SANDIA REPORT

SAND2013-0724
Unlimited Release
January 2013

LDRD Final Report: First Application of Geospatial Semantic Graphs to SAR Image Data

Randy C. Brost and William C. McLendon III

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@adonis.osti.gov
Online ordering:	http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.fedworld.gov
Online order:	http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2013-0724
Unlimited Release
Printed January 2013

LDRD Final Report: First Application of Geospatial Semantic Graphs to SAR Image Data

Randy C. Brost
Discrete Mathematics & Complex Systems, 1465

William C. McLendon III
Scalable Analysis and Visualization, 1461

Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-1326

Abstract

Modeling geospatial information with semantic graphs enables search for sites of interest based on relationships between features, without requiring strong *a priori* models of feature shape or other intrinsic properties. Geospatial semantic graphs can be constructed from raw sensor data with suitable preprocessing to obtain a discretized representation. This report describes initial work toward extending geospatial semantic graphs to include temporal information, and initial results applying semantic graph techniques to SAR image data. We describe an efficient graph structure that includes geospatial and temporal information, which is designed to support simultaneous spatial and temporal search queries. We also report a preliminary implementation of feature recognition, semantic graph modeling, and graph search based on input SAR data. The report concludes with lessons learned and suggestions for future improvements.

ACKNOWLEDGMENTS

We would like to thank Mr. Jarlath O'Neil-Dunne for many informative technical discussions, and his feature recognition work executed under a very tight schedule. We also thank Jean-Paul Watson for conceiving this work and writing the proposal which enabled it. We also appreciate the help provided by Cameron Musgrove, Steve Castillo, and Bill Hensley in obtaining example SAR data suitable for release. Finally, we gratefully acknowledge the Sandia LDRD office's financial support of this work.

CONTENTS

FIGURES

1. MOTIVATION AND OVERVIEW

Prior work developed semantic graph models of geospatial image data, and showed how these may be used to identify sites matching patterns of features, even if individual features have varying shape and visual attributes (Strip and Watson 2011). The work described in this paper results from a proposal to extend the prior work to include analysis of time, and to apply the extended result to perform search and change detection analysis on synthetic aperture radar (SAR) image data.

However, limited funding led to a narrowing of scope to address initial study of extending semantic graph analysis to include time, and an initial application of semantic graph techniques to SAR data. This report may thus be viewed as a progress report describing work in pursuit of geospatial-temporal semantic graphs, applied to image data captured by a variety of sources including SAR.

This report will review geospatial semantic graphs and outline a design for their extension to include temporal analysis. We will then review feature recognition results obtained for a SAR test data suite, and report the results of applying semantic graph analysis to the resulting recognized features. Finally, we will wrap up with lessons learned and a discussion of possible next steps.

2. GEOSPATIAL SEMANTIC GRAPHS

Remote sensing data, such as overhead aerial imagery, is available in large quantities for a wide range of locations. However, these data sets are difficult to automatically search, for two primary reasons. First, the phenomenology of sensing causes complex artifacts to appear in images that must be distinguished from fundamental terrain features. Second, items of search interest may be described by characteristics that do not map directly to image attributes such as intensity, color, or shape.

Geospatial semantic graphs were proposed as a means to enable automated search for objects using high-level descriptions (Strip and Watson 2011). In this approach, raw image data is reduced to a collection of discrete features, with associated attributes. These features are then represented in the form of a discrete graph, where nodes and edges have attributes. Search for objects of interest is then expressed as a search on this graph. The next section will briefly review prior work developing semantic graphs for the static case without time, and the following section will explain a design for how to extend this work to include time information to enable graph-based change detection.

2.1. Static Case

The basic idea of geospatial semantic graphs is to reduce the complex and voluminous detail in raw image data to its most salient components, represent these distilled components in a discrete graph, and then perform pattern search in the resulting graph. This allows search for items described by relationships between primary components, and also in principle allows simultaneous representation of information from heterogeneous data sources.

Here we review the prior work reported in (Strip and Watson 2011), to provide background for the current work. The basic flow of this prior result is shown in Figure 1.

We begin with remote sensing data. Semantic graphs are fundamentally independent of the specific data modality; in the prior work, the raw input data was a combination of RGB+IR imagery, LiDAR, and GIS data (see Section 3.1).

This data ensemble was analyzed using automated feature recognition techniques to produce a map of basic terrain cover categories: building, trees, grass/shrub, bare earth, road, other paved, and water. This recognition computation is summarized in Section 3.1.

The resulting terrain cover map can be viewed as a discrete representation of the raw input data, where each connected region of a given terrain cover category corresponds to a different discrete feature. A graph is constructed, where each node of the graph corresponds to a region in the terrain cover map. Each node also has a list of associated attributes computed from the region,

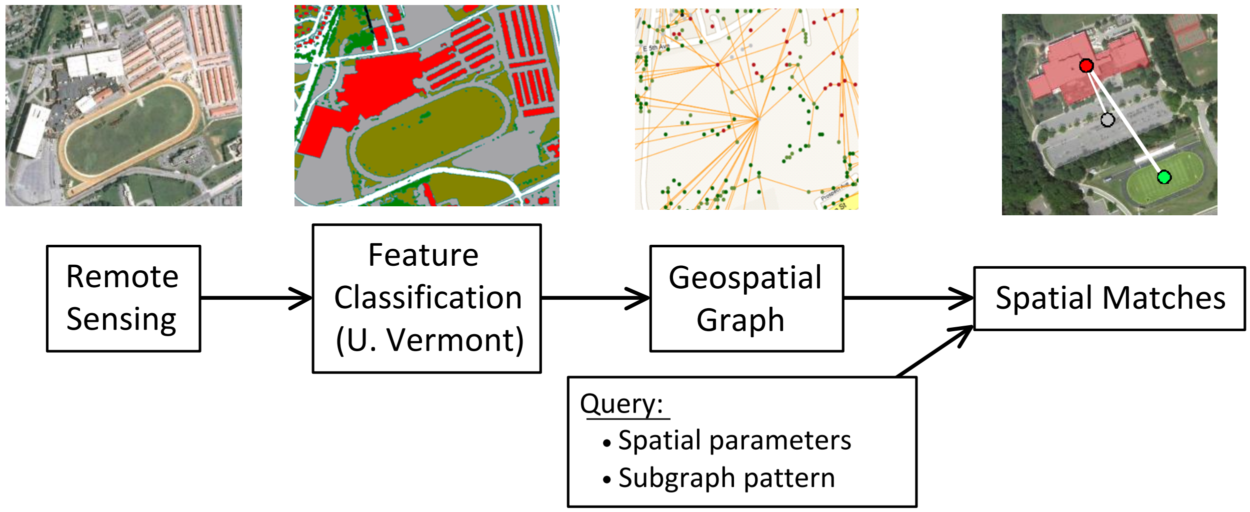


Figure 1. Geospatial semantic graph computation flow.

such as area, aspect ratio, etc. Edges in the graph describe relationships between the nodes; in this case the distance between the convex hulls of the corresponding regions in space. Because the nodes and edges of the graph describe objects in the physical world and retain location and spatial properties, this is a geospatial graph.

The geospatial graph may now be searched for patterns of interest. This is achieved by formulating a query, defined by a small graph search template, where each node and edge in the template has a set of associated attribute bounds. Search is then performed by first filtering the geospatial graph to only include nodes and edges that satisfy the attribute bounds, and then performing an approximate subgraph isomorphism search to find matches between the template subgraph and the remaining elements of the geospatial graph.

This search produces a collection of candidate matches, which are then presented to the user for review. The intent of this process is to provide cues to an analyst for further study by a human expert — not to produce final, definitive answers.

This approach allows automated search for items that defy simple description via pixel-based or shape-based queries. For example, this method was successfully used to search for public high schools in Anne Arundel County, Maryland, based on a search template defined by a combination of a building, parking lot, and grassy field meeting certain specific attribute constraints. Reviewing the search results revealed substantial diversity in the building shapes and layouts of the resulting high school campuses.

2.2. Extension to Include Temporal Analysis

We seek to extend our model of semantic graphs to include temporal analysis. This would allow representation and search of relationships not only in space, but also time. In particular, this would enable change detection informed by feature-level analysis. This in turn would allow future systems to automatically search and analyze complex sequences of large image data sets. The design described in this section was developed in conjunction with a related project which seeks to develop geospatial-temporal semantic graphs for other types of input data. This work is funded by the NA-221 Simulation, Algorithms, and Modeling program, sponsored by Colonel David LaGraffe.

The envisioned extended computational flow is illustrated in Figure 2. Here a time sequence of remote sensing data is received, and again subjected to image recognition analysis to produce a time sequence of terrain cover maps. These maps would then be used to construct a geospatial-temporal graph, which simultaneously represents the spatial and temporal relationships between features in the input data. A query, similar to the pure spatial case but including temporal information in addition to spatial information, would then be used to filter the geospatial-temporal graph to identify relevant elements. The remaining graph would then be searched for matches to the query. This would again return a set of candidate matches for review, but the resulting search would be capable of analyzing spatial, temporal, and combined spatiotemporal properties, as well as feature-level change detection.

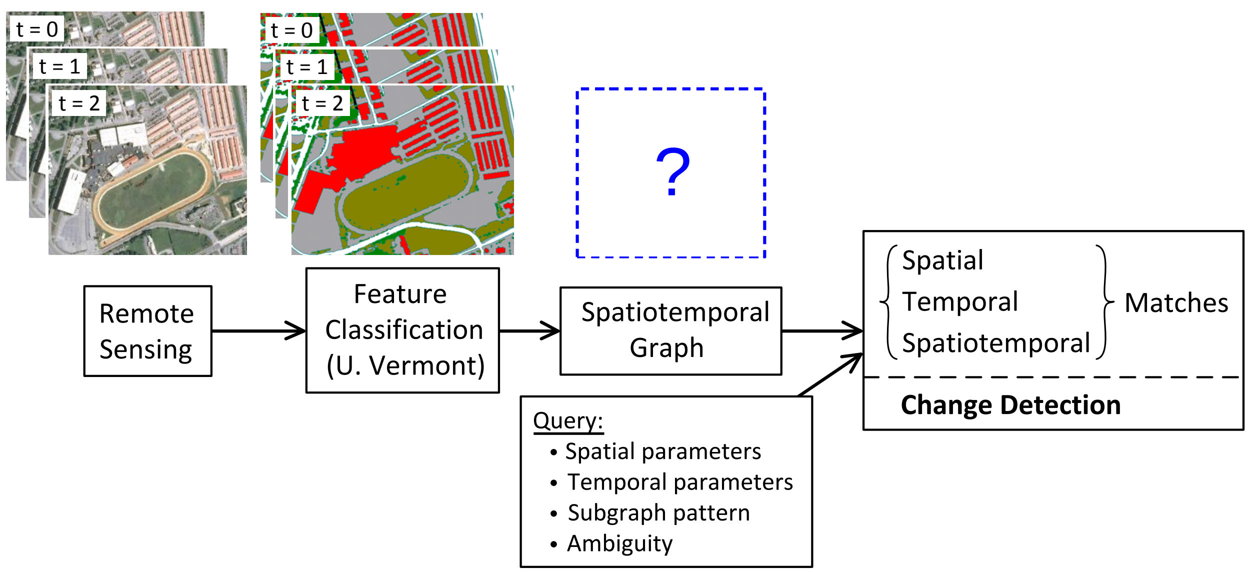


Figure 2. Geospatial-temporal semantic graph computation flow.

This work aims to develop software implementing the computational flow illustrated in Figure 2. A key question then becomes: How should we represent the geospatial-temporal graph?

Figure 3 shows two possible approaches. Part (a) shows the first approach that might spring to mind: Simply construct a geospatial semantic graph for each time slice, and then add edges representing the correspondence between nodes across time slices. This graph contains a full representation of the history of all the time slices, but it also includes significant redundancy. Since many nodes do not change from one time slice to the next, these nodes are unnecessarily replicated.

Part (b) of Figure 3 shows a representation that avoids this redundancy. Nodes are only added to the graph if they have changed; the resulting correspondence edges then imply a change condition. This representation is much more efficient, and tends to concentrate graph complexity where change is happening, which is the area of interest.

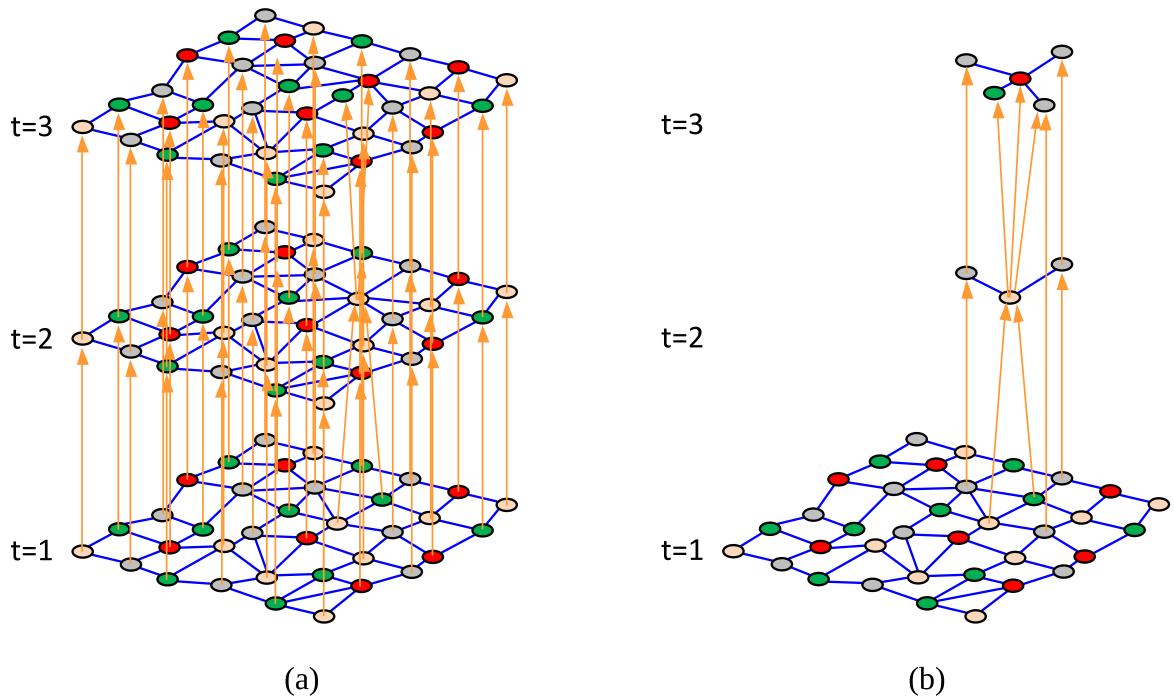


Figure 3. Two approaches to representing time in a semantic graph.

