus, adding a new measurement \( x_q \) to an existing embedding of the training samples is

\[
x_q = -\frac{1}{2} X_t (\delta_q - \delta_\mu),
\]

where \( \delta_q \) is the vector of squared distances to the landmarks and \( \delta_\mu \) is the mean vector of squared distances from each landmark to all of the other landmarks.

To implement a multiclass classifier using MDS/LMDS in our SAR example, we chose to use a binary decision tree. During offline training, we apply MDS to the \( N \) choose 2 pairings of the training classes. Then, in the decision process, we apply LMDS to add the query samples to the low-dimension embeddings. For \( M \) classes, the decision tree requires \( M-1 \) binary decisions.

### 4.3 Latent Fingerprint Application

To demonstrate the breadth of the algorithm, we provide a section on classifying latent fingerprints prior to the SAR vehicle identification section. Fingerprints of varying quality that are incidentally left and extracted from a scene are referred to as latent, while a tenprint refers to fingerprints that are collected and recorded in a controlled setting onto fingerprint cards [87]. In this section, we apply set distances to classify latent fingerprints to a database of tenprints.
Figure 4.2: The image in (a) is a latent fingerprint with overlaid minutiae, and (b) is a tenprint image with overlaid minutia. The plot in (c) shows the registration of the latent and tenprint location information, while (d) adds the orientation field information.

4.3.1 Classifying Latent Fingerprints

The National Institute of Standards and Technology (NIST) special database 27 (SD27) contains 258 latent fingerprint images, which are subdivided into 88 good, 85 bad, and 85 ugly images [87]. SD27 also contains 258 tenprint images corresponding to the latent images. It is common to extract key points on a fingerprint such as ridge endings or bifurcations that are collectively called fingerprint minutiae. For both the latent and tenprint images, SD27 provides sets of minutiae with location, orientation, type, and quality attributes. Minutiae in the latent fingerprints of SD27 typically represent a small subset of the minutiae in the tenprint. Some of the latent fingerprints have fewer than ten minutiae, while the tenprint images typically have on the order of 100 minutiae.

We independently classified each set of latent minutiae to the 258 tenprint classes using the NN classifier path of Figure 4.1, where Figure 4.2 summarizes the registration and set distance process in comparing a latent image to a tenprint. Figure 4.2(a)
offers an example latent fingerprint image with minutiae locations indicated by blue circles, where the small line in each blue circle indicates the direction of the minutiae orientation attribute. Next, the image in Figure 4.2(b) shows the corresponding tenprint image with minutiae shown in red. By minimizing the Hausdorff distance over \( T \) as in (4.3), Figure 4.2(c) and (d) show the latent minutiae in blue registered to the tenprint minutiae in red. In (c), the view of the \( x \) and \( y \) location attributes easily relates back to the minutiae in (a) and (b), while the plot in (d) provides a visualization of the orientation attribute on the \( z \)-axis. The orientation attribute increases the diversity of information, making it less likely to find the same point pattern in a different part of the tenprint.

### 4.3.2 Feature Vector and Covariance

The minutiae type and quality attributes are undetermined for the tenprint database entries of SD27. Thus, we consider only the location and orientation attributes of the minutiae, and the transformation in calculating the set distance is given by

\[
T_i \left( \begin{bmatrix} q_{ix} \\ q_{iy} \\ q_{i\theta} \end{bmatrix} \right) = \left[ \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix} \right] \left( \begin{bmatrix} q_{ix} \\ q_{iy} \end{bmatrix} \right) + \left[ \begin{bmatrix} t_x \\ t_y \end{bmatrix} \right] + \left( \text{mod}(q_{i\theta} + \psi, 2\pi) \right) \right] .
\]  

(4.15)

In calculating the LTS-HD of (4.2), the norm is calculated as a Mahalanobis distance in (4.4). To select the covariance matrix \( \Sigma \) of measurement errors, we populated the diagonal elements with location variances \( \sigma_x^2 \) and \( \sigma_y^2 \) and orientation variance \( \sigma_\theta^2 \). We first selected variances based on an estimate, then trained on three latent/tenprint pairs to maximize the inter class separations in the LTS-HD. To start, we assumed 95% confidence that the location information is within two pixels of the true locations. Thus,

\[
2 = 1.98\sigma_x \Rightarrow \sigma_x^2 = \sigma_y^2 \approx 1, \quad (4.16)
\]
where 1.98 is the z-score associated with 95% of the area under a normal distribution. Using the same logic, we assumed that the orientation attribute is measured accurately within 5° with 95% confidence. Thus,

\[ 5 = 1.98 \sigma_{\theta} \Rightarrow \sigma_{\theta}^2 = 6.38. \] (4.17)

After training on the three pairs, we determined that these initial estimates worked well.

Next, defining the LTS-HD requires setting \( K \) using \( r \) in (4.5). Intuitively, a smaller \( K \) increases the chance of finding multiple pattern matches in the tenprint images (i.e. \( K = 1 \) would allow a zero set distance at every point in the tenprint). By increasing \( K \), it is less likely to violate the nondegeneracy property of distances. However, if \( K \) is too large, then false minutiae could be matched to the tenprint. Based on training with three images (identical to that used for determining \( \Sigma \)), we set \( r = 0.75 \) to ignore a quarter of the latent minutiae.

### 4.3.3 Latent Fingerprint Classification Results

Classification results are displayed in Figure 4.3, which displays the probability that the correct tenprint appears among the \( N = \) rank minimum set distances. For example, a \( P_{id} \) with rank 10 means that an image is correctly classified in the top 10 selections with a probability \( P_{id} \). Results are displayed for good, bad, and ugly images, where rank-1 \( P_{id} \) is 0.89, 0.64, and 0.60 respectively. The rank-20 accuracy is 0.93, 0.89, and 0.81. When combining all three image types, the rank-1 \( P_{id} = 0.71 \) and the Rank-20 \( P_{id} = 0.88 \).

To demonstrate the effect of the measurement error covariance matrix \( \Sigma \), we classified the good latent fingerprint data using a unity covariance matrix. The effect
Figure 4.3: Higher quality latent imagery results in higher probability of identification. The black line shows that good latent images are identified in the top five or so tenprint images around 90% of the time. Bad images are shown in red with ugly in blue.

of changing $\sigma^2_\theta$ from 2.5 to 1 is to add more importance to the minutiae orientation attribute since $\Sigma$ is inverted in the Mahalanobis distance. As shown in Figure 4.3 with the black dashed line, results are slightly degraded with a rank-1 performance of 0.85. On the other hand, by increasing $\sigma^2_\theta$ to a relatively large number such as 1000, the orientation attribute has little importance in the distance measurements due to inverse of $\Sigma$ in (4.4). In doing so, the rank-1 performance dropped to 0.19, which indicates that the information provided by the orientation attribute is significant.

### 4.3.4 Time complexity

The time necessary to classify a latent fingerprint using the NN path of the algorithm in Figure 4.1 is dominated by the set distance calculation. The particles in PSO are searching for the registration that minimizes the LTS-HD measure. A typical swarm of 300 particles searching a tenprint for 150 iterations requires approximately
Figure 4.4: Typical attributed point patterns in the latent fingerprint experiment contained 20 points in the latent image and 100 points in the tenprint. Calculating the minimized LTS-HD takes approximately 5 s.

5 seconds to settle on a global minima as illustrated in Figure 4.4 using an Intel Core i7-920. A query of the 258 tenprints of the SD27 database required approximately $258 \times 5 = 1290$ s; however, we accomplished the query in 5 seconds using 258 cores of the Ohio Supercomputer Center’s Opteron Cluster in parallel.

4.3.5 Comparison to Prior Work

Matching partial fingerprints is a challenging problem with a significant degradation in performance as partial match area approaches 10% [88]. Few studies have been published using SD27. Recently, Feng and Jain [89] reported an accuracy of 73.3% against a background database containing 10248 images. Jain et al. [29] reported an accuracy of 79.5% with SD27 against a background database containing 2258 images; their method ranks the correct tenprint among the top 20 closest images in 93.4% of cases.
Our results in Chapter 4.3.3 reported rank-1 and rank-20 classification performances of 71% and 88%, respectively; also, the background database was smaller, containing only the 258 tenprints in the SD27 database. Clearly, the two prior studies [29, 89] showed superior results. However, the prior studies were finely tuned to fingerprint matching and used unspecified algorithms to extract additional minutiae attributes. As shown in Chapter 4.3.3, we achieved a dramatic improvement in classification performance by adding the minutiae direction field to the location attributes. Including additional minutiae features to our attributed point patterns, such as those used in the Jain and Feng studies, may be a promising future study.

4.4 Vehicles in Circular Synthetic Aperture Radar

The purpose of this section is to test the hypothesis that X-band radar with only 640 MHz bandwidth (23 cm resolution), operating in circular SAR mode, can effectively classify civilian passenger vehicles. The classifier must overcome the spatially variant behavior of radar signatures illustrated in Figure 2.4 and Figure 2.5. Numerous previous studies have considered large military vehicles [20], while our presentations [30,32,33,42] have addressed passenger vehicles.

The next few sections analyze the CVDomes experiment of Chapter 2.2.1, followed by several interesting results from the 2006 Collection of Chapter 2.2.2. The applications use a subset of the information from (2.1), where each point in an image query is represented by

\[ q = (x, y, \theta, \iota, \nu). \]  (4.18)

A vehicle sample with \( N_q \) attributed points is represented by a set

\[ Q = \{ q_1, \ldots, q_{N_q} \}. \]  (4.19)
Likewise, the database samples are referred by the $H_i$.

### 4.4.1 Set Distance Parameters

To measure the set distances, we apply the LTS-HD based set distance as described in (4.3). This requires defining a transformation $T$, a covariance matrix $\Sigma$, and a subset cardinality $K$. First, the transformation is defined by

$$
T_i\left(
\begin{bmatrix}
q_{ix} \\
q_{iy} \\
q_{ia} \\
q_{iv}
\end{bmatrix}
\right) = 
\begin{bmatrix}
\cos \psi & -\sin \psi \\
\sin \psi & \cos \psi \\
\end{bmatrix}
\begin{bmatrix}
q_{ix} \\
q_{iy}
\end{bmatrix} + 
\begin{bmatrix}
t_x \\
t_y
\end{bmatrix} + 
\begin{bmatrix}
q_{ia} \\
q_{iv}
\end{bmatrix}
\mod(q_{i\theta} + \psi, 2\pi).
$$

(4.20)

In SAR, the image scale is known and is independent of range. Thus, it is only necessary to search over rigid transformations with the translations $t_x$ and $t_y$ and the pose rotation $\psi$. The azimuth attribute $\theta$ is transformed directly with the pose rotation, while the amplitude $A$ and even/odd-bounce bit $\nu$ are not transformed.

To design $\Sigma$ in the Mahalanobis distance, we chose to populate only the main diagonal for simplicity. Our strategy begins by creating rough estimates for the measurement error variances of the five attributes. Suppose that we are 95% confident that our scattering center peak locations $x$ and $y$ are within two pixels of their true locations (i.e. within a 12 cm). Thus,

$$
2 = 1.98\sigma_x \Rightarrow \sigma_x^2 = \sigma_y^2 \approx 1.
$$

(4.21)

Now, suppose that the azimuth attribute for each peak is uniformly distributed over the 5 degree aperture; hence

$$
\sigma_\theta^2 = \frac{(b - a)^2}{12} = \frac{5^2}{12} \approx 2.
$$

(4.22)

For the amplitude attribute, log-normal and quarter-power normal models have been widely adopted for non-clutter peaks [90]. Under the log-normal assumption, suppose
that with 95% confidence, the amplitude measurement is accurate within 5 dB (zero dB is the maximum peak in the data set); thus

\[ 5 = 1.98\sigma, \Rightarrow \sigma^2 \approx 2.5^2 = 6.25. \]  

(4.23)

Lastly, suppose that the even/odd bounce bit \( \nu \) is correct with probability \( \phi = 0.95 \); thus

\[ \sigma^2_{\nu} = \phi(1 - \phi) = .95(1 - .95) = 0.05. \]  

(4.24)

Next, we selected the training samples for two similar vehicles, v1 and v5, to test class separation based on the estimated \( \Sigma \). Then, we applied MDS in \( \mathbb{R}^3 \) to visualize the class separation. By individually tuning the five variance terms, it was possible to slightly improve class separation; however, the initial estimates were close to the tuned selections.

The last parameter is the subset cardinality parameter \( K \). Using the method in equation (4.5), we selected \( r = 0.7 \), applied to the smallest set of points in the training set. This allowed for the set distance to ignore at least 30% of the points in calculating the LTS-HD between two point patterns.

4.4.2 CVDomes Classification Results

Using the experiment described in Chapter 2.2.1 with set distance parameters described in Chapter 4.4.1, we applied the NN classifier of Figure 4.1, which finds the minimum set distance from each query to the 60 training samples. Table 4.1(a) shows the classification result as a confusion matrix. Each row of the matrix represents the actual vehicle in the 440 query samples, while the columns represent the classification results. An ideal confusion matrix would contain all 44s on the diagonal and zeros elsewhere. Dividing the trace of the confusion matrix by 440 quantifies performance
as a probability of correct identification, $P_{id}$. The result is a classification performance of $P_{id} = 0.977$. With only one training sample circled at the center of Figure 2.5, as shown in Table 4.1(b), the performance drops to $P_{id} = 0.918$.

Next, we investigated visualization, analysis, and alternative classification using an MDS/LMDS embedding, starting with a visualization. Some pairs of vehicle classes were easily separated in an $\mathbb{R}^2$ embedding. Figure 4.5 shows an example MDS/LMDS embedding of the Sentra (v8) and the Avalon (v9); notice that these vehicles were also used in the example imagery of Figure 2.4. Applying the minimized LTS-HD to all of the pairings of the six training samples for each class generated a $12 \times 12$ EDM of set distances. Next, by solving the eigen decomposition from equation (4.11), the red pluses in Figure 4.5 (a) and (b) along with the green asterisks represent the MDS embedding for the Sentra and Avalon, respectively. Based on the six training samples, now in a Euclidean space, a support vector machine with a linear basis easily separates the classes as shown with a line using the circled support vectors. Using LMDS from Chapter 4.2.3, Figure 4.5(a) shows the embedding and proper classification of the 44 Sentry samples with pink pluses. Likewise, Figure 4.5(b) shows the embedding and proper classification of the 44 Avalon samples with blue asterisks.

Based on the example in Figure 4.5, the LTS-HD, although not a true distance metric, was able to separate two classes in an $\mathbb{R}^2$ embedding with perfect classification results. In a more general sense, we can look at the eigenvalues of the Schoenberg augmented EDMs to see how well the selected set distance adheres to the triangle inequality. Figure 4.6 describes the eigenvalues for the SAR application. The plot in Figure 4.6(a) shows the eigenvalues for the v1 to v3 comparison, which had the best set of eigenvalues for all pairings in the dataset. It has only one small negative
Table 4.1: These confusion matrices show actual vs. classified results for ten vehicles using a NN classifier; (a) has 6 training samples per class, while (b) uses only one training sample per class.
Figure 4.5: MDS enables visualization of the point patterns as vectors in $\mathbb{R}^2$. The plot in (a) shows v8 samples properly classified, while (b) shows v9 samples properly classified.
Figure 4.6: Eigenvalues for the Schoenberg augmented Euclidean distance matrices are displayed. (a) and (b) show representative examples, while (c) and (d) show the average trends for the dataset. There are several large positive eigenvalues and the few negative ones are small.
eigenvalue, and the embedding is nearly perfect in $\mathbb{R}^9$. One of the worst pairings is the pattern shown in Figure 4.6(b) for v4 to v8, where the first seven dominant eigenvalues are positive. In general, for the entire set of data on average, as shown in Figures 4.6(c) and (d), most of the information is stored in the dominant eigenvalues, and few small eigenvalues are negative.

Next, we applied MDS to the 45 vehicle pairings in the 10 class problem. For each pairing, we investigated the performance of a binary decision tree using nearest neighbor, LDA, linear SVM, and radial basis function (RBF) SVM classifiers. Each decision was similar to the plots described in Figure 4.5. By using LMDS, each of the 440 classification samples is embedded and classified using nine binary comparisons in the classification tree. For example, a sample is classified in each of the pairings (v1,v2),(v3,v4),(v5,v6),(v7,v8), and (v9,v10), thus leaving five winners. Subsequently, we continue classifying pairs until a winner is selected. The ten class problem requires four levels of decisions (10, 5, 3, 2) for 9 total decisions (5+2+1+1).

Figure 4.7 displays the results for a variety of classifiers and embedding dimensions. There were at least seven positive eigenvalues for each MDS embedding of the training samples; thus, we tested the classifiers in each of the first seven dimensions. The RBF-SVM in $\mathbb{R}^4$ was the best performing classifier with $P_{id} = 0.941$, as highlighted with the confusion matrix in Table 4.2.

We note two observations from the confusion matrices. First, in the NN classifier of Table 4.1(a), most of the classification errors were caused when distinguishing the two models of Jeep in v3 and v4. Second, the SVM classifier in $\mathbb{R}^4$ resolves confusion between the Jeep models, but causes several vehicles to be misclassified as a Civic, v2.
Figure 4.7: The probability of correct classification for several classifiers is displayed versus embedding dimension. Generally, performance peaks around $\mathbb{R}^4$ with similar performance for all four classifiers.

These results show that a LTS-HD minimized over rigid transformations can classify a 10 class circular SAR application in a large scene with greater than 97% accuracy using six scattered templates. Even using a single training sample, the performance is greater than 91%. The MDS/LMDS processing did not improve performance in this case; however, it did provide a tool for visualizing samples for class separation and analyzing the minimized LTS-HD via the eigenvalues generated during the embedding process.

4.4.3 Time Complexity

The time required to classify a query vehicle from the circular SAR experiment is dominated by the time required to solve the nonconvex PSO optimization of the LTS-HD. Figure 4.8 shows the progression of a typical minimization between a pair of 200 point patterns. Although, efficient selection of stopping criteria for a PSO
Table 4.2: Confusion matrix showing actual vs. classified results for ten vehicles after applying MDS/LMDS with an RBF-SVM classifier.