Figure 4 shows a hand-generated example temporal graph construction, for a very simple example. The pictures on the left side of the figure show the sequence of input terrain cover maps. In this scenario, there are four time slices. In the first time slice (a), the terrain cover map includes two buildings, one on each side of a vertical road. Both buildings are surrounded by grass. In the second time slice (b), we see signs of construction for the building on the left; some grass has been replaced by bare dirt. In the third time slice (c) construction is complete; there is a new wing on the building. The fourth time slice (d) includes an additional change, a bridge connecting the two buildings.

The right side of Figure 4 shows the evolution of the semantic graph representing this sequence of events. For the first time slice $t = 1$ (a), the topology of the graph is the same as a geospatial graph, since change cannot be seen in a single time slice. There is a node in the graph for each terrain cover region. The nodes of the graph include various spatial attributes such as area, etc.; these are not shown. The blue edges of the graph denote geometric adjacency relationships. (A full geospatial graph would include additional edges describing the distance between non-adjacent nodes; these are omitted for clarity.)

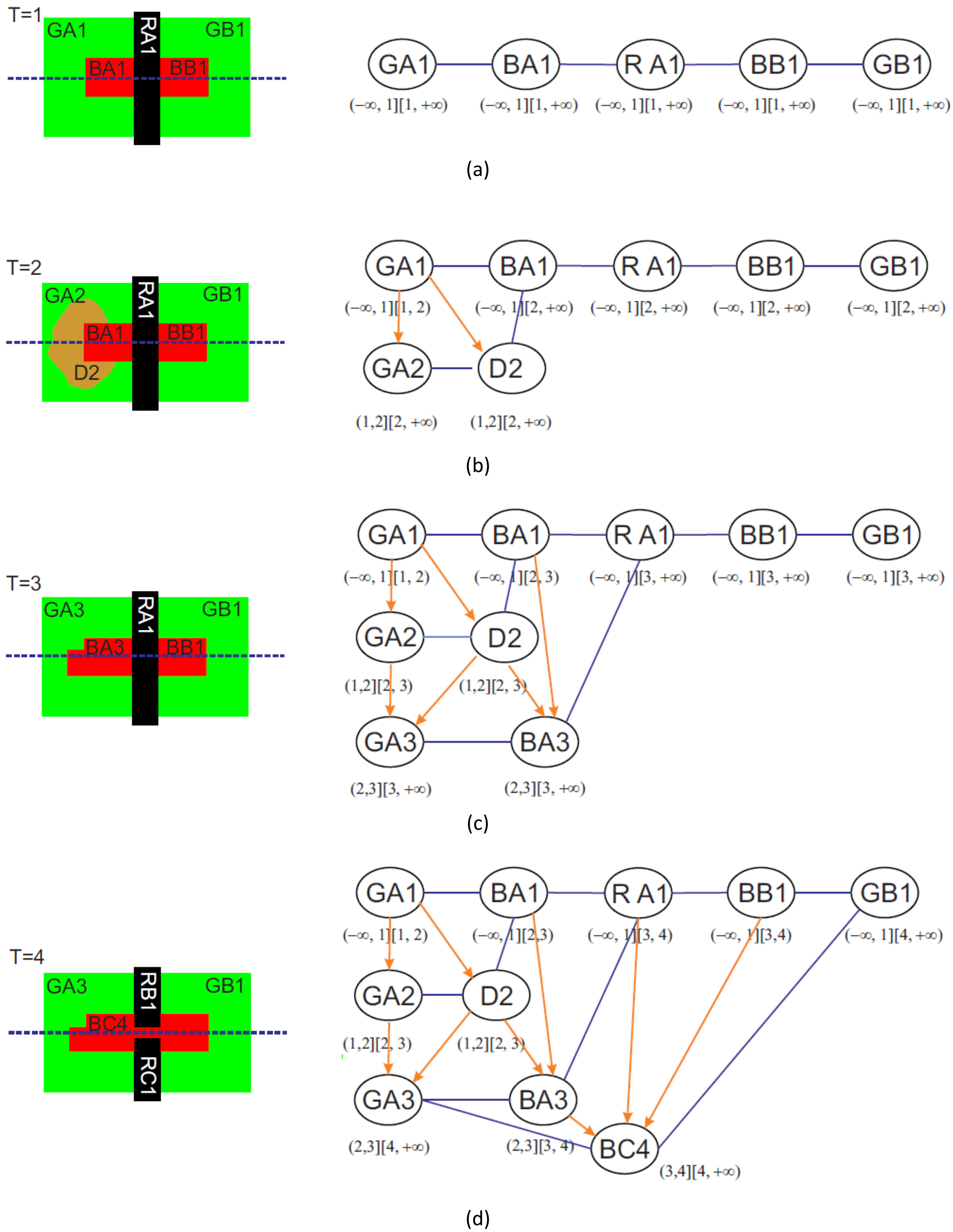


Figure 4. Example temporal graph construction.

However, even the simple graph in (a) is different than a pure geospatial graph, since each node also includes time chronology information, shown by the $(-\infty, 1][1, +\infty)$ attribute. This attribute encodes what is known about the existence of the node over time, given data observations seen so far. The fields of this attribute are $(t_{last.absent}, t_{first.seen}]$ $[t_{last.seen}, t_{first.absent})$. Time $t_{last.absent}$ is the time of the latest observation in which the node was seen not to exist. If there are no such observations, then we set this value to $-\infty$, since as far as we can tell from our data, the corresponding object has been there since the beginning of time. Time $t_{first.seen}$ is the time of the earliest observation when the node was seen. Time $t_{last.seen}$ is the time of the latest observation when the node was seen. Time $t_{first.absent}$ is the time of the first observation after the node was seen, in which the node was observed to have changed or disappeared (either of which means that this node's time span is terminated). If we have not seen an observation indicating the node's disappearance or change, then we set this value to $+\infty$, since as far as we can tell, the node will persist forever.

This representation supports detailed time analysis, under a simple assumption that if a node is seen at time t_1, and then is again observed at a later time t_2, then then node persisted unchanged between times t_1 and t_2. This assumption is reasonable for durable nodes such as buildings. Under this assumption, this attribute gives us a precise representation of the node's history, up to the limits of available data. For example, suppose we have processed eight observations, and an example node has chronology attribute $(2, 3][8, +\infty)$. Suppose we ask: Did the node exist at time 1? The answer is no, because we saw at time 2 that it had not yet appeared. Did the node exist at time 5? Yes; it was seen at earlier time 3 and later time 8. Did the node exist at time 2.5? We cannot know for sure; it was seen not to exist at time 2, and seen to exist at time 3, so it may or may not have appeared by time 2.5. Does the node exist now? We can't be sure, but it was last seen at time 8, and we have not seen any data that provides a counterexample.

Returning our attention to the simple graph in Figure 4(a), we set that all nodes have an initial chronology attribute set to $(-\infty, 1][1, +\infty)$, consistent with a new node supported by only one observation, and which does not supplant any previously-seen nodes.

In the second time slice $t = 2$ (b), new nodes are added to describe the observed change. In this case a node has been added to represent the new bare dirt, and a node has also been added to represent the new grass shape. The grass in a new shape is considered a new node; the time span of the grass in its initial shape is considered ended. The original graph of time slice (a) is now updated to include the new nodes, plus orange edges indicating change relationships.

The time attributes of all nodes are updated as required to reflect what is known after time $t = 2$. Unchanged nodes have their chronology attribute extended to $(-\infty, 1][2, +\infty)$, indicating that these nodes are assumed to have existed continuously at least from time 1 through time 2. The initial grass node GA1's chronology is updated to $(-\infty, 1][1, 2)$, indicating that it was seen at time 1, but had disappeared by time 2. The new orange change edge from GA1 indicates what happened to it: it changed into nodes GA2 and D2. These nodes have associated chronology attributes to $(1, 2][2, +\infty)$, indicating that they were seen at time 2, were known not to exist at time 1, and so far we have seen no counterexample to their continued existence.

The remaining time slices $t = 3$ (c) and $t = 4$ (d) show similar extensions to the graph in response to observed changes.

The graph design shown in Figure 4 forms the basis of our current ongoing work developing semantic graphs with temporal analysis. We are currently developing solutions to a number of important algorithmic issues, such as efficient identification of corresponding nodes, change analysis, and proper handling of sensor noise and missing data. This work is in progress, and will be reported in a later document.

The following sections will describe our work applying semantic graph techniques to SAR data. This work focused on geospatial graphs, without time information. This was due to two reasons: First, our temporal graph analysis code is not complete, and second, suitable SAR data spanning multiple time slices was not available.

3. SAR FEATURE ANALYSIS

The first step in semantic graph analysis is to convert raw image data into discrete features, suitable for encoding in a graph. Our goal in this study is to apply semantic graph analysis techniques to SAR image data. Consequently, the conversion of SAR image data to discrete features is a key first step.

Our group's focus is on semantic graph representations and search algorithms; the image feature recognition required as pre-processing is generally outside the scope of our primary effort. To obtain the required features recognized from SAR data, we contracted Mr. Jarlath O'Neil-Dunne, who is the Director of Spatial Analysis Laboratory at the University of Vermont (UVM). Mr. O'Neil-Dunne brings several strong qualifications to this work. He has successfully performed geospatial feature recognition from remote sensing data for a variety of problems, producing very high-quality results. In addition, Mr. O'Neil-Dunne is a former Marine Corps officer. He served is key leadership positions in several geospatial units in which the analysis of SAR data was performed, including during Operation Iraqi Freedom. This combination of experience made Mr. O'Neil-Dunne an excellent choice for engaging this work.

This section will describe Mr. O'Neil-Dunne's initial results in recognizing static terrain features in SAR image data. To maximize understanding, we will first present a summary of his methods for recognizing terrain cover features in other data. As we shall see, this summary will provide useful context for understanding the SAR feature recognition algorithm and results, described in the sections that follow.

3.1. Terrain Categorization Using LIDAR and RGB-IR Image Data

Mr. O'Neil-Dunne provided terrain categorization data for another Sandia project applying semantic graph analysis to geospatial data (Strip and Watson 2011). In this study, RGB+IR imagery was combined with LiDAR and GIS road data to obtain a detailed model of land cover for Anne Arundel County, Maryland. This data was originally collected and analyzed to measure tree cover for urban and county tree canopy monitoring programs.

In this section we will review Mr. O'Neil-Dunne's method for constructing a land cover model for Washington, DC. These data were provided to support a current project aiming to perform semantic graph temporal analysis, funded by the DOE NA-22 program. This analysis may be summarized by the following problem statement:

> Given input overhead imagery (RGB+IR), LiDAR, and GIS road network data, develop an accurate model of the location and extent of the terrain cover categories: building, trees, grass/shrub, bare earth, road, other paved, and water.

This is an important problem in geospatial imagery analysis, and Mr. O'Neil-Dunne has achieved good success for a variety of locales. We will summarize his method in the paragraphs that follow; see (O'Neil-Dunne, et al 2012) for a fully detailed description. Other researchers have achieved similar successful results using different techniques (Kluckner and Bischof 2010).

The inputs to this procedure for Washington, DC are shown in Figure 5 and Figure 6. The images in Figure 5 are both raster files. The left image in Figure 5 shows an orthorectified mosaic of RGB+IR images, represented as 1 m square pixels and clipped to the DC boundary. These aerial images were captured in the summer of 2011, during which time the deciduous trees had their leaf on. The right image in Figure 5 shows an elevation map, indicating the elevation of the highest terrain feature for each 1 m square pixel. This digital surface model (DSM) depicts LiDAR data collected in 2008, during a season when leaves were off the trees. Figure 6 depicts vector data outlining roads throughout the DC area. The road models used for this computation were released in 2011.