RBF SVM, $\mathbb{R}^4$, $P_{id} = 0.941$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>42</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>43</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>44</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>44</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>42</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>34</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>37</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>42</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>42</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>44</td>
</tr>
</tbody>
</table>

Figure 4.8: Typical attributed point patterns in the SAR experiment contained on the order of 200 points per set, and calculating the minimized LTS-HD, using a particle swarm optimization, required approximately 15 seconds.
is an open problem [84], based on training data, we determined that 100 particles for 100 iterations provides stable results. Each minimized LTS-HD was calculated in approximately 15 s; a nearest neighbor test on the 60 training samples in the SAR database required 15 minutes, when single threaded. Both the PSO and nearest neighbor tests are highly parallelizable for real time performance.

4.4.4 Comparison to MSE Classifier

To provide a benchmark comparison, we use the mean squared error (MSE) classifier. MSE provides a generalized likelihood ratio test for translated patterns in additive white Gaussian noise, and is therefore widely used; but, MSE is known to be brittle in the presence of occlusion, clutter, and other nonideal conditions [34–36]. We compared the minimized LTS-HD classifier to the MSE classifier as applied to the data set of Chapter 2.2.1. Whereas the LTS-HD based classifier compares sets of vectors (attributed point patterns), the MSE classifier compares images like those in Figure 2.4a.

When comparing a query image to a database image, the minimized LTS-HD classifier uses a particle swarm optimization to find the registration that minimizes the distance over rigid transformations (both rotation and translations in a plane). On the other hand, the MSE classifier uses an FFT based method to align the images in translation prior to calculating the Euclidean distance between high dimensional vectors of columnwise ordered image pixels. We minimize the MSE distance over 72 possible rotations with 5 degree spacing. The 60 vehicles of the point pattern database required 240 kB of storage, while the database images for the MSE method required 15.7 MB.
(a) MSE, $180^\circ$, $P_{id} = 0.159$

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

(b) LTS-HD, $180^\circ$, $P_{ID} = 0.864$

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>38</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 4.3: When classifying the Maxima using 180 degrees of aspect, the MSE classifier performs poorly (a), while the minimized LTS-HD shows a relatively good performance (b).

The results showed that the MSE classifier had only two misclassifications of the 440 samples for a 99.6% performance; this compares to the LTS-HD method performance of 97.7% from Table 4.2(a). Classification time for the MSE was also faster, requiring 6.1 s to generate a distance between the query and each training sample (using MATLAB on the Intel Core i7-920) versus 15 s for the LTS-HD method. These results for the MSE based classifier, are impressive; however, under partial occlusions of the query, the LTS-HD shows a strong performance advantage.

Suppose that half of data from the query samples is occluded [91]; i.e., the line of sight to the vehicles was blocked for 180 degrees of the circular aperture. The LTS-HD based classifier appropriately registers the available data within the training database images, while the MSE based classifier fails to register the images properly.

We ran a comparative experiment on row five of Table 4.2(a), and the results are shown in Table 4.3. The MSE classifier performs at 15.9% compared to 86.4% for the LTS-HD classifier. Furthermore, with fewer points in the query sets, the LTS-HD classifier query times are reduced by approximately 50 percent.
Figure 4.9: After extracting 3D information for attributed scattering centers (a) displays all of the data for a Camry. Reducing the data set to bright peaks gives (b). Similarly, the reduced Maxima is shown in (c). The red rectangles represent estimated base outlines.

### 4.4.5 Single-pass 3D Example

In this section, we present a case study of 3D feature extraction for vehicle classification. Ideally, 3D imaging provides some invariance to vehicle position across the wide-area scene, thereby reducing the measured distance between training samples of the same class. Using MDS visualizations in $\mathbb{R}^2$, we compared the CVDomes Camry (v1) to the Maxima (v5), which are similar midsize sedans.

To extract 3D information, we processed the 2D imagery for each of the circled training locations of Figure 2.5(b) using the single pass 3D algorithm [21] of Chapter 2.5. Figure 4.9(a) shows a 3D extraction of the Camry using all peaks down to -30dB of the brightest for 360 overlapping 5-degree apertures. Figures 4.9(b) and (c) show the attributed scattering centers, of the Camry and Maxima, retained after thresholding to the top 15dB amplitudes and a decimating to every third azimuth. For all three figures, the blue points represent elevated odd-bounce scattering centers.
while the green points represent scattering centers left on the original 2D imaging plane.

Next, using the minimized LTS-HD pseudometric described in Chapter 4.2.1, we filled a distance matrix between training samples and applied MDS for visualization. Figure begins with a 2D example showing that dissimilar vehicles such as the Camry (v1) and Tacoma (v10) are well separated using both linear and RBF SVMs in (a) and (b). On the other hand, v1 and v5 are similar vehicles; Figure 4.10 (c) shows the training samples barely separated with an RBF SVM, while Figure 4.10 (d) shows a failure to separate the training samples with a linear SVM. Finally, using 3D information, Figure 4.10 (e) and (f) show smaller intra-class distances and greater inter-class separability. As a result, both the linear SVM and RBF SVM classifiers easily separate the two sets of training samples. Our preliminary assessment shows a potential for improved classification performance when using 3D information.

### 4.4.6 Results for 2006 Collection

When applying the minimized LTS-HD to the 2006 Collection described in Chapter 2.2.2, we train on the first pass of the radar and classify the vehicles in remaining seven passes. Using the same parameters as for the CVDomes experiments resulted in 100% classification performance, when the query has 360 degrees of aspect [32]. Thus, we ran an experiment based on 45 degrees of aspects.

Figure 4.11 shows the results of reducing the aperture extent to 45°, where the aperture low angle indicates where in polar coordinates the radar begins its counterclockwise collection. Different aspects are particularly affected by clutter or a lack of reflectors. Since all of the vehicles are generally oriented in the same direction in the
Figure 4.10: The 2D training samples of two dissimilar vehicles like a Camry (v1) and Tacoma (v10) are well separated by an RBF SVM and a linear SVM as shown in (a) and (b), respectively. However, similar vehicles like a Camry and a Maxima (v5) are more difficult to separate in (c) and (d). After extracting 3D information, the Camry and Maxima are more easily separated as as displayed in (e) and (f).
Figure 4.11: Classification performance, using a 45-degree aperture, is dependent upon viewing angle. Dips in the plot may coincide with corners of the vehicle or clutter from curbs. Some viewing angles such as the 250-300 degree range give higher performance.

parking lot there are some interesting trends. The dips tend to be at corners of the vehicles, while peaks coincide with the broad sides or front/back of the vehicles.

4.5 Discussion

In this chapter, we described a procedure for classifying sets of attributed point patterns, where each point pattern is a collection of feature vectors. Registration of point patterns in the translation-variant case is obtained using a version of the Hausdorff distance, called the LTS-HD, that is minimized over a set of transformations. The minimized LTS-HD was termed the set distance. As a novel contribution to the point pattern registration and set distance calculation, we chose a Mahalanobis distance to calculate distances between individual points. This allowed for a principled
fusion of feature attributes using the normalization provided by the error covariance matrix. The minimized LTS-HD, when comparing a query to a database, was used in a nearest neighbor classifier.

For the case of multiple training samples per class, we described an implementation of multidimensional scaling to embed sets of point patterns in a Euclidean space. The MDS algorithm requires a matrix of pairwise distances between samples; in our application, we used the minimized LTS-HD based pseudo-distance. After embedding point patterns within the Euclidean space, it was possible to visualize the class separation in dimensions of 3D or less. In addition, the eigenvalues that were generated as part of the MDS process were useful in determining the suitability of the pseudo-metric as a distance measure. It was also possible to train standard classifiers such as LDA or SVM, and embed query samples in the same Euclidean space using LMDS.

The proposed classification strategy was applied to yield the first reported radar classification results for civilian passenger vehicles. The use of attributed point sets provided high-confidence classification with robustness to obscuration. The use of a Mahalanobis point distance allowed principled scaling of attributes that enhance class separability. In addition, a fingerprint application extended minutiae registration results from [3] to demonstrate the generic applicability of the proposed classification approach using attributed point sets.
CHAPTER 5: Classifying Vehicles with Pyramid Match Hashing

The material for this chapter is based on the manuscript: K. E. Dungan and L. C. Potter, “Classifying vehicles in wide-angle radar using pyramid match hashing,” submitted to IEEE Journal of Selected Topics in Signal Processing, April 2010.

5.1 Introduction

In this chapter, we present a new classification algorithm that uses small file sizes and has fast query times. The proposed algorithm compresses SAR imagery into attributed scattering centers [2] that are stored in a database of pose invariant hash codes for a later query. To accomplish this, we combine three concepts.

1. Map attributed scattering centers to localized sets that are invariant to rotations and translations [27–29].

2. Apply an indexing method called pyramid match hashing [31] (PMH); PMH provides an implicit correspondence between local features, yields a partial match similarity score, and enables time complexity that is sublinear in the size of the library database.