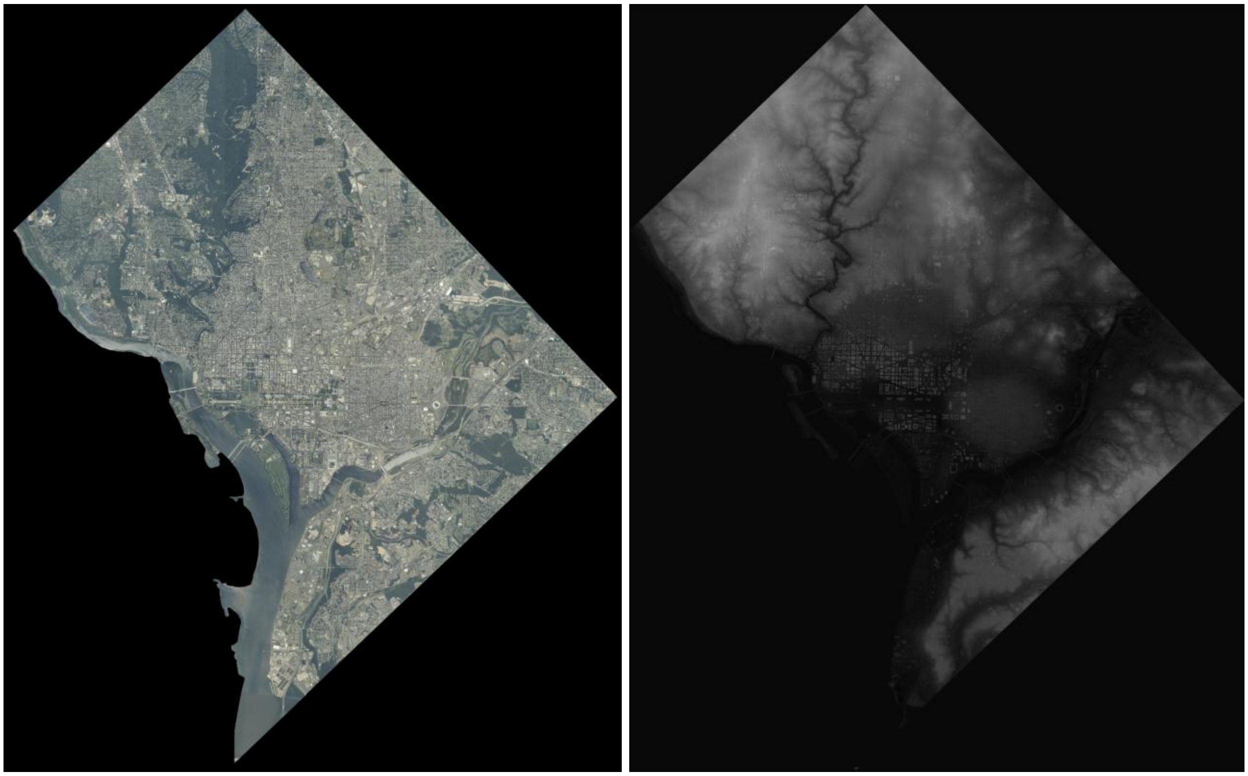


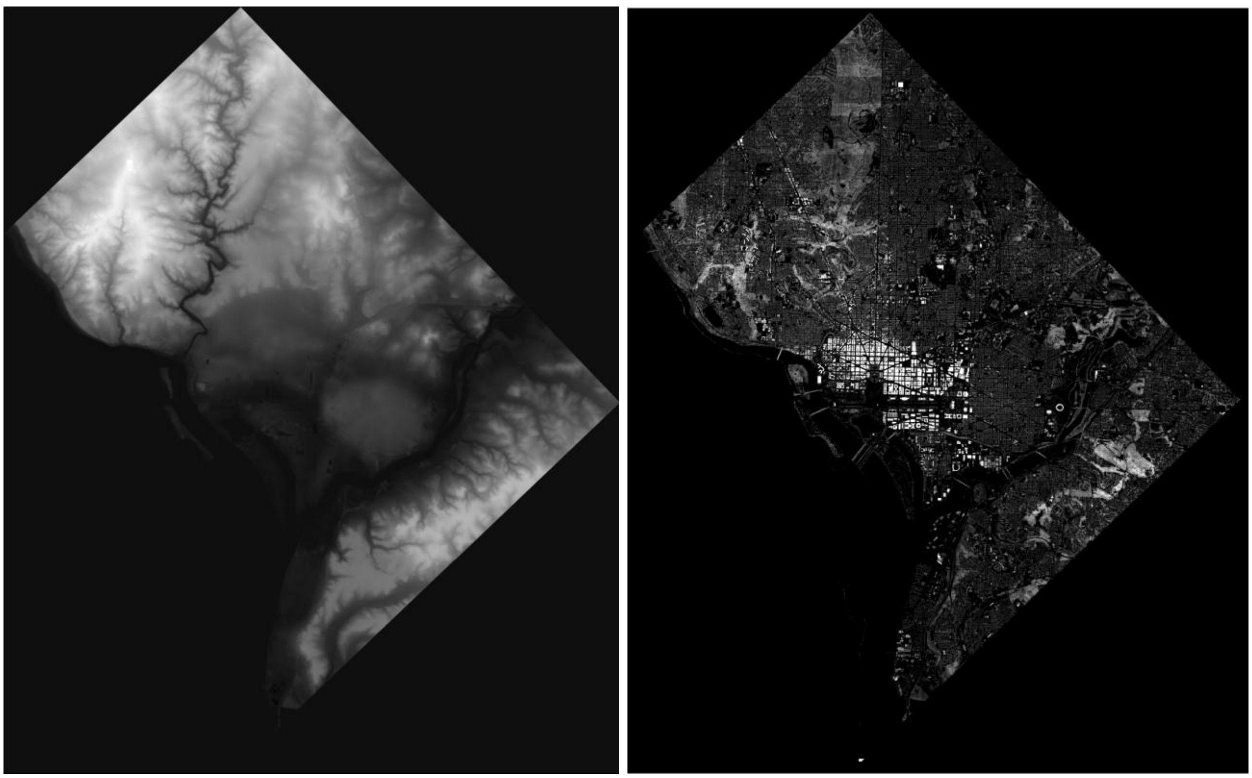
Figure 5. Input: RGB-IR image and digital surface model (DSM).

Figure 6. Input: Road polygons.

The first step in the construction is to convert the digital surface model (DSM), showing elevations above sea level, to a normalized digital surface model (nDSM), showing heights above local ground level. This is accomplished in two steps:

1. Construct a digital elevation model (DEM), showing elevation of the ground above sea level. First, LiDAR points corresponding to "ground" are classified and then interpolated to construct a model of the ground that would occur if there were no buildings, trees, or other tall structures. There are a number of techniques for solving this problem; see (Evans and Hudak 2007) for an example.
2. Subtract the DEM from the DSM, to obtain a model of the net height of the top-most surface above the ground. This is the normalized digital surface model (nDSM).

Figure 7 shows the result of this calculation for the DC data set.



(a) (b)

Figure 7. Digital elevation model (DEM) and normalized digital surface model (nDSM).

The RBG+IR image, nDSM, and road network data may then be used to obtain the desired terrain categorization. This algorithm may be summarized as follows:

1. Use the nDSM to identify tall objects.
2. Among the tall objects, differentiate trees from buildings using either the normalized difference vegetation index (NDVI) computed from the RGB+IR image, or if LiDAR point cloud data is available, by using the standard deviation of the *z* values (see O'Neil-Dunne, et al 2012 for details).
3. Among the remaining ground objects, differentiate grass/shrub areas using NDVI.

4. Among the remaining objects, differentiate bare earth, pavement, and water regions based on spectral, geometric, and contextual properties.
5. For those pixels marked as pavement, differentiate roads and other paved using the GIS road shapes.
6. Apply morphological and context-based routines to refine the landcover classification using spatial characteristics.

This calculation produces a terrain cover categorization that partitions the input space. This result is shown in Figure 8.

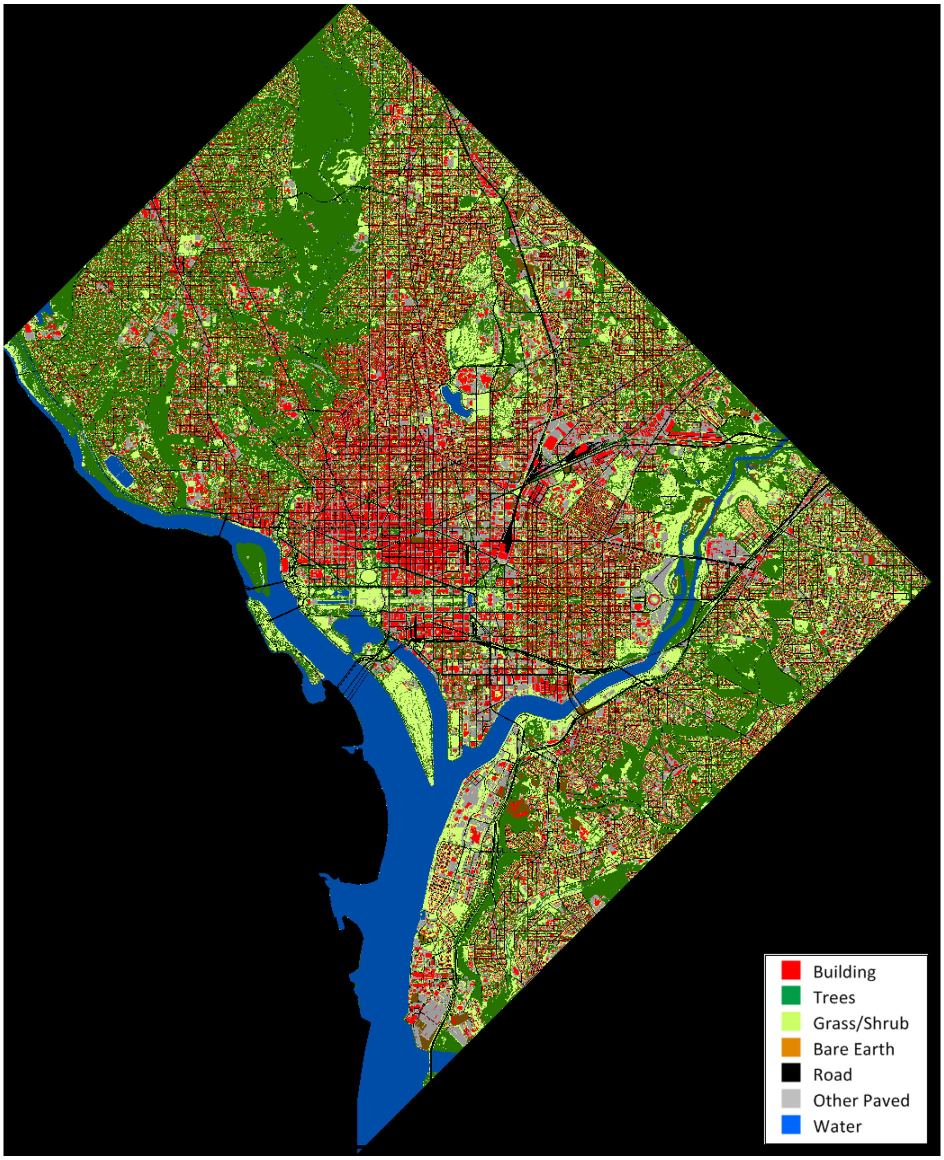
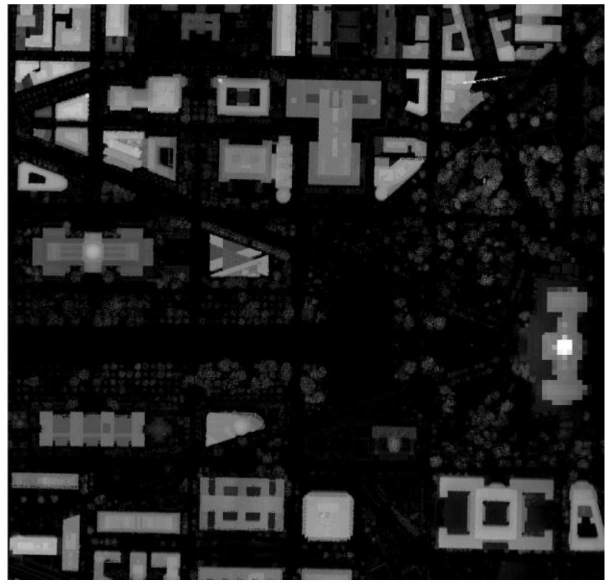
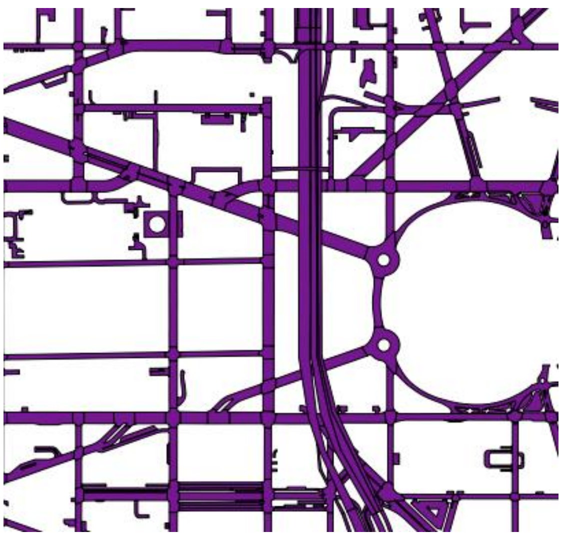


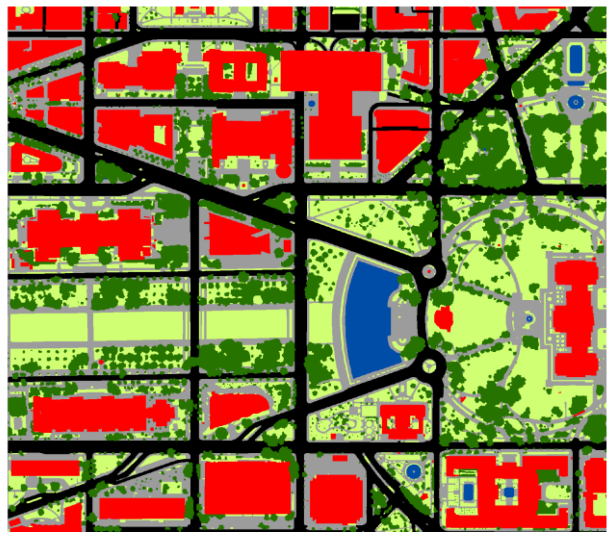
Figure 8. Resulting terrain cover category map.

This terrain cover map is represented by a run-length encoded GeoTiff file with 1 m square pixels. Figure 9 shows a zoomed-in view of the input image, nDSM, input road geometry, and output terrain cover map for the vicinity of the U.S. Capitol.



(a) (b)





(c) (d)

Figure 9. (a) RGB-IR image. (b) nDSM. (c) Road polygons. (d) Land cover.

Using well-accepted techniques for measuring the accuracy of remotely sensed data (Congalton and Green 2009), the UVM group has determined that the final terrain land cover data set has both a producer's and user's accuracy of better than 95% for the tree canopy category that was the focus of their study. See (O'Neil-Dunne, et al 2012) for details.

Note that in urban areas, vector GIS building polygons may be available, eliminating the need for using LiDAR for identifying these features. But in sparsely populated areas such polygons are often not available or incomplete, and so LiDAR is used to map buildings in these areas. In either case, LiDAR is used to measure tree canopy, for which no up-to-date data are available.

3.1.1. Comments

The successful terrain categorization achieved by the UVM group for Washington, DC and other areas indicates what is possible when sensors, algorithms, and desired semantics are well-matched. The UVM terrain categorization algorithm is designed to include information from multiple sources, exploiting the characteristics of each data source to extract the information that the source is best able to discriminate. For example, the (x, y, z) values of the LiDAR nDSM provide a robust method for distinguishing tall features from ground, and if the original LiDAR point cloud is available, for distinguishing porous tree canopy from solid building roofs. LiDAR contains some spectral information in the form of intensity data that can be used to differentiate ground cover types such as grass vs. pavement. Unfortunately techniques for the tonal balancing of LiDAR intensity have not been perfected, limiting the utility of this information. RGB-IR data provides a rich spectral source to differentiate live vegetation from non-living materials. Similarly, the GIS data enables distinguishing roads vs. parking lots, which are defined by subtle differences that are difficult to discern using automated methods alone.

The UVM group is well versed at designing algorithms that improve recognition performance by combining the best capabilities of multiple data sources. During our initial discussions with Mr. O'Neil-Dunne, we sketched a similar study that would combine SAR image data captured over multiple time frames with other data modalities, producing a representation that would enable semantic analysis and change detection.

However, during the process of identifying test data to send to UVM, this goal changed into a much more narrow definition, focusing only on static feature recognition, and based only on SAR data. Further, suitable test data obtained at multiple times were not available. These changes limited the methods that UVM could bring to bear on this problem.

When we explained this narrowing of scope to Mr. O'Neil-Dunne, he offered this description of his experience using SAR data while interpreting remote sensing data in support of combat operations in Iraq:

> "Rarely, if ever, is a single SAR image used for feature recognition. A given feature has the potential to have very different characteristics in a SAR image based on acquisition properties. Typically the interpretation of SAR imagery is carried out in conjunction with optical imagery. Even if the optical image is acquired at a different time period, it can provide important indicators to guide the interpretation of the SAR imagery. SAR is most useful for change detection. Examples include coherent change detection (CCD) and moving target indicators (MTI). Once again, such analysis is typically carried out in conjunction with some reference optical imagery to guide the interpretation."

This challenge was compounded by a very short time window for recognition algorithm development, resulting from delays that accumulated from multiple causes. Nonetheless, UVM engaged this problem with vigor, and developed a preliminary feature recognition algorithm which enabled initial semantic graph studies. The following sections will describe the SAR data that was provided, and UVM's algorithm for identifying terrain cover categories in the resulting image.

3.2. SAR Data for Study

The selected SAR test data covered a roughly square area, 6.3 km × 6.6 km, in Valencia County south of Albuquerque. This region is shown in Figure 10, and includes a mixture of barren, rural, semi-rural, and sparse suburban areas. The image set was captured in December 2011, a time when most trees had lost their leaves.

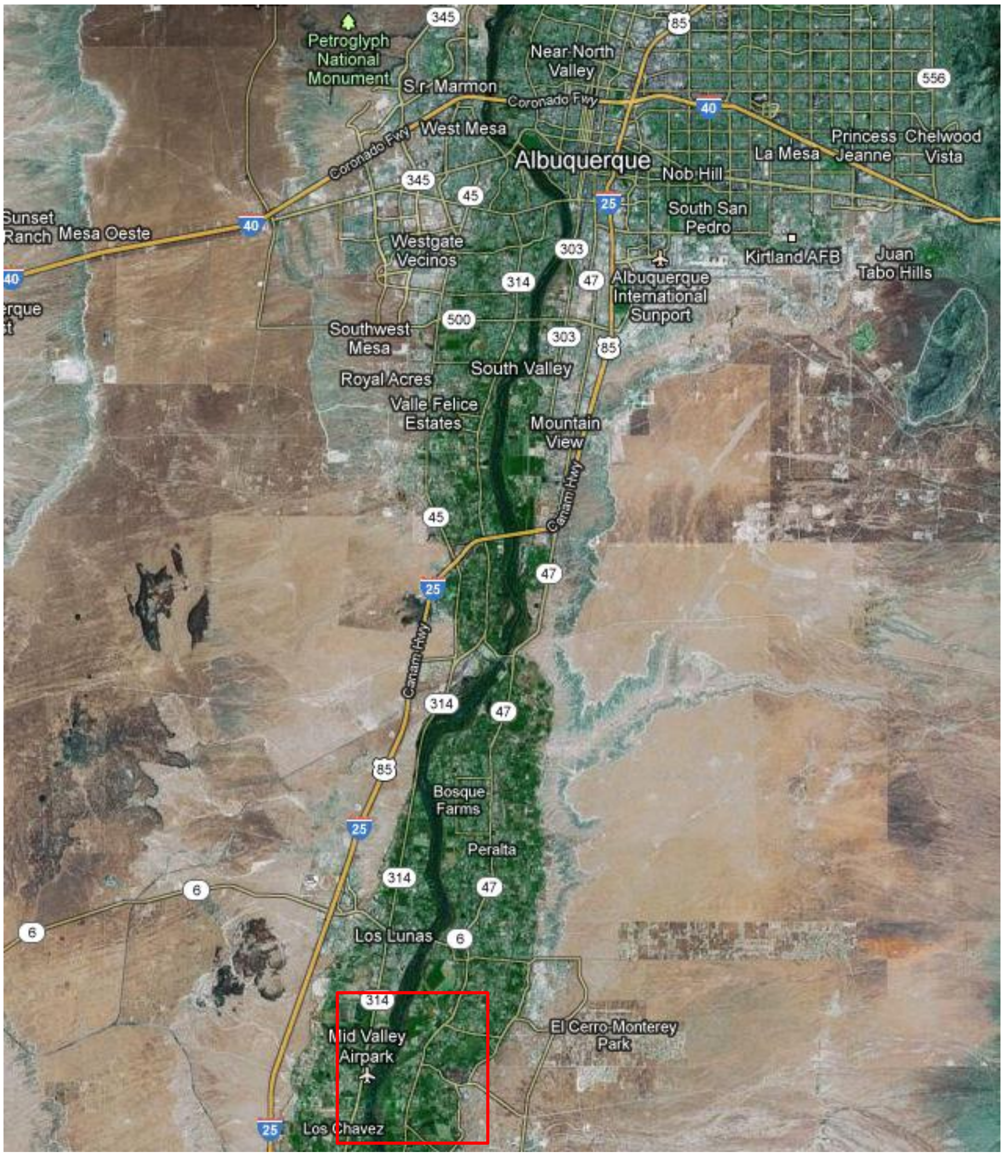


Figure 10. Sample region for test SAR data, south of Albuquerque.

The SAR images were captured by an aircraft flying overhead in 10 east-west passes. The sensor was looking north, so SAR shadows appear on the north side of tall objects. In each pass, 27 image patches were captured. The 10 passes were completed in roughly two hours' time.

Figure 11 shows several example image patches. Each image patch is roughly 250 m × 700 m, represented in a 4 MB GeoTiff file with 1437 × 3389 pixels. Thus each patch is approximately 5 Mpixels, and the pixel size is roughly 17 cm × 21 cm.

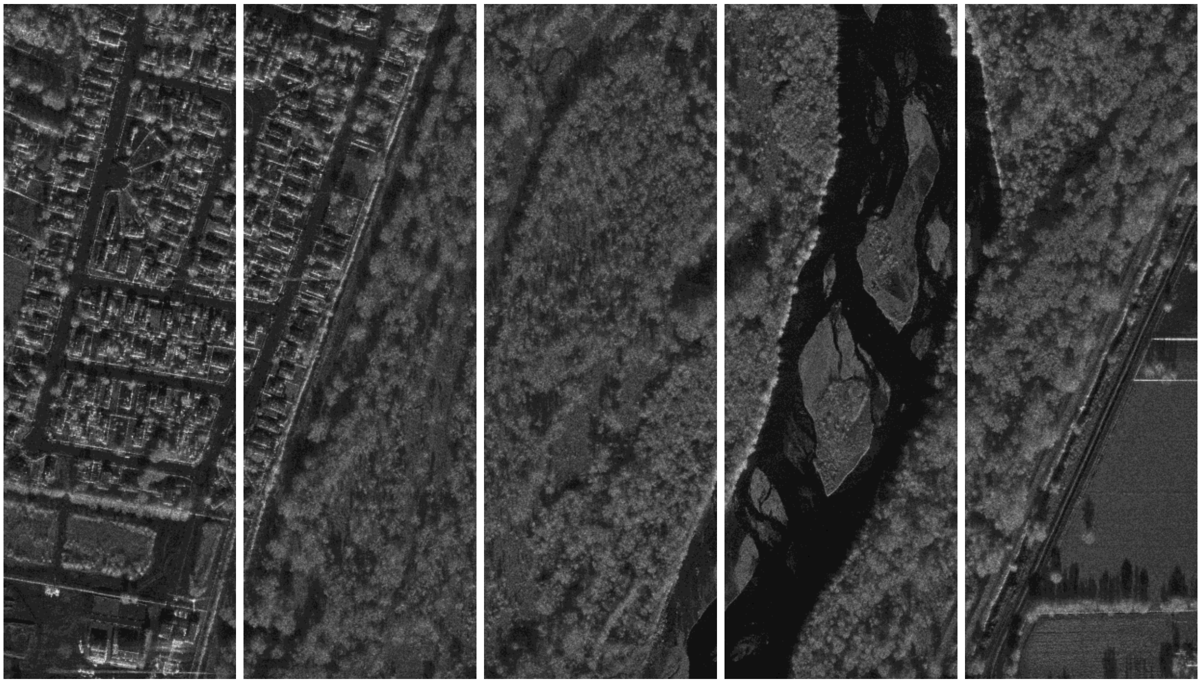


Figure 11. Example SAR image patches.

Figure 12 shows the layout of image patches that cover the region shown in Figure 10. After obtaining appropriate approvals, Sandia transmitted this collection of image patches to UVM, along with a few explanatory documentation files.

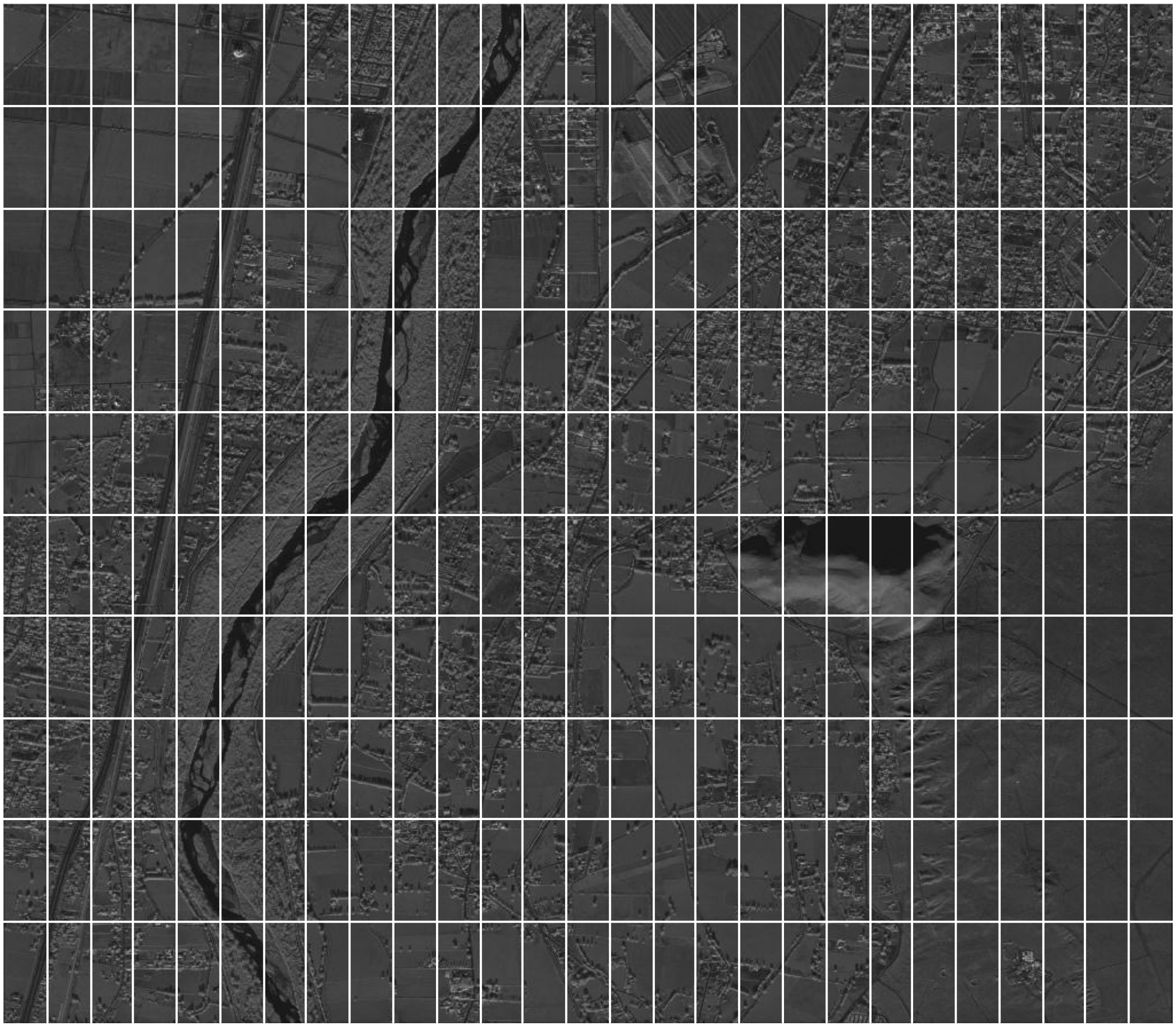


Figure 12. Ensemble of patches that form the mosaic.