3. Perform a fusion [92,93] of results for individual scatterers.

Although the presentation of this chapter is focused on SAR vehicle classification, the proposed algorithm is presented in generic terms and may be applicable to other
problems that use point patterns with attributes and unknown translation or orientation. The experimental results are designed to both illustrate the proposed point pattern matching approach and to establish a performance baseline for millimeter-wave classification of civilian vehicles.

### 5.1.1 Overview of Algorithm

Figure 5.1 displays an outline of the proposed classification algorithm. For both the training database and query inputs, we assume inputs are a series of well-focused subaperture SAR images that are oriented to a global frame of reference. In an offline step, wide aperture circular SAR training imagery is abstracted to attributed scattering centers (i.e., attributed point patterns) that are approximated from peak locations in the images, as described in Chapter 2. The input parameter “Limit Sets” simply indicates that the extraction routine may have options. In the SAR example, we limit the set size by thresholding on a minimum peak amplitude.
The point patterns have two-dimensional location information, and hence vary with respect to rotations and translations in a plane. The invariant mapping block describes each point by a local neighborhood of its $K$ nearest neighbors in a local frame of reference that is defined by a vector from the point to its nearest neighbor or by a measured attribute, if available. In the SAR application, the azimuth to the radar is an attribute that orients the local set. Distances between attributed points are measured using a Mahalanobis distance described by the measurement error covariance matrix $\Sigma$. As a result, each scattering center is represented as a set of attributed points, describing the local scattering behavior. The vehicle is represented by a set of sets.

Subsequently, the PMH algorithm represents each local set with an $M$-bit binary hash code for fast digital processing. The hash coding process requires quantizing the data into a multi-level hypercube with a base side length defined by the parameter $N_0$. Furthermore, PMH requires weights $W$ that are related to the distances between implied correspondences when comparing two coded point patterns; we develop a method to approximate Bayes optimal weights.

As shown in Figure 5.1, a query image is subjected to the same extraction, invariance, and PMH steps as during offline processing. Each hash code, from the query, is compared to the hash codes of the database using a Hamming distance to find the $N_n$ Hamming nearest neighbors (HNN) quickly. For these few HNNs, a pyramid match kernel (PMK) [45] is computed to provide the final measure of similarity in comparing the invariant, local, attributed point patterns. Finally, we fuse evidence from the local point sets to classify the query.
The remaining sections of this chapter are organized as follows. First, in Chapter 2.2, we describe the extraction of attributed point patterns and the formation experiments using the CVDomes and 2006 data sets. Next, in Chapter 5.2, we describe the pose invariant mapping of point patterns along with Mahalanobis-based distances and whitening of the data. Then, in Chapter 5.3, we describe the PMH algorithm along with a method to approximate Bayes optimal weights. Chapter 5.4 overviews the classification process along with a discussion of hash code lengths and a fusion of decisions. Finally, in Chapter 5.5, we analyze the proposed algorithm as applied to the CVDomes and 2006 data sets of Chapters 2.2.1 and 2.2.2.

5.2 Invariantly Mapped Database

As described in Chapter 2.1, circular SAR imagery is extracted into sets of attributed scattering centers. For the proposed algorithm, each scattering center is represented as a vector containing the $x$ and $y$ location, the center azimuth $\theta$ of the associated imaging aperture, the amplitude $\iota$ of the peak, and an even/odd bounce bit $\nu$ to encode polarization. The features are a portion of the information in (2.1); we do not use the elevation or height attributes in this section, although the algorithm supports any number of attributes. Thus, a vehicle sample is represented by a set containing all of the attributed peaks,

$$P = \{p_1, \cdots, p_{N_p}\} \subset \mathbb{R}^f,$$  

(5.1)

where $f = 5$ is the number of features and $|P| = N_p$ is the total number of peaks. Individual points are represented by the vector

$$p_i = (x_i, y_i, \theta_i, a_i, \nu_i)^T \in P.$$  

(5.2)
Figure 5.2: An example attributed point pattern with an orientation (azimuth) attribute is displayed on the left. A point \( p_i \) is represented by a set \( H_i \) containing the point along with three nearest neighbors in a local frame of reference.

A set of attributed points describes a class sample; this set is considered equivalent to another set under rigid transformations for purposes of pattern recognition. For example, in the SAR application, a translated or rotated vehicle is the same vehicle. However, matching sets of points under rigid transformations is computationally intensive. For example, to estimate unknown pose and translation, Dungan and Potter applied a nonlinear optimization called a particle swarm to minimize a distance between point patterns [32,33].

To avoid the nonlinear optimization, some researchers have aligned point patterns using an orientation attribute [28,29]. We have adapted a variation of local alignment to the SAR recognition problem. Given a set \( P \) from (5.1), we proceed with the invariant transformation and whitening steps.

First, as illustrated in Figure 5.2, we represent each point in \( P \) by a set of its \( K \) nearest neighbors, counting itself. Distances between points, \( \{p_i, p_j\} \subset P \), are measured using a Mahalanobis distance,

\[
d(p_i, p_j) = \sqrt{(p_i - p_j)^T \Sigma^{-1} (p_i - p_j)},
\]

(5.3)
where $\Sigma$ is the measurement error covariance matrix for the vectors in a point set. The Mahalanobis distance facilitates the comparison of vectors, where the various dimensions have different scales, different error sources, or correlated errors. For example, in radar, some attributes may contain spatial locations of bright reflectors, while other attributes contain information about intensity, polarization, or direction of illumination. Use of an appropriate error covariance matrix increases class separability. In practice, it is typical to estimate the measurement error variances of each feature in the feature vectors to populate the diagonal of $\Sigma$; however, determining the off-diagonal covariance terms may improve results.

A local origin is given by the point, and the direction of the azimuth attribute defines a local $x'$-axis with the $y'$-axis 90 degrees counterclockwise. Each of the $x$, $y$, and $\theta$ attributes for the $K$ nearest neighbors of $p_i$ is transformed to the local frame of reference ($a$ and $\nu$ attributes do not change). Thus, each $p_i \in P$ is represented as a set of $K$ points.

Second, the transformed local sets are whitened to give an identity error covariance, a desirable property for the PMK of the next section. Sets are whitened using the transformation,

$$T_w = \Lambda^{-\frac{1}{2}} V^T,$$

such that $\Sigma = V \Lambda V^T$.

In summary, each $p_i \in P$ is mapped to a transformed and whitened set along with its $K$ nearest neighbors; hence,

$$p_i \rightarrow H_i = \{h_{\xi_1^{(i)}}, h_{\xi_2^{(i)}}, \ldots, h_{\xi_K^{(i)}}\},$$

(5.5)
where \( \{\xi_1^{(i)}, \xi_2^{(i)}, \cdots, \xi_K^{(i)}\} \) is a sequence of indexes in \( P \) to the \( K \) nearest neighbors of the \( i^{th} \) element of \( P \). Thus \( P \) is mapped to the set of sets,

\[
P = \{p_1, p_2, \cdots, p_{N_p}\} \rightarrow \{H_1, H_2, \cdots, H_{N_p}\}.
\]

(5.6)

As a result, each point of each class sample is represented as a set that is considered a database entry. The training database \( \mathcal{H} \) consists of a union of the mappings for the class samples, creating one big database. Thus,

\[
\mathcal{H} = \{H_1, H_2, \cdots, H_{N_h}\}
\]

(5.7)

is a set of sets containing an entry for every point of every training sample; it is necessary to maintain an index cross referencing the \( |\mathcal{H}| = N_h \) members of \( \mathcal{H} \) to a class label.

When accessing the database, a query is invariantly mapped and whitened to a set of sets

\[
\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_{N_q}\},
\]

(5.8)

where \( |\mathcal{Q}| = N_q \) is the same as the number of points in the query attributed point pattern. The next section develops a method to measure similarity between members of the query \( \mathcal{Q} \) and the database \( \mathcal{H} \).

### 5.3 The Pyramid Match

The pyramid histogram [45], pyramid match kernel [45], and pyramid match hashing [31] are described in detail by Grauman and Darrell; here, we provide a brief overview of the concept along with insight into our implementation.

A critical computational challenge in partial point pattern matching is correspondence – the assignment between a subset of query points and a subset of a database.
entry. Here, we adopt a multiresolution pyramid to implicitly produce an approximate nearest-neighbor correspondence, while also computing a measure of similarity. As shown below, for class-conditional Gaussian feature uncertainties, the measure approximates the relative posterior probabilities of each class, conditioned on the implicit correspondence. A hash code implementation of pyramid matching enables complexity that is sublinear in the number of database entries.

5.3.1 The Pyramid Histogram

To begin, the data must be represented as pyramid histograms, described as follows:

- For the base layer of a pyramid histogram, scale, shift, and quantize the $K$ points for a set $H_i \in \mathcal{H}$ onto a discrete grid of integers in $\{1, \cdots, N_0\}$ for each of the $f$ dimensions. The $N_0^f$ discrete locations represent single bins of a very sparse histogram;

- For subsequent layers, repeatedly decimate the grid by a factor of two in each dimension while accumulating the bin counts. Eventually all points in $H$ are counted in a single bin;

- For $L = \lceil \log_2 N_0 \rceil$ layers, there are $(N_0/2^l)^f$ bins per layer for $l \in \{0, \cdots, L-1\}$.

5.3.2 The Pyramid Match Kernel

For a database entry $H_i \in \mathcal{H}$ and a query $Q_j \in \mathcal{Q}$, the PMK is a measure of similarity between two pyramid histograms given by

$$
P_{\Delta}(\Psi(H_i), \Psi(Q_j)) = w_{L-1}I_{L-1} + \sum_{l=0}^{L-2} (w_l - w_{l+1})I_l, \quad (5.9)
$$
where $\Psi$ is the pyramid histogram function, $w_l$ is a weighting function, and $I_l$ is an intersection between the $l^{th}$ layers of the two pyramid histograms. To create a Mercer kernel, the weighting function must be monotonically decreasing [45]. The intersection between each level of two pyramid histograms $\Psi(H_i)$ and $\Psi(Q_j)$, $I_l$, is defined as the sum of the minimum number in corresponding bins between the hypercubes at that pyramid level. This has the effect of approximating a one-to-one correspondence of nearest neighbors between the two point patterns.

For the SAR application, we quantize the whitened data, using $N_0 = 512$ and $f = 5$, to a $512^5$ bin histogram. The entries of the invariantly mapped databases, for both the CVDomes and 2006 collection, adequately fit in this grid. The hypercube is large, and the complexity of calculating the PMK is based in large part on the time to intersect the two hypercubes. However, the hypercubes are sparsely populated and the intersection is a matter of comparing populated indices. Thus time complexity is on the order of the number of points in the sets $H_i$ and $Q_j$ as opposed to the size of the hypercube.

### 5.3.3 Pyramid Match Hashing

The PMK provides a measure of similarity that can be used to implement a classifier. Because the data does not exist in a Hilbert space, classical methods such as the KD-tree are not suitable enhancements to speed up the search. However, the PMH algorithm provides a method to preprocess the database into binary hash codes, where queries are compared to the database using Hamming distances (sum of XOR between binary codes). A Hamming nearest neighbor (HNN) search requires linear time complexity in calculating Hamming distances, but it is significantly faster.
than calculating the PMK. Furthermore, it is possible to perform an approximate Hamming nearest neighbor (AHNN) search in sublinear time [31].

Let $z$ denote a vectorized and square-root, weighted pyramid histogram [31] for a member of $H$ or $Q$. An $M$-bit hash code $b$ is generated from a 1-bit quantized linear map of $z$,

$$b = \text{sgn}(\Phi z),$$

where $\Phi \in \mathbb{R}^{M \times N}$ is a zero mean unit variance random matrix. The column count $N$ is the size of the vectorized pyramid histogram, i.e.,

$$N = \sum_{l=0}^{L-1} (N_0 2^{-l})^f. \quad (5.11)$$

Since $\Phi$ is very large, while $z$ is sparse, we seed a fast Mersenne Twister [94] pseudorandom number generator with an index to the entry of $\Phi$ to generate values on demand, when computing (5.10).

### 5.3.4 Approximating Bayes Optimal Weights in PMK

Weights in the PMK algorithm are inversely proportional to the size of the histogram bins at each level [45]. Thus, as bins increase in size, the importance of an intersection decreases. Notionally, an intersection between two histogram pyramids gives information about the expected distance between a point in the query set and in the hypothesis. Here we demonstrate design of PMK weights to approximate Bayes optimal decisions. This section adapts Chiang’s [41] model-based Bayesian feature matching concept to a database driven PMK/PMH-based algorithm.
Deriving the Bayes Equations

The probability \( \pi_{ij} \) that a query sample \( Q_j \in Q \) comes from hypothesis \( H_i \in \mathcal{H} \) is represented by the equation

\[
\pi_{ij} = P(H_i|Q_j). \tag{5.12}
\]

By design, both the query and hypothesis sets contain a fixed number of feature vectors \( K \). From Bayes rule,

\[
P(H_i|Q_j) = \frac{P(H_i|Q_j, K)}{f(Q_j|K)} \tag{5.13}
\]

Using a generalized likelihood ratio test (GLRT) classifier,

\[
f(Q_j|H_i, K) \approx \max_{\Gamma_{ij} \in \mathcal{G}} f(Q_j|\Gamma_{ij}, H_i, K), \tag{5.14}
\]

where \( \mathcal{G} \) is the set of one-to-one correspondence maps between the elements of \( q_{jk} \in Q_j \) and \( h_{im} \in H_i \). The correspondence \( \Gamma_{ij} = \{\Gamma_{ij}^1, \Gamma_{ij}^2, \ldots, \Gamma_{ij}^K\} \) lists \( K \) pairs of indices between \( Q_j \) and \( H_i \) that maximize (5.14).

Chiang describes an uncertainty model and the related integral used to calculate \( f(Q_j|\Gamma, H_i, K) \). The uncertainty model for the query is characterized by the error covariance matrix \( \Sigma_q \), while the actual class template \( G_i \) differs from the actual object hypothesis \( H_i \) by the covariance matrix \( \Sigma_h \). Thus, \( \Sigma_q \) describes the measurement noise and \( \Sigma_h \) describes within class object variability.

Next, since the feature vectors describe physically different scatterers, we assume the feature vectors are conditionally independent; thus

\[
f(Q_j|\Gamma, H_i, K) = \prod_{k=1}^{K} f(q_{jk}|\Gamma_k, H_i, K), \tag{5.15}
\]
where the feature vector $q_{jk}$ is in correspondence with the $\Gamma^h_k$ feature vector of $H_i$.

Next, focusing on a particular query feature $q_{jk}$ and assuming uncorrelated Gaussian errors in the query and hypothesis,

$$f(q_{jk}|\Gamma_k = m, H_i, K) \sim \mathcal{N}(g_{im}, \Sigma_h + \Sigma_q), \quad (5.16)$$

where $g_{im} \in G_i$ is in correspondence with $q_{jk}$. Hence, using the squared Mahalanobis distance between $q_{jk}$ and $g_{im},$

$$d_s(q_{jk} , g_{im}) = (q_{jk} - g_{im})^T \Sigma_s^{-1} (q_{jk} - g_{im}), \quad (5.17)$$

we have

$$f(q_{jk}|\Gamma_k, H_i, K) = \frac{1}{(2\pi)^{f/2} |\Sigma_s|} e^{-d_s(q_{jk} , g_{im})/2}, \quad (5.18)$$

where $\Sigma_s = \Sigma_h + \Sigma_q$. In (5.18), we have two corresponding feature vectors, one from a database sample and one from a query sample. The query vector is conditionally distributed about the database vector with a Gaussian distribution. In the following section, we describe how to estimate the distance in (5.17) in generating PMK weights.

**Quantizing Data for the PMK**

For the PMK, it is desirable that $\Sigma_s$ in (5.18) be a scalar multiple of the identity matrix since the quantization grid is uniform in all dimensions. As described in Chapter 5.2, the data is approximately whitened during the transformation invariant mapping process such that $\Sigma_h$ and $\Sigma_q$ are approximately identity. Combining the errors for use in (5.17), requires either multiplying the data sets by $1/\sqrt{2}$ such that $\Sigma_s = I$ or simply using $\Sigma_s = 2I$. 

96
In the PMK, the vectors in the database and query samples are compared through a progression of multi-resolution levels in the pyramid histogram. At the first occurrence of an intersection, the vectors are considered a correspondence, $\Gamma_k$. This approximates the nearest neighbor between a database vector and query sample using the Mahalanobis distance (5.17) in the exponent of (5.18). The nearest neighbor is only approximate since the quantization caused by the pyramid histogram grid may occasionally push a close correspondence to a higher level.

Chiang’s [41] method finds $\Gamma$ in (5.14) using the Hungarian algorithm search over correspondences between $Q_j$ and $H_i$ with a polynomial time complexity of $O(|Q_i| + |H_j|^3)$ or $O((2K)^3)$. As the Hungarian search is over an exhaustive set of correspondences, our nearest neighbor method may be suboptimal. But, by using the PMK, an implied correspondence is generated in $O(fK \log N_0)$ from Chapter 5.3.1, where $f$ is the feature dimension and $N_0$ is the maximum size of the quantization grid. Thus, the PMK offers a significant advantage in time complexity. Furthermore, the PMK facilitates the use of PMH for a reduction in the number of PMK calculations in a multiclass problem.

When applying the PMK between a query and a database template, a particular correspondence from an intersection at some level $l$, tells us that $g_{im} \in A$ and $q_{jk} \in B$ as shown in Figure 5.3. Due to the quantization, when storing the database in pyramid histograms, we do not have the exact location of $g_{im}$ in (5.18). Therefore, we consider $g_{im}$ a uniform random variable in the region $A$, and $q_{jk}$ is normally distributed about $g_{im}$ and in the region $B$. 

97
Figure 5.3: If an intersection is detected in the first level of a pyramid histogram, the two feature vectors exist within the smallest resolution bin, as shown on the left. In higher levels of the pyramid histogram, the hypothesis feature vector exists in set $A$, while the query vector is in $B$.