Mr. O'Neil-Dunne combined these patches to form a single mosaic image, shown in Figure 13. The SAR tiles overlapped approximately 14% in each direction. The mosaic process used weighted seam lines, in which cut lines that minimize adjacent pixel difference are generated, to handle the overlap. Weighted seam lines were automatically generated using the ERDAS IMAGINE 2011 software package. This image mosaic spans 6.3 km × 6.6 km, and has 33,848 columns and 29,524 rows. This 1 Gpixel image is about 1 GB in file size.

The mosaic image shown in Figure 13 was the primary input to Mr. O'Neil-Dunne's feature recognition algorithm.

Figure 13. Mosaic SAR image.

3.3. SAR Feature Recognition

This section will explain the feature recognition algorithm, and then evaluate its performance.

3.3.1. Method

UVM's approach to developing the recognition algorithm was first to study the image data manually to determine what features could be reliably identified by a human analyst, and then to develop an algorithmic approach to finding those features. The automated code was developed using eCogntion®, a tool for developing image analysis and recognition procedures which is designed for object-based image analysis (OBIA). Unlike more traditional pixel-based approaches that only make use of spectral information, OBIA allows spectral, textural, geometric, and contextual information to be incorporated into the feature extraction process (eCognition 2012).

This effort produced an eCognition rule set that accepts an input mosaic SAR image, and partitions it into terrain cover regions labeled with the following cover classes:
Water, Tall, Road, Shadow, and Ground.

The algorithm encoded by this rule set is summarized below:

1. Break input mosaic image into tiles, to allow parallel processing.
2. For each tile:

   - Recognize features:

     A. Recognize dark pixels corresponding to water features.
     B. Of remaining, recognize bright pixels corresponding to tall features.
     C. Of remaining, recognize road shapes and dark shadow features.
     D. Assign all remaining pixels to ground features.
3. Stitch tiles back into a single mosaic map with categories Ground, Road, Shadow, Tall, and Water, and output to a GeoTiff file.

The basic structure of the algorithm is to break the image into tiles, apply feature recognition to each tile, and then re-combine the tiles into a single monolithic result. This enables parallel processing, to reduce total wall-clock time required to perform the computation. The selected tile size was 5000 × 5000 pixels, resulting in the 42 tiles shown in Figure 14.

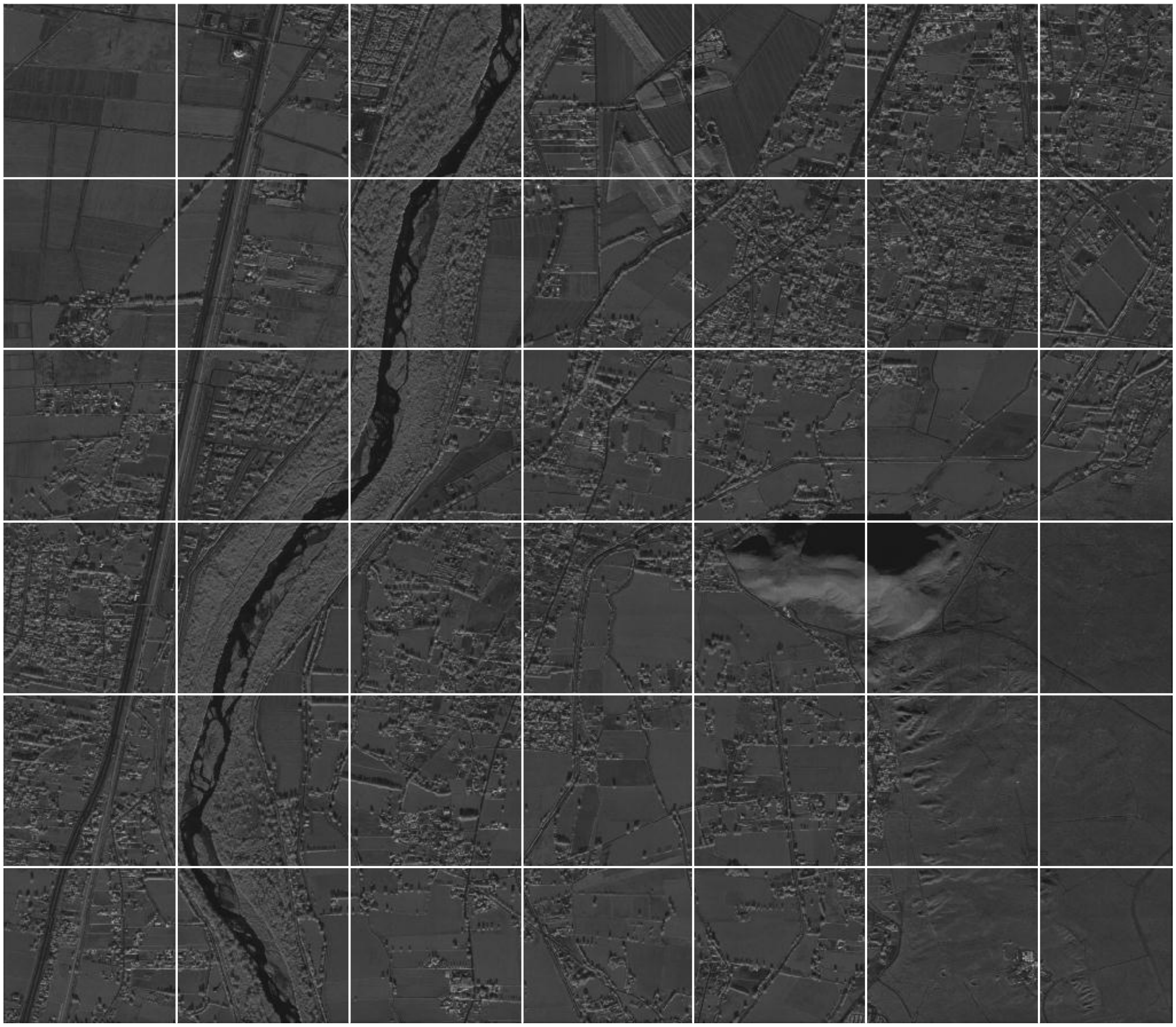


Figure 14. Tiles used for image processing.

Given this overall structure, the primary recognition work is performed by steps A – D within the loop. The following paragraphs will explain the procedure to recognize features within a single tile (step 2).

A. Find Water Features

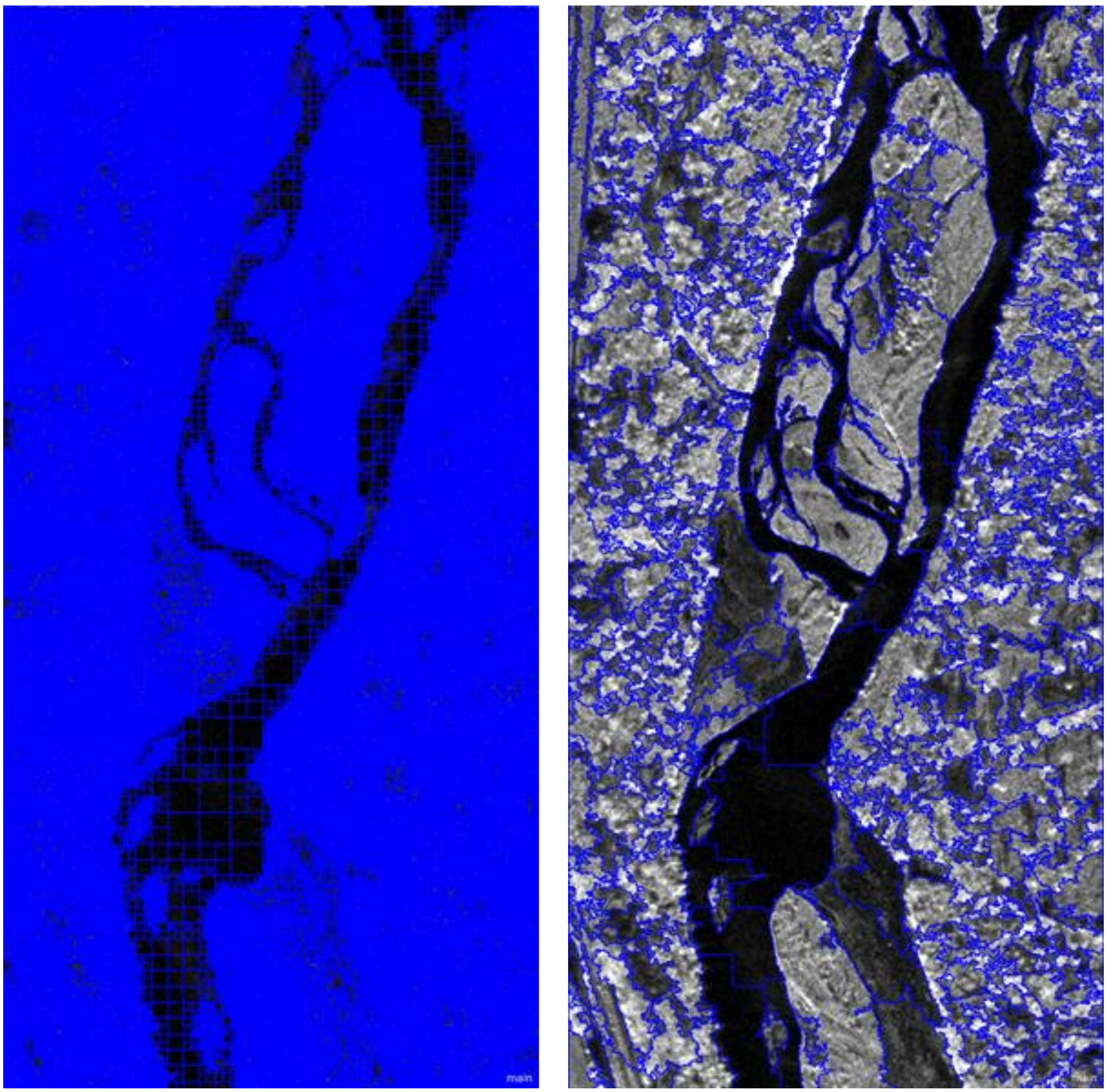
Flat water in SAR imagery appears dark, because the water's specular surface causes most of the radar beam to reflect away from the antenna. Bodies of water thus appear as dark contiguous regions in the image.

To find water features, the algorithm first considers all pixels and applies a general-purpose segmentation algorithm. This segmentation algorithm groups pixels that have similar spectral properties. A quadtree representation of the output is generated, allowing adjacent similar pixels to be represented more efficiently. The result of this procedure for a sample image portion is shown in Figure 15(a).

Next, the algorithm applies a procedure to increase the size of found segmented regions, by merging adjacent regions which have similar mean pixel intensity values, and where the resulting grown region shape is moderately compact, where “compact” is a parameter defined by eCognition that is closely related to the region’s area/perimeter ratio (eCognition 2012).

The algorithm evaluates this merge criterion using a fuzzy logic calculation which weights pixel intensity similarity 90% and shape compactness 10%. (See the road feature analysis below for an example fuzzy logic calculation.)

The resulting segmentation is shown in Figure 15(b). Note that all regions of the image are segmented; so far we have applied no criterion to prefer water pixels over other pixels.



(a) (b)

Figure 15. After (a) quadtree segmentation and (b) multi-resolution grow operations.

The algorithm then identifies water features by finding seed regions, and then growing those regions to include adjacent regions that meet a similarity criterion. The algorithm identifies initial water seeds by selecting all regions with a mean pixel intensity darker than a pre-defined intensity threshold $I_{w.max}$ and intensity standard deviation less than $\sigma_{w.max}$, merging adjacent qualifying seeds, and then discarding results smaller than a pre-set area threshold.

Given the remaining seeds, the algorithm applies an iterative procedure to find regions adjacent to seeds where (a) the neighbor's mean pixel intensity equals the seed's mean pixel intensity within a pre-set threshold, and (b) the border with the seed is at least 15% of the neighbor's border. As with most other criteria in this algorithm, this criterion is evaluated using fuzzy logic. Sub-criteria (a) and (b) each have a degree of match which maps to a likelihood probability; these subcriteria probabilities are combined to produce an overall probability that the criterion is true, and then the algorithm compares this criteria probability against a pre-set threshold to decide whether to merge the neighbor with the seed. This process is applied to all seed neighbors, and the overall procedure is repeated three times, to capture neighbors whose common boundary with water grew as a result of other region merge operations. This computation is illustrated in Figure 16.

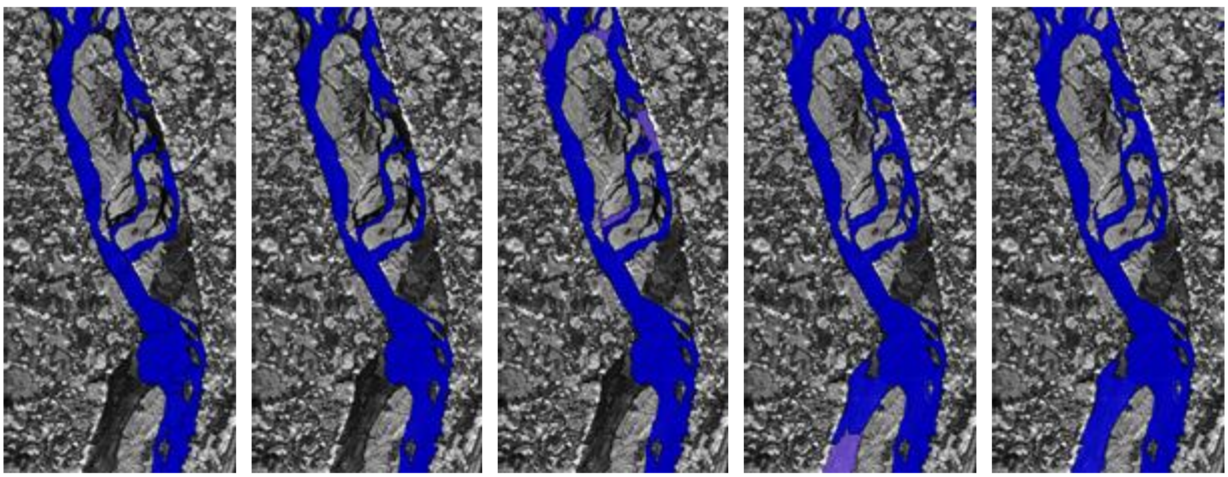


Figure 16. Aggregating regions to recognize water features.

This procedure successfully identifies water regions, but also sweeps up non-water regions as well. To eliminate these undesired regions, the algorithm then filters the results by keeping only those regions whose mean pixel intensity is less than the tighter threshold ($0.8 \cdot I_{w.max}$). The yellow regions in Figure 17 are example regions that fail this criterion. All regions surviving this filter are labeled water and returned by the algorithm.

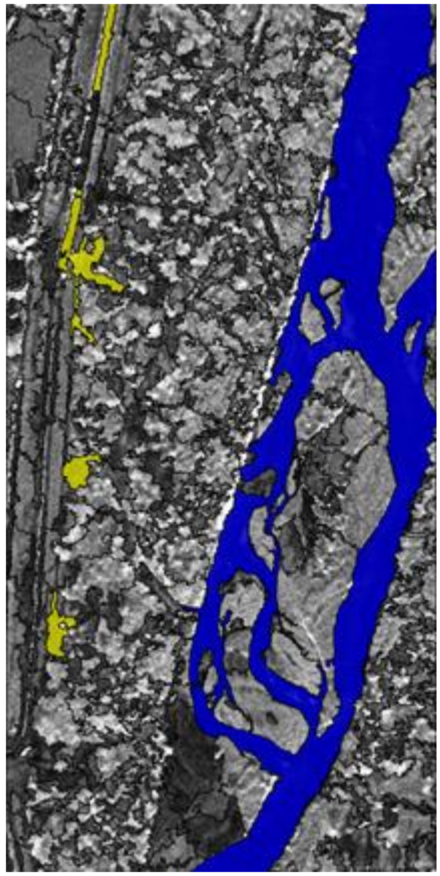


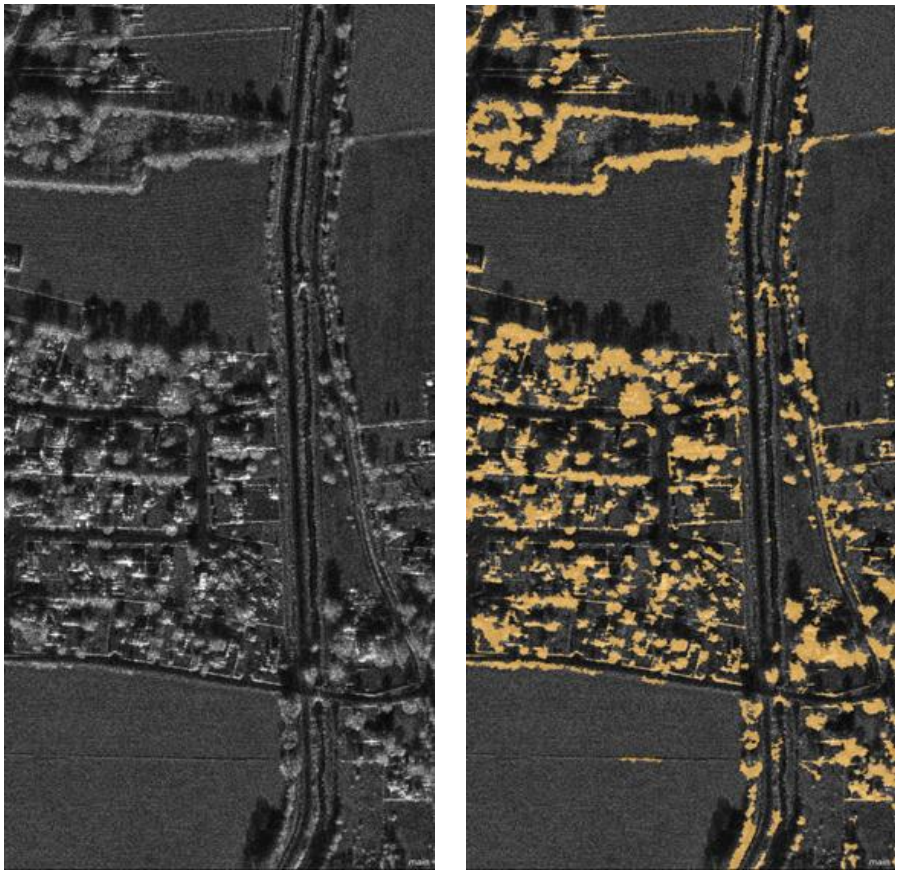
Figure 17. Removing spurious non-water regions.

B. Find Tall Features

After labeling water regions and removing their pixels from further consideration, the algorithm proceeds to seek "tall" features. The tall-object recognition algorithm is based on the assumption that tall objects are characterized by bright reflections distributed in a shape that has significant extent in both length and width. Short objects such as curbs or fence rails can also give rise to bright reflections, so the algorithm includes steps to reject thin bright regions characteristic of these objects.

The first step of the algorithm is to repeat the general-purpose quadtree segmentation and multi-resolution grow operation on the remaining (non-water) pixels. This produces a result similar to the prior segment/grow operation, but slightly different due to different control parameters. Compared to the first segmentation preparing for water region search, the segmentation for tall features uses a size threshold one-fourth the previous size threshold, and weights pixel intensity only 80% compared to the 90% used for water segmentation. In addition, this segmentation requires results with a much higher compactness (area/perimeter ratio).

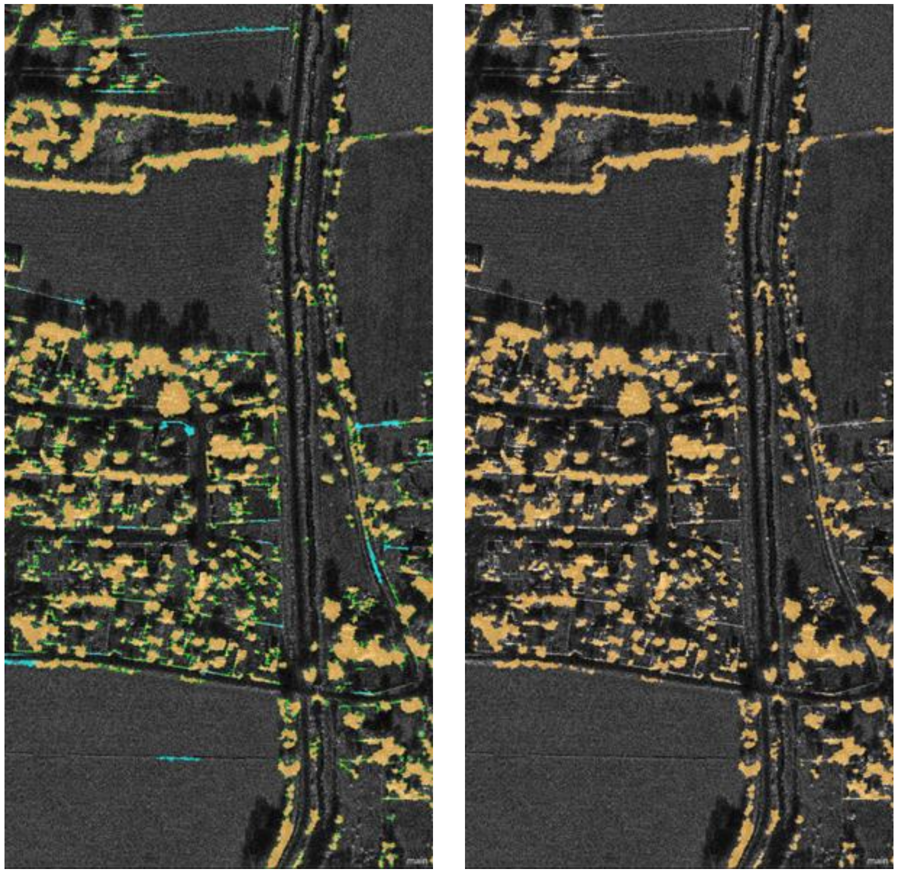
The algorithm then finds candidate tall regions by filtering the segmentation results to retain only those regions with a mean pixel intensity brighter than a pre-set threshold. This result is shown in Figure 18.



(a) (b)

Figure 18. Initial classification of tall features.

The algorithm then applies two methods to find and discard thin, spindly features. The first method finds the subset of the candidate regions whose shape density less than 1.0, where "density" is defined by eCognition to be the number of region pixels divided by the region's approximate radius based on the covariance matrix (eCognition 2012). Then for each such thin candidate, if the thin candidate shares more than 40% of its perimeter with another candidate, then the two candidates are merged; otherwise the thin candidate is discarded. This procedure is repeated twice. The second method takes the remaining candidates and first shrinks their boundaries by five pixels, and then grows the remainder back by five pixels. This has the effect of geometrically removing candidates that are less than 10 pixels wide, and separating large regions connected by a narrow isthmus. Figure 19(a) shows regions removed by the two methods. The surviving features are labeled tall and returned by the algorithm (Figure 19(b)).



(a) (b)

Figure 19. Removing spindly regions from tall feature set.

C. Find Road and Shadow Features

At this point the algorithm has identified water and tall features, and their corresponding pixels have been removed from consideration. The algorithm then searches the remaining pixels for road and shadow features.

The algorithm's first step is to repeat the quadtree/multi-resolution grow segmentation analysis on the remaining pixels, this time allowing small features (as in the tall feature segmentation), and allowing a lower area/perimeter ratio (as in the water feature segmentation).

Road features are then found by identifying segmented regions that meet either of two selection criteria, both of which are computed by fuzzy logic.

The first criterion is illustrated in Figure 20. This figure shows two screen captures of eCognition dialog windows used to define the criterion. The first window lists the sub-criteria and the calculation used to combine them to obtain an aggregate score. The criteria are density and length/width ratio. The second window shows the fuzzy logic mapping of length/width ratio to the subcriterion probability. Length/width ratios less than three score a sub-criterion probability of zero, while length/width ratios greater than six score a subcriterion probability of one. Ratio between three and six score sub-criterion probabilities indicated by the curve shown in the plot. The density sub-criterion performs a similar calculation, and then the overall criterion score is taken to be the minimum of the individual sub-criterion scores.

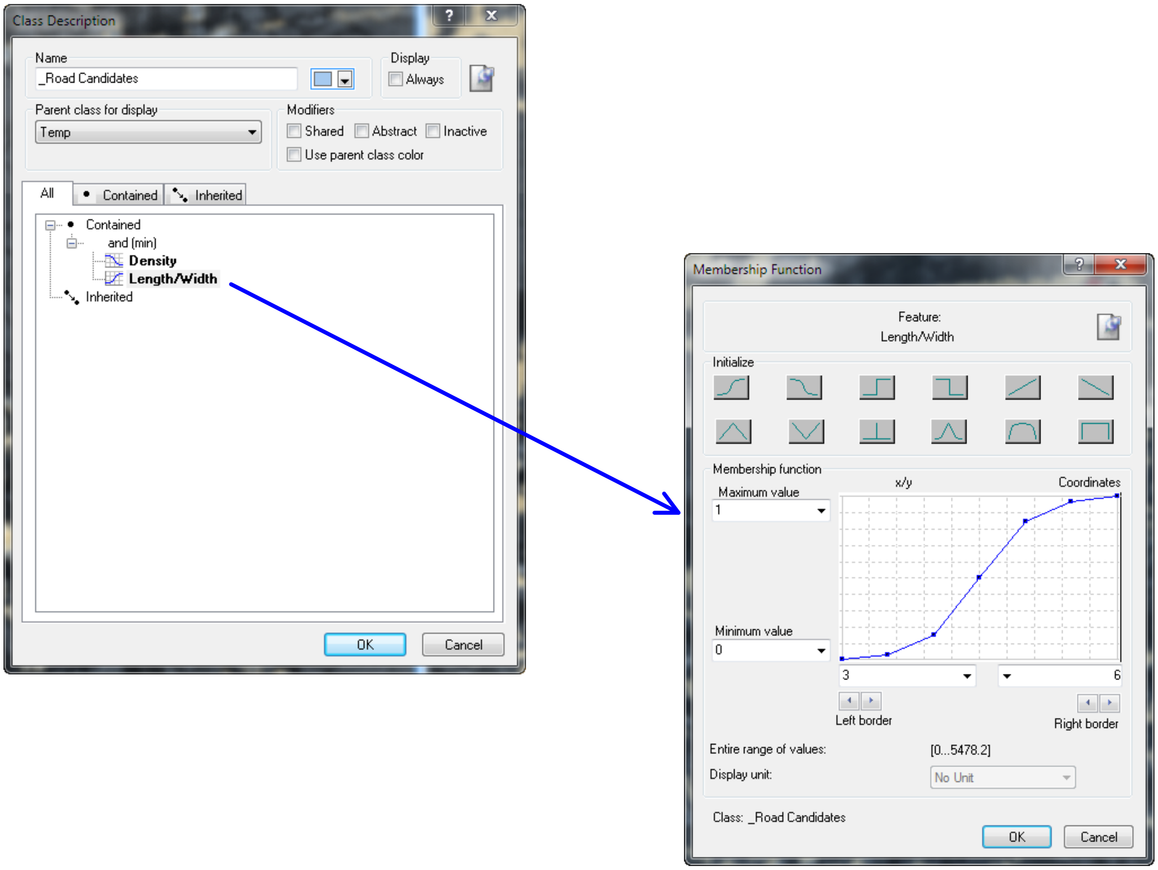


Figure 20. eCognition fuzzy logic definition windows.

Given this criterion definition, regions with a criterion score greater than 0.2 are taken to be road features. A second criterion is also used to identify roads, similar to the first but designed to accept longer and thinner regions. Figure 21 shows the result of this road feature detection algorithm.