Suppose we detect a correspondence; conditioned on this correspondence, the mean squared distance at level $l$, 

$$
    d_l(q_{jk}, g_{im}) = E\left[ d_s(q_{jk}, g_{im}) | q_{jk} \in B, g_{im} \in A \right],
$$

is given by an integral equation not expressible in closed form [95]. Here, we approximate this expectation via numerical integration using Monte Carlo trials. Figure 5.4 displays a 2D example of a Monte Carlo simulation for $l \in \{0, 1, 2, 3\}$ and $\Sigma_s = I$. The red circles indicate a sample $g_{im} \in A$, while the blue asterisks represent a corresponding $q_{jk} \in B$. This simulation displays only the 1000 pairs corresponding between regions $A$ and $B$. The mean squared distances are $[0.476, 1.01, 1.41, 1.54]$.

In the SAR vehicle classification application, we quantize five-dimensional feature vectors onto a $512^5$ grid, which implies $L = \log_2 512 - 1 = 10$. Using $10^6$ Monte Carlo trials and $\Sigma_s = 2I$, the mean expected distances between corresponding points detected at a level $l \in \{0, 1, \cdots, 9\}$ in a pyramid match are

$$
    \delta = [0.66, 1.2, 3.0, 4.4, 4.8, 4.9, 4.9, 5.0, 5.0, 5.0].
$$

98
Figure 5.4: Hypothesis points are uniformly distributed in a resolution bin, while the query samples are normally distributed with respect to the hypothesis in the next level of the pyramid histogram. The expected distance at any level is determined by the average distance between corresponding pairs.

The distances are only shown to two significant digits; however, it is clear that correspondences must be detected in the first four to five levels to have any significant contribution to a decision, as expected.

To apply the distances in (5.20) to the low-complexity PMK procedure, the distances must be converted to positive, monotonically decreasing weights [45]. This can be accomplished by solving (5.18) for the distances in (5.20), which effectively converts the distances to radial basis function kernel values. The PMK finds the sum of the kernels found in the correspondence. However, to be consistent with the
Bayesian interpretation in (5.15), which requires a product, we take the sum of the log of the kernels. Using the nearest neighbor approximation to optimal correspondence to generate $\Gamma_{ij}$,

$$
\log f(Q_j|H_i, K) \approx \sum_{k=1}^{K} \log \left( \frac{1}{(2\pi)^{f/2}|\Sigma_k|} e^{-\hat{d}(q_{i,k}, g_{i,m})/2} \right) \Lambda_{ij}(k),
$$

(5.21)

where $\hat{d} \in \delta$ represents the quantized distance in the PMK. For abbreviation, we use $\Lambda_{ij}(k)$ as in (5.21) and

$$
\lambda_{ij} = \exp \left( \sum_{k=1}^{K} \Lambda_{ij}(k) \right) \approx f(Q_j|H_i, K).
$$

(5.22)

Since $\hat{d}$ takes on values in the finite set $\delta$, its values are bounded, and a positive shift $\beta$ by the minimum value of $\Lambda_{ij}$ over the elements of $\delta$ and dividing by the maximum resulting value $\alpha$ gives valid PMK weights. Thus, for our application (showing two significant digits),

$$
W = [1.0, 0.87, 0.46, 0.14, 0.052, 0.024, 0.012, 0.0043, 0.0021, 0],
$$

(5.23)

provides the approximate Bayes optimal weights for use in the PMK.

### 5.4 Classification Process

By applying the Bayesian based weights of Chapter 5.3.4 to the hashing process of Chapter 5.3, we can generate a database of hash codes. Figure 5.5 shows an overview of the invariant mapping and hash coding process of Figure 5.1. The scattering centers of each class sample are represented by an $f$ dimensional attributed point pattern. Each point of each class sample is mapped to a rotation and translation
Figure 5.5: Each point of each training sample is mapped to a set of nearest neighbors in a local frame of reference. These sets are coded into $M$-bit hash keys. The points of a query sample are also coded to $M$-bit hash keys. The query codes are compared to database entries using Hamming distances.

invariant set along with its $K - 1$ nearest neighbors. These sets are then hash coded into a database. For example, 60 class samples with 200 points each would generate a database of 12,000 hash codes in an offline process.

Next, given a query image, a point pattern is extracted, invariantly mapped, and hash coded to generate a code for each point in the query pattern. As shown in Figure 5.1, the query hash codes are compared to the database hash codes using Hamming distances. For each query hash code, we select a number $N_n$ of Hamming nearest neighbors (HNN), and subsequently, we calculate the PMK from the noncoded weighted pyramid histograms.

From the reduced set of PMK calculations, for each point in the query, we fill a matrix $D \in \mathbb{R}^{N_c \times N_q}$, where $N_c$ is the number of classes and $N_q$ is the number of points in the query pattern. For local region $Q_j$ and class $c$ we have

$$D_{cj} = \max_{i: \mathcal{C}(i) = c} \sum_{k=1}^{K} \alpha(A_{ij}(k) + \beta), \quad (5.24)$$

101
where $C$ is a “class of” operator. By adopting the maximum in (5.24), we approximate the posterior probability of class $c$ given $Q_j$ using the single largest score within class.

The final classification of a query sample is determined from a fusion of information in the columns of $D$. The following two subsections provide insight into setting the hash code length and the method of fusion.

### 5.4.1 Hash Code Length

Intuitively, a smaller Hamming distance between hash codes indicates that the point patterns being compared are more similar. More specifically, the probability that two hash bits are the same is [31]

$$
\text{Pr}[\text{same bit}] = 1 - \frac{\cos^{-1}(\mathcal{P}_\Delta(\Psi(H_i), \Psi(Q_j)))}{\pi};
$$

hence, a closer match from the PMK implies a higher probability that the hash bits are the same. In a classification, a longer hash code means that $N_n$ can be smaller, resulting in fewer PMK calculations. On the other hand, a longer hash code increases the computation time necessary to encode a query and calculate the Hamming distances. It appears that the benefits of lengthening the hash code are not linear and diminish with length.

The compressive sensing literature shows that it is possible to recover a signal from a 1-bit encoding given a key of length of $M = O(\kappa \log(N/\kappa))$ [96], where $\kappa < KL$ is the number of non-zero bins in the pyramid histogram. In our SAR example, using $f = 5$, $L = 10$, and a $N_0 = 512$, implies $M = O(10^4)$. However, in a classification problem, it is not necessary to reconstruct the original signal, and the key length can be shortened such that the closest members of the database can be distinguished [97].
It is not yet clear how to analytically determine the shortest sufficient key length such that the HNN is the maximum PMK. To mitigate this problem, we find a number of HNNs in the database for a given key length, and evaluate the PMK on only those database entries. In the applications section, we empirically analyze the effects of key length and number of HNNs on identification performance.

5.4.2 Fusion of Decisions

Each point in a sample represents a physical scattering center and its relation to neighboring reflectors. In this section, we combine the entries in the array $D$ of PMK values to arrive at a final class decision.

Starting with the matrix of PMK values $D$ from (5.24) and (5.22), it is possible to invert the shifts and scales such that

$$
\lambda_{c_j} = \exp \left( \sum_{k=1}^{K} \Lambda_{c_j}(k) \right) = e^{D_{c_j} \alpha / K \beta},
$$

where $\Lambda_{c_j}$ and $\lambda_{c_j}$ are associated with the index $i$ of the class maximum kernel found in (5.24). Also, substituting $c$ into (5.22) gives

$$
f(Q_j|H_c, K) \approx \lambda_{c_j}.
$$

From the Bayes rule in (5.13),

$$
\pi_{c_j} \approx \frac{\lambda_{c_j} P(H_c)}{f(Q_j|K)}.
$$

We assume that all priors $P(H_c)$ and distributions $f(Q_j|K)$ are the same, yielding $\pi_{c_j}$ proportional to $\lambda_{c_j}$.
Finally, invoking conditional independence among the local neighborhoods, we get

\[
\log P(H_c|Q) = \sum_{j=1}^{N_q} \log \pi_{c_j} + \text{constant}
\]

\[
\approx \sum_{j=1}^{N_q} \log \lambda_{c_j} + \text{constant}. \quad (5.29)
\]

Thus to approximate a maximum a posterior (MAP) decision, we work with \(\lambda_{c_j}\). Here we consider four methods: the product, sum, majority vote, and median rules [92,93].