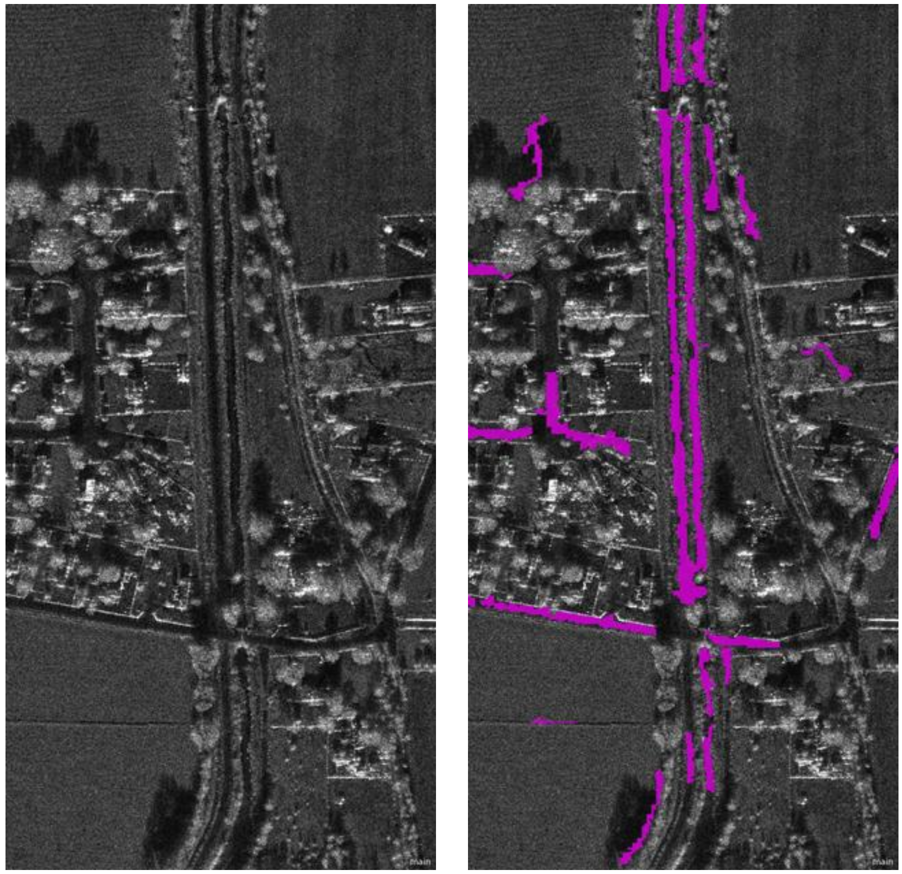


Figure 21. Road detection results.

After removing road features from consideration, the remaining segmentation results are checked to identify shadow features. All regions with a mean pixel intensity darker than a pre-defined threshold are taken to be shadow features.

D. Assign Remaining Pixels to Ground

The algorithm then completes its analysis by categorizing all remaining pixels as ground features.

Finally, merge operations are applied to combine all adjacent regions with identical feature classifications, and the resulting Water, Tall, Road, Shadow, and Ground regions are output to a GeoTiff file.

3.3.2. Results

Given the SAR mosaic image shown in Figure 13 and repeated in Figure 22 for reference, the detection algorithm recognizes features and outputs the terrain cover map shown in Figure 23. This computation was performed in just under two hours, on a single workstation with dual six-core Xeon 3.16 GHz processors, running 14 threads with hyper threading enabled. The number of threads was limited by the number of eCognition licenses.

Figure 22. Full SAR image mosaic, with focus region.

An examination of the results in Figure 23 immediately reveals basic aspects of the recognition output. Water features corresponding to the Rio Grande river are properly identified throughout almost all the entire river length, and numerous tall features with expected adjacent north shadows can be seen. Road features are visible revealing overall road network structure, but gaps are also noticeable. We can also observe errors in feature categorization. The long river feature is interrupted with a misclassified shadow feature, and the large shadow north of the hill in the middle right is misclassified as water.

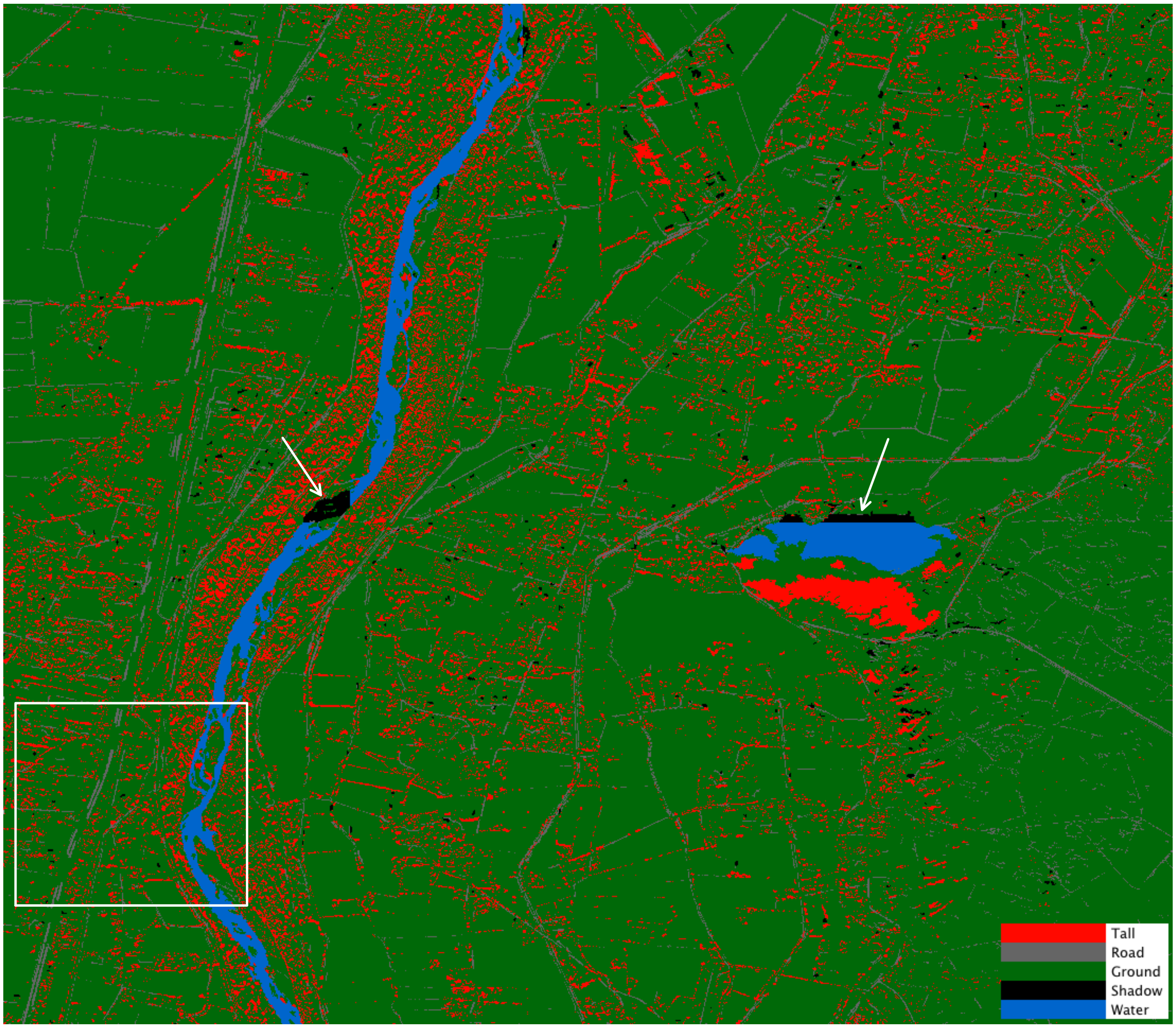


Figure 23. Full terrain cover categorization results.

A better understanding of the recognition capability of the algorithm can be gained by examining zoomed-in views to see greater detail. In the sections that follow, we will focus on the smaller area outlined by the white rectangle shown in Figure 22 and Figure 23.

Figure 24. Focus region SAR image, with red close-up view box.

Figure 24 shows a zoomed-in view of this portion of the input image. (The red close-up view box will be referenced in later sections.) Individual plots of land and the trees and houses they contain can now be seen, along with the network of roads and irrigation ditches and the forested area adjacent to the river.

Figure 25 shows the feature recognition results for this same region. In this section of the image, water features are correctly recognized, and properly distinguished from adjacent sand bars. Road features that correctly correspond to real roads are also seen, but large gaps are evident. We also see narrow features recognized as roads that upon closer examination turn out to be irrigation ditches; when viewed in SAR imagery, these long linear water surfaces look very similar to paved road surfaces. Further examination at an even closer zoom level reveals that many tall features such as trees and houses are properly recognized, while many are also missed; we shall see examples in later sections.

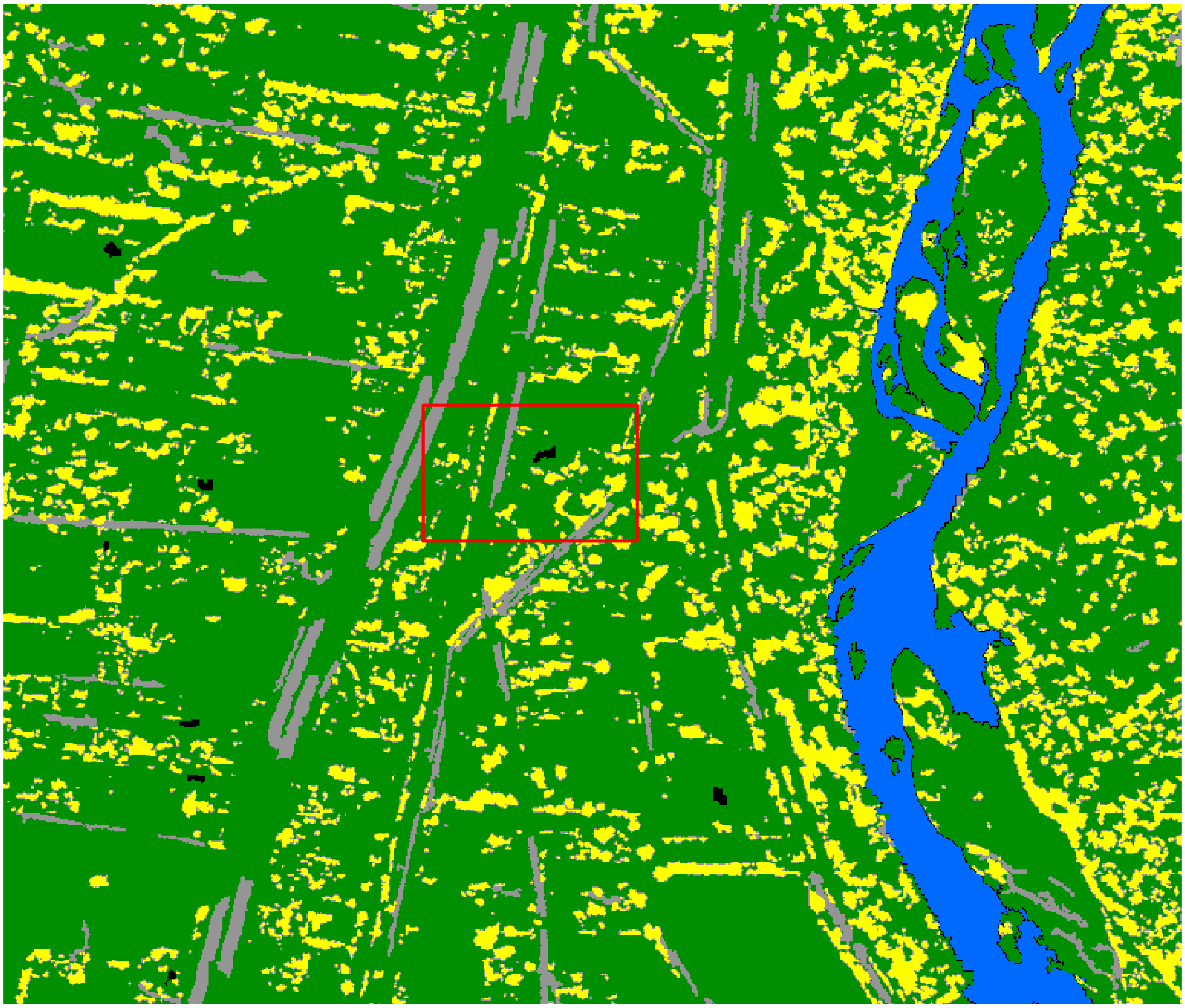


Figure 25. Terrain cover categorization for focus region.

Mr. O'Neil-Dunne provided an objective, quantitative assessment of recognition accuracy. This assessment was performed following the methods described in (Congalton and Green 2009), a standard practice in assessing the accuracy of products derived from remotely sensed data. In this method, 500 random points were generated, and an independent reviewer who has never seen the SAR data or the resulting terrain categorization reviewed each point and determined its proper classification using recent 6-inch RGB optical imagery. These classifications are then treated as ground truth, and compared against the feature recognition algorithm's output for the same points.

The tabulated results are shown in Figure 26. This matrix shows the results of the terrain categorization for all 500 sample points. The vertical column under the heading “Total” shows all of the ground-truth features. For example, the 500 sample points contained 91 points that were classified Tall by the independent human analyst. Of these 91 points, 34 were classified Tall by the algorithm, one was classified Road, 55 were classified Ground, and one was classified Water.

	Tall	Road	Ground	Shadow	Water	Total	**Producer's Accuracy**	
Tall	34	1	55		1	91	37%	
Road		6	17			23	26%	
Ground	15	10	340	1	2	368	92%	
Water		1	2	1	14	18	78%	
Total	49	18	414	2	17	500		
User's Accuracy	69%	33%	82%		82%		**79%**	**Overall Accuracy**

Figure 26. SAR recognition error matrix.