From (5.29), the Bayes MAP rule (or product rule) is

\[
C = \arg\max_c N_q \prod_{j=1}^{N_q} \pi_{c_j} \approx \arg\max_c N_q \sum_{j=1}^{N_q} \log \lambda_{c_j}
\]

\[
= \arg\max_c N_q \sum_{j=1}^{N_q} D_{c_j} / \alpha - K\beta. \quad (5.30)
\]

The sum rule is given by

\[
C = \arg\max_c N_q \sum_{j=1}^{N_q} \pi_{c_j} \approx \arg\max_c N_q \sum_{j=1}^{N_q} \lambda_{c_j}
\]

\[
= \arg\max_c N_q \sum_{j=1}^{N_q} e^{D_{c_j} / \alpha - K\beta}. \quad (5.31)
\]

The median rule is given by

\[
C = \arg\max_c \text{med}_{j} \pi_{c_j} = \arg\max_c \text{med}_{j} \log \pi_{c_j}
\]

\[
\approx \arg\max_c \text{med}_{j} \log \lambda_{c_j}
\]

\[
= \arg\max_c \text{med}_{j} D_{c_j}. \quad (5.32)
\]

And, the majority vote rule is given by

\[
C = \arg\max_c \sum_{j=1}^{N_q} \Delta_{c_j}, \quad (5.33)
\]

where \(\Delta\) is a binary operator assigning a 1 to the class with the maximum probability for each point and 0 to all others. With similar reasoning as applied to the median
rule,
\[
\argmax_c \pi_{cj} = \argmax_c \log \pi_{cj} \\
\approx \argmax_c \log \lambda_{cj} \\
= \argmax_c D_{cj}.
\] (5.34)

The product rule follows from Bayes rule and the independence assumption of the \(\lambda_{cj}\) scores. The sum, median, and max rules, on the other hand, can be understood as simplified rules that dampen the deleterious effects of approximation errors in computing \(\lambda_{cj}\) values [92].

### 5.5 Experimental Results

In this section, we evaluate the performance of the proposed classification algorithm as applied to the data sets described in Chapter 2.2. Part of this investigation is to determine the effects of the parameters that were outlined in Figure 5.1. Based on the physics of the application, the point set sizes are limited to avoid extracting peaks from noise and excessive clutter, while the error covariance matrix \(\Sigma\) is initialized based on the following.

To design \(\Sigma\) in the Mahalanobis distance, we choose to populate only the main diagonal for simplicity. Our strategy begins by creating rough estimates for the measurement error variances of the five attributes. Suppose that we are 95% confident that our scattering center peak locations \(x\) and \(y\) are within two pixels of their true locations (i.e. within a 12 cm). Thus,

\[
2 = 1.98\sigma_x \Rightarrow \sigma_x^2 = 1.98^2 = 1.96.
\] (5.35)
Now, suppose that the azimuth attribute for each peak is uniformly distributed over the 5 degree aperture; hence

\[
\sigma^2_\theta = \frac{(b - a)^2}{12} = \frac{5^2}{12} = 2.
\] (5.36)

For the amplitude attribute, log-normal and quarter-power normal models have been widely adopted for non-clutter peaks [90]. Under the log-normal assumption, suppose that with 95% confidence, the amplitude measurement is accurate within 5 dB (zero dB is the maximum peak in the data set); thus

\[
5 = 1.98\sigma_a \Rightarrow \sigma^2_a = 2.5^2 = 6.25.
\] (5.37)

Lastly, suppose that the even/odd bounce bit \(\nu\) is correct with probability \(p = 0.99\); thus

\[
\sigma^2_\nu = p(1 - p) = 0.99(1 - 0.99) = 0.01.
\] (5.38)

After experimenting with a training sample, we choose to set \(K = 32\) for the nearest neighbors in the local groups; a typical point pattern had approximately 200 points. After invariant mapping, all sets in \(\mathcal{H}\) had 32 members each with the five dimensions of Equation (5.2). A pyramid histogram with a base level side of \(N_0 = 512\) was sufficient to digitize all of the data; as a result, the histogram pyramids had \(L = 10\) levels.

The PMK weights \(W\) were set as described in Chapter 5.3.4. For the PMH, the hash key length \(M\) and number of Hamming nearest neighbors \(N_n\) are varied to show the effects.
5.5.1 Classification Results using CVDomEs

As described in Chapter 2.2.1, we are classifying vehicles placed on the red dots of Figure 5.6a based on a training vehicles placed on the circled dots. The 60 training vehicles are processed into sets of attributed point patterns, and each point is represented by a pose invariant set of its local surroundings. The resulting database $\mathcal{H}$ contains 13,371 entries; each training vehicle contributed 222 database entries on average.

Figure 5.6b shows a histogram classifying vehicle 6, sample 1 (a pose rotation at the center), using only 400 HNNs ($\approx 3\%$) and a 120-bit hash key. The histogram shows that 173 of 216 points in the attributed point pattern were properly classified in the database. To illustrate a more challenging example, Figure 5.6c shows classification results for vehicle 6 imaged at sample 43 on the edge of the scene. The peak in the histogram is smaller but clear. The farthest position from a training sample is labeled 44; as shown in Figure 5.6d the peak is smaller. To illustrate the effect of using too few hash bits or HNNs, Figure 5.6e shows a near misclassification of sample 44.

After applying the PMH to all 440 test samples in the database, a confusion matrix such as the one in Figure 5.7 accumulates the results. A 120-bit hash key was used to find the 500 HNNs resulting in the identification performance, $P_{id} = 0.986$, which is the trace of the matrix divided by 440.

Next, we ran a series of experiments to show the effects of changing various design parameters. Starting with Figure 5.8, performance is shown to improve with longer hash key length $M$ and fixed $N_n = 400$; the marginal improvement diminishes as $M$ increases. The trend is consistent with the notion that a longer hash code preserves
Figure 5.6: A query sample (vehicle) contains 216 scattering centers; each are identified to a member of the training set. A majority vote based fusion method identifies the query. In (b), v6 is rotated and placed at the center is easily identified. In (c), (d), and (e), the vehicle is placed at the edge of the scene. Setting the Hamming NN parameter too small causes a near misclassification in (e).

Also, the figure shows the result of choosing amongst the product, sum, majority, and median fusion rules. Since the majority vote was the easiest to calculate and gave the best performance, we used this method for the remaining experiments.
Figure 5.7: To calculate classifier performance as parameters are varied, the 440 samples are accumulated in a confusion matrix. The trace over 440 give $P_{id}$. This example shows a 98.6% performance for a 120-bit hash key to find 500 Hamming nearest neighbors.

Figure 5.9 shows the effects of key length and the number of Hamming nearest neighbors on performance. For each HNN, we calculate the slower PMK to find the largest kernel. The plot shows that a longer hash code allows for fewer HNNs when finding the largest PMK values. For the three code lengths displayed, increasing the HNN count beyond 200 offers little benefit. Notice that the far right circle on the $M = 120$ line was generated from the sample confusion matrix in Figure 5.7.

Another observation of note is that neither Figure 5.8 nor Figure 5.9 are strictly monotonic increasing for all cases since we only generated one random $\Phi$ matrix from (5.10) for each data point in the plots. Repeating the experiments by re-randomizing $\Phi$ would show a performance variance around each point in the plot with a smaller variance as the key length or number of HNNs increases.

The final parameters to investigate are local group size $K$ and the error covariance matrix $\Sigma$. As shown in Figure 5.10 (a), there is a sharp increase in performance as
Figure 5.8: As the hash key length increases, classification performance improves. While the sum and majority vote fusion rules provide similar results, the product and median rules lag in this application.

As $K$ approaches 32. This likely implies small local neighborhoods of feature points are less distinguishable than larger groups. The remaining figures in Figure 5.10 show the effects of changing the variances of the attributes. All variances are initialized as described in Section 5.5 and individually tuned to see the effects. Starting with the spatial $x$ and $y$ attributes, our estimated $\sigma_x^2 = 1$ was a good choice. The azimuth attribute was initially selected as $\sigma_\theta^2 = 2$, however, a smaller variance showed improved results; this indicates that the azimuth may have less error than predicted. On the other hand, the amplitude variance $\sigma_a^2$ was likely larger than predicted. This observation is consistent with the high amplitude variability observed for a coherent sensor, due to scintillation and specular scattering [98]. For level terrain, the polarization attribute appears to be reliable, and setting $\sigma_\nu^2 = 0.01$ was a good choice.

Two final tests included turning all attributes off, except for $xy$ coordinates, and reducing the training sets to a single sample at the scene center. For the no attributes
Figure 5.9: More HNNs increase the chance of finding the largest PMK values. A longer hash code reduces the need for more HNNs to get the same performance.

case, performance dropped to $P_{ld} = 0.955$. Using only one training sample per class resulted in $P_{ld} = 0.843$ due to increasing errors in the outer rings of Figure 5.6a.

**Time Complexity**

Using MATLAB with some mex C files, calculating the PMK required 9 ms on an Intel Core i7-920. Thus, calculating the PMK for all 13,371 members of the database $\mathcal{H}$ would require 120 s. By calculating the PMK on only the 400 Hamming nearest neighbors, the CPU required 3.6 s. The overhead required to hash code the query and find the Hamming nearest neighbors required 10.9 s, for a combined total of 14.5 s per query. We expect improved coding and moving the algorithm to C would reduce query times significantly. Decreasing the length of the hash code decreases the overhead, while increasing the necessary number of Hamming nearest neighbors and subsequent PMK calculations. A variety of hash code lengths and HNNs resulted in values in the 10 to 20 second range under the current implementation.
Figure 5.10: Changing the local neighborhood size (a) and error covariance matrix affects performance. The variances are individually tuned for (b) location, (c) azimuth, (d) amplitude, and (e) polarization.

**Comparison to MSE Classifier**

As described in the introduction, we chose to benchmark the proposed algorithm against an MSE algorithm that is minimized over pose rotations. Here, we compare the PMH based classifier to the MSE classifier as applied to the data set of Chapter 2.2.1. Whereas the point pattern based classifier compares sets of vectors
(attributed point patterns), the MSE classifier compares images like those in Figure 2.5.

The MSE classifier uses an FFT based method to align the images in translation prior to calculating the Euclidean distance between high dimensional vectors of columnwise ordered image pixels. We minimize the MSE distance over 72 possible rotations with 5 degree spacing. For 10 vehicle classes and 6 templates per class (see Figure 2.5), the MSE method required 1420 MB versus 7.68 MB for the PMH method.

The results showed that the MSE classifier had only two misclassifications of the 440 samples for a 99.6% performance compared to 98.6% for the PMH based algorithm. However, classification time for the MSE required 6.1 s per database entry for a total of 366 s to classify a query; this compares to a 14.5 s query time for the PMH method. Timing tests used MATLAB on the Intel Core i7-920.

**Partial Occlusion**

The MSE classifier provides a MAP decision under additive white Gaussian noise; thus, excellent performance under ideal conditions is expected. However, the MSE classifier has been reported to be brittle under non-ideal conditions [91]. Here we test classification performance under partial occlusions. To this end, for each of the 440 query images, we randomly select a continuous half of the radar orbit. This simulation approximates the effect of obscuring the radar line of site to a vehicle for 180 degrees of the circular aperture.

Under this occlusion, the MSE classifier performance drops to 15.9%; the errors arise from failure to properly register a query image. In contrast, the PMH method showed 92.6% performance with the majority vote fusion (94.1% with the sum rule).
Figure 5.11: It is possible to achieve 100% classification on the 2006 data. The histogram in (a) shows an example of v5 correctly classified with a 40-bit hash key. Performance (b) peaks with a 30-bit hash key and time complexity increases with key length.

5.5.2 Classification Results using the Circular SAR 2006 Collection

The experiment for the 2006 collection was described in Chapter 2.2.2, where training samples are generated from the first pass of the SAR and test samples are generated from the remaining seven passes. The peak performance of classifying the vehicles using the PMK was 100%; in addition, we evaluate performance versus reduced time complexity.

Figure 5.11 shows results from applying PMH to the 2006 collection. The first plot is a histogram generated when classifying vehicle 5 on pass 4 using a 40-bit hash key and 2% of the Hamming nearest neighbors; vehicle 5 is clearly selected as the decision. Although not shown, the peak is more pronounced as the hash key length is increased to 80 or 120. The second plot shows that a 30-bit hash code was sufficient to achieve 100% classification for the 49 test samples with approximately 0.9 second
query times in the seven member database. The second plot also shows that query times increase as hash key length increases.

5.6 Discussion

The experimental results presented here give the first demonstration that a millimeter wave airborne radar can accurately classify passenger vehicles with only 640 MHz bandwidth. These vehicles have similar length, width, appearance, and total radar cross section. While this observation may be unexpected considering results obtained from traditional narrow-angle approaches [40], the diversity of views in circular SAR enable class separation.

The proposed algorithm for classification of attributed point patterns is fast, scalable, and principled. The technique is generically suitable for point clouds encountered in many application domains. Significantly, the approach is robust to occlusion, in contrast to MSE template matching.

Five characteristics distinguish the proposed method for point pattern matching. First, the attributed point representation provides significant compression of the radar image data. Second, local neighborhoods of points provide invariance to translation and rotation, obviating the need for a computationally expensive search over pose. Third, the pyramid match kernel approach provides an implicit correspondence for partial matching, bypassing the difficult correspondence issue. Fourth, as demonstrated for the first time in this work, the PMK, when applied to whitened point attributes, can be interpreted as an approximation to Bayesian inference, pro-
viding a principled approach to classification. Fifth, a randomized hash code enables approximate computation of the PMK with time-complexity sublinear in the size of the object library.
CHAPTER 6: Conclusion

We were able to identify civilian vehicles from wide-angle SAR imagery using attributed scattering centers. Despite relatively small signatures, with respect to the radar resolution, and similar dimensions, the vehicles were readily distinguishable when placed near the scene center. Vehicles that were located a large ratio of flight path radius from the scene center required additional training samples to maintain a high level of classification performance. The algorithms also performed well under unknown pose and occluded views by accurately registering and comparing the query images to the database templates. The PMH method in particular operated fast on a single processing thread; the performance was over 20 times faster than an image template based approach. Finally, the compressed representation of attributed scattering centers enabled small memory usage with respect to storing SAR images.

The success of the classification algorithms is motivation for continued research in the area; here are four possible extensions. First, improved coding and parallelization could significantly reduce query times; furthermore, developing the approximate nearest neighbor capabilities of the PMH method would allow scaling to large databases with a sublinear increase in complexity. Second, the partial match capability of the PMH and PHD classifiers implies that the algorithms would perform well at rejecting clutter; testing this would require developing a new database with clutter experiments. Third, initial results showed that 3D scattering centers could reduce the number of
database templates in a large scene problem. Improving the single pass 3D or using a rapid multipass 3D algorithm to extract scattering centers could be investigated. In addition to 3D spatial attributes, an advanced feature extractor could provide more information describing scattering centers to improve classification performance. Fourth, in implementing the PMH based classifier, we approximated a Bayes MAP decision rule; a desirable continuation of this approach would add confidence levels to decisions.
APPENDIX A: Appendix

A.1 Cartesian to spherical coordinates

Following conventional radar nomenclature, the angle $\phi$ used throughout this paper is the elevation angle from a scene point to a radar position (and is equivalent to the depression angle for a downward-looking radar platform). Hence, spherical and Cartesian coordinates are related by

$$r = \sqrt{x^2 + y^2 + z^2}$$ \hfill (A.1)

$$\theta = \tan^{-1} \left( \frac{y}{x} \right)$$ \hfill (A.2)

$$\phi = \frac{\pi}{2} - \cos^{-1} \left( \frac{z}{r} \right),$$ \hfill (A.3)

where $r$ is the distance from the origin to the $(x, y, z)$ coordinate and $\theta$ is the angle measured from the $x$-axis toward the $y$-axis. Also, we assume the radar is always in the $+z$ half-space.

Similarly, converting from spherical to Cartesian coordinates requires that

$$x = r \cos \theta \sin(\pi/2 - \phi)$$ \hfill (A.4)

$$y = r \sin \theta \sin(\pi/2 - \phi)$$ \hfill (A.5)

$$z = r \cos(\pi/2 - \phi).$$ \hfill (A.6)
Figure A.1: From the top view of a circular SAR collection, a vehicle $V$ is offset from the patch center $O$. Attaining a desired cross-range resolution on the vehicle requires the aperture $\delta'$, which in turn requires backprojection of the phase history over the aperture $\delta$.

A.2 Variable aperture imaging

The azimuth attribute in the point cube model captures the anisotropic [60, 61] nature of a vehicle’s SAR signature. For any vehicle located away from the scene center, there is a difference between the radar azimuth $\theta$ referenced to scene center and the azimuth $\theta'$ from the vehicle. The difference $\theta - \theta'$ varies over the 360-degree orbit of the radar platform, as does the aperture necessary to maintain a constant crossrange resolution. For example, Figure A.1 shows a top view of a system looking in the negative $\hat{z}$ direction, where the vehicle location is represented by a point $V$ that is located at edge of scene. Radar pulses are labeled in azimuth relative to the scene center; however, to create an image with aperture $\delta'$ requires identification of the pulses to be used in image formation.
Suppose that $\Psi = [R_1, R_2, \cdots, R_m] \in \mathbb{R}^{3 \times m}$ is a set of radar locations for all of the pulses around an orbit of the circular SAR. Then applying (3.4) with $O = O_A$ and $V = O' = O_B$ gives,

$$
\begin{bmatrix}
B \Psi \\
1^T
\end{bmatrix} = B A^T
\begin{bmatrix}
A \Psi \\
1^T
\end{bmatrix},
$$

(A.7)

where $A \Psi = \Psi$ and $B \Psi = \Psi'$. Next, converting the columns of $\Psi'$ to spherical coordinates provides a set of azimuths relative to the local imaging plane. Finally, the azimuth labels in $\Psi'$ can be used to select pulses for local imaging at point B.

### A.3 Derivation of $\phi'$ for Chapter 3.4.2

Referring to Figure 2.3, $q = \|v\|/\|r_v\|$, and

$$
\begin{align*}
 r_v - v &= \|r_v\|e^{j\phi} - \|v\|e^{j\theta_v} = \|v\| \left( \frac{e^{j\theta} - e^{j\theta_v}}{q} \right).
\end{align*}
$$

(A.8)

Since $\phi_0 = \pi/4$,

$$
\|r - r_v\| = \|r_v\| = \frac{\|v\|}{q};
$$

(A.9)

$$
\phi' = \cot^{-1} \left( \frac{\|r_v - v\|}{\|r - r_v\|} \right) = \cot^{-1} |e^{j\theta} - q e^{j\theta_v}|.
$$

(A.10)