The values in the matrix are then used to compute the producer’s and user’s accuracy for each terrain cover category. The producer’s accuracy reports the recognition algorithm’s capability to find features. For example, of the 91 points that were truly Tall, the algorithm found 34, resulting in a producer’s accuracy of 34/91 = 37%. The user’s accuracy reports the reliability of the found features. For example, the algorithm found 49 Tall features, and 34 of these were truly Tall, yielding a user’s accuracy of 34/49 = 69%.

The producer’s accuracy column and user’s accuracy row report the producer’s and user’s accuracy for each terrain cover category. Note that some cover types, such as Water, were found much more reliably than others, such as roads. The Ground category had the highest producer’s accuracy, resulting at least in part from the preponderance of area in the input image that corresponded to this broad terrain cover class.

Overall accuracy is measured by summing the total number of correctly identified features summed along the matrix diagonal, and dividing this by the total number of sample points. The resulting high value of 79% is heavily influenced by the large number of points classified as Ground. Note that Shadow features were excluded from the calculation, because from Mr. O’Neil-Dunne’s perspective these are always viewed as categorization errors, since Shadow is not an intrinsic terrain cover characteristic.

This method of assessing errors was also used by the UVM group to assess other terrain cover recognition algorithms with much higher accuracies; see (O’Neil-Dunne, et al 2012) for an example.

The recognition code includes many “magic numbers” that serve as algorithm control parameters and decision thresholds. The sensitivity of recognition to the values chosen for these parameters was not assessed.

4. SEMANTIC GRAPH MODELS OF SAR DATA

This section will describe our work to apply semantic graph analysis techniques to the SAR feature recognition results described in the previous section. Budget constraints limited this to a preliminary study. In the sections that follow, we will describe our construction of a semantic graph from the recognized features, and the outcome of a single template search performed on the resulting graph.

4.1. Graph Construction

As explained above, we applied pure geospatial semantic graph analysis to the recognized SAR features, rather than a geospatial-temporal analysis. This was accomplished using the geospatial graph construction code reported in (Strip and Watson 2011). This code is actually a set of utilities which convert an input GeoTiff file through a set of intermediate representations, ultimately producing a graph represented as an ensemble of csv files. This program suite was written to generate initial proof-of-principle results, and is not optimized for speed, memory, or disk usage.

Given the input terrain cover map shown in Figure 23 and a maximum inter-feature edge length of 560 m, the code generated a geospatial graph containing 52,018 nodes and 10,288,678 edges. Here is the breakdown of node types:

Tall	46,404
Ground	2,668
Road	2,620
Shadow	323
Water	3
TOTAL	52,018

The resulting files were 8 MB for the graph nodes, 309 MB for the edges, and 224 MB for the node polygons. (After graph construction, the polygons are only used for rendering purposes.)

When computed as a single monolithic graph as described above, graph construction required 48 hours from start to finish on a Mac workstation with dual 6-core Intel Xeon CPUs running at 2.66 GHz. However, the code allows partitioning the input into tiles to support parallel processing, enabling significant speedup. After instructing the code to partition the input into a 4×4 matrix of 16 tiles, graph construction was completed in 2.3 hours. These tiles overlap slightly, which changes the number of graph nodes and file size. We report the case of the single-tile run here, since the resulting monolithic graph was used to generate the search results reported below.

Figure 27 and Figure 28 show partial visual renderings of the constructed graph, for the close-up view shown by the red box outline in Figure 24. These are partial views, because not all edges are shown. In Figure 27, only edges with edge length attributes up to 40 m are shown, while in Figure 28, edges with lengths up to 80 m are shown. Since edges are computed with lengths up to 560 m, drawing all graph edges would result in a very cluttered picture, even for the small number of nodes shown in this close-up view.

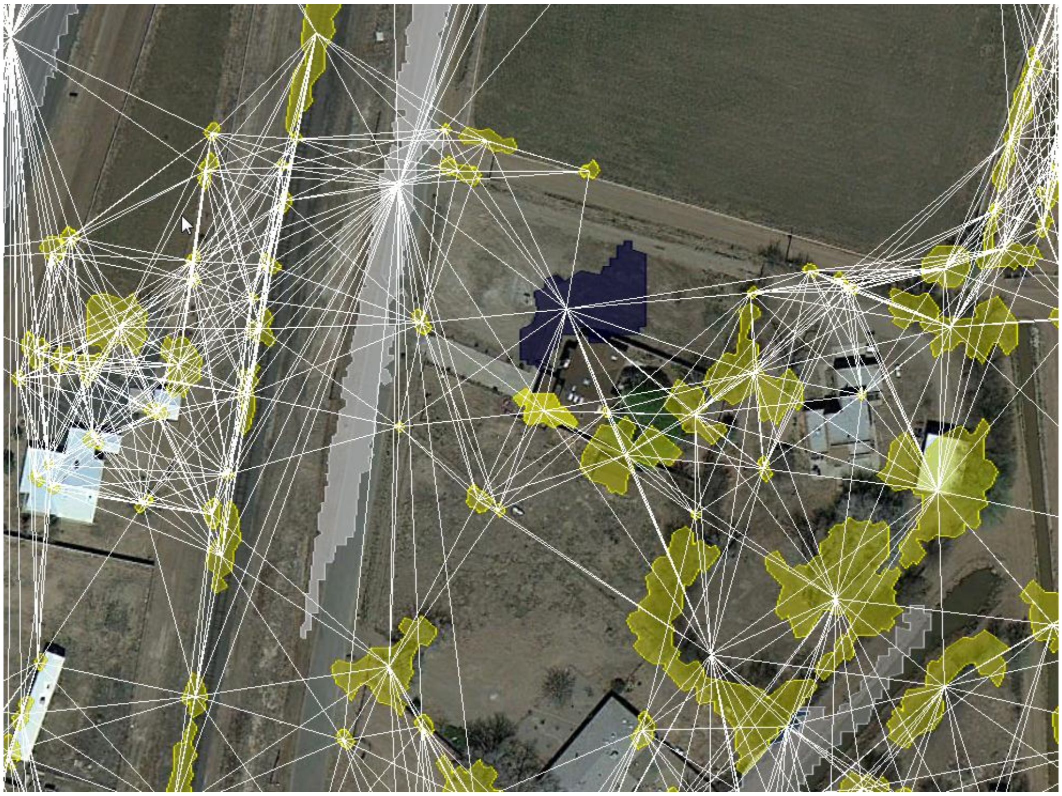


Figure 27. Close-up view of graph, with edge lengths up to 40 meters.

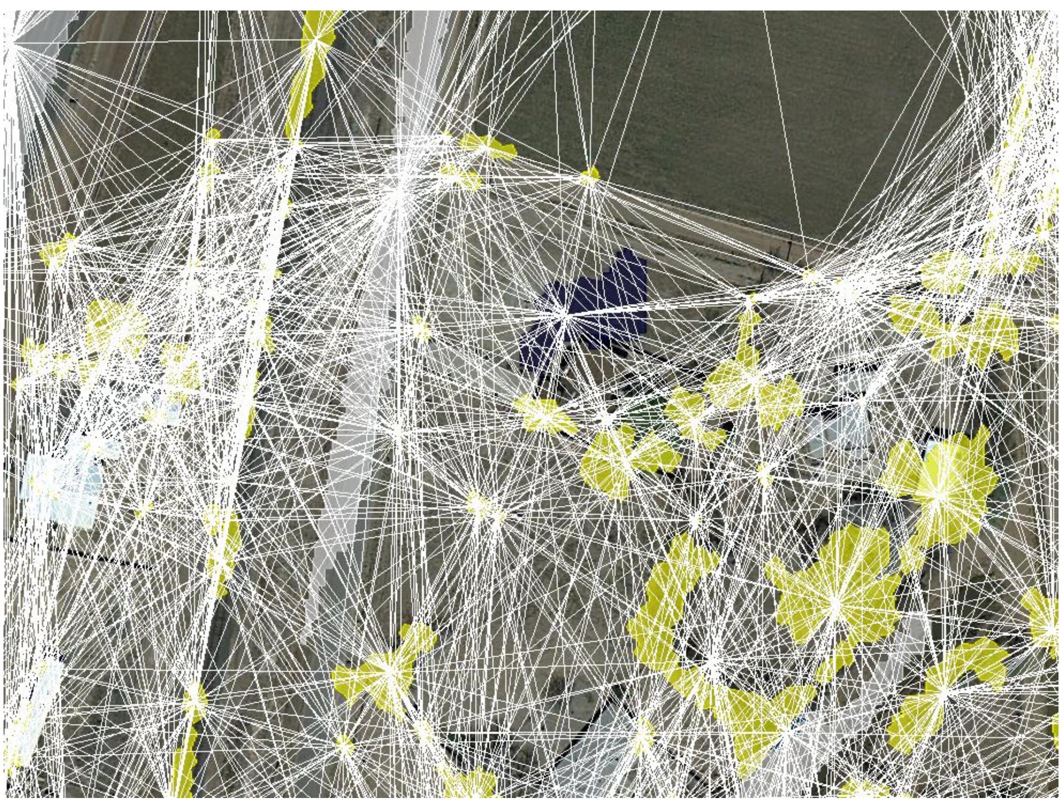


Figure 28. Close-up view of graph, with edge lengths up to 80 meters.

4.2. Graph Search Results

After constructing the graph from the recognized SAR features, we specified a search template and performed a search. Given the general uniformity of the rural area covered in the test data, and the broad nature of the terrain categories returned by the recognition algorithm, we chose a simple template for initial study.

The conceptual goal of the template was to identify small buildings near a road. The presence of a significant SAR shadow might aid discrimination of buildings compared to short features such as rail fences, so we included a shadow feature in the template. The template thus included a Tall node, Shadow node, and Road node. Here is the full definition:

Tall node:	$20\ \text{m}^2 \leq \text{area} \leq 500\ \text{m}^2$
Shadow node:	$1\ \text{m}^2 \leq \text{area} \leq 500\ \text{m}^2$
Road node:	$1\ \text{m}^2 \leq \text{area}$
Tall-Shadow edge:	$\text{distance} \leq 10\ \text{m}$
Tall-Road edge:	$\text{distance} \leq 50\ \text{m}$

Given this input, the search procedure applied to the full monolithic graph returned 115 matches in under 20 minutes on the same Mac workstation. These results are shown in Figure 29.

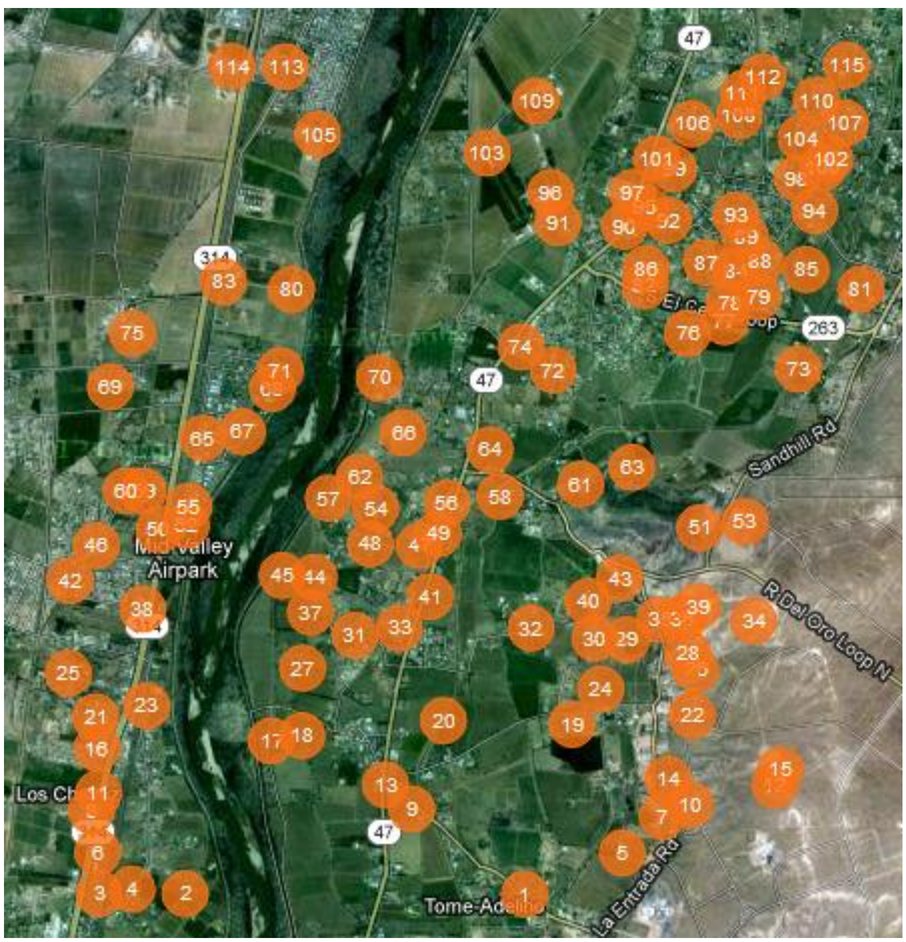


Figure 29. Matches found by the graph search.

How accurate are these results? To gain insight into this question, we made a quick, rough estimate. We visually surveyed a region starting in the lower-left corner of the image, proceeding northeast until reaching a position near match #23 in Figure 29. This study region covered perhaps 5% of the SAR image area. While studying this region, we made note of correct matches, and also counted small buildings that were not found. Items near the edge of the image were discarded to eliminate the special influence of boundary clipping.

Within this informal survey region, we found five matches that were not discarded. Of these, four were correct in the sense that the shadow matched the tall object that was a building, which was indeed close to a correctly identified road, paved or unpaved. We counted these as correct matches. One match was incorrect, because the road feature was not an actual road but instead a dirt lot, and the SAR shadow was not a real SAR shadow but rather a concrete parking lot that gave a low reflection back to the sensor. Meanwhile, we counted 40 occurrences of mostly buildings and a few large, well-defined trees as tall things that were not identified because they did not have the shadow behind them identified as a shadow.

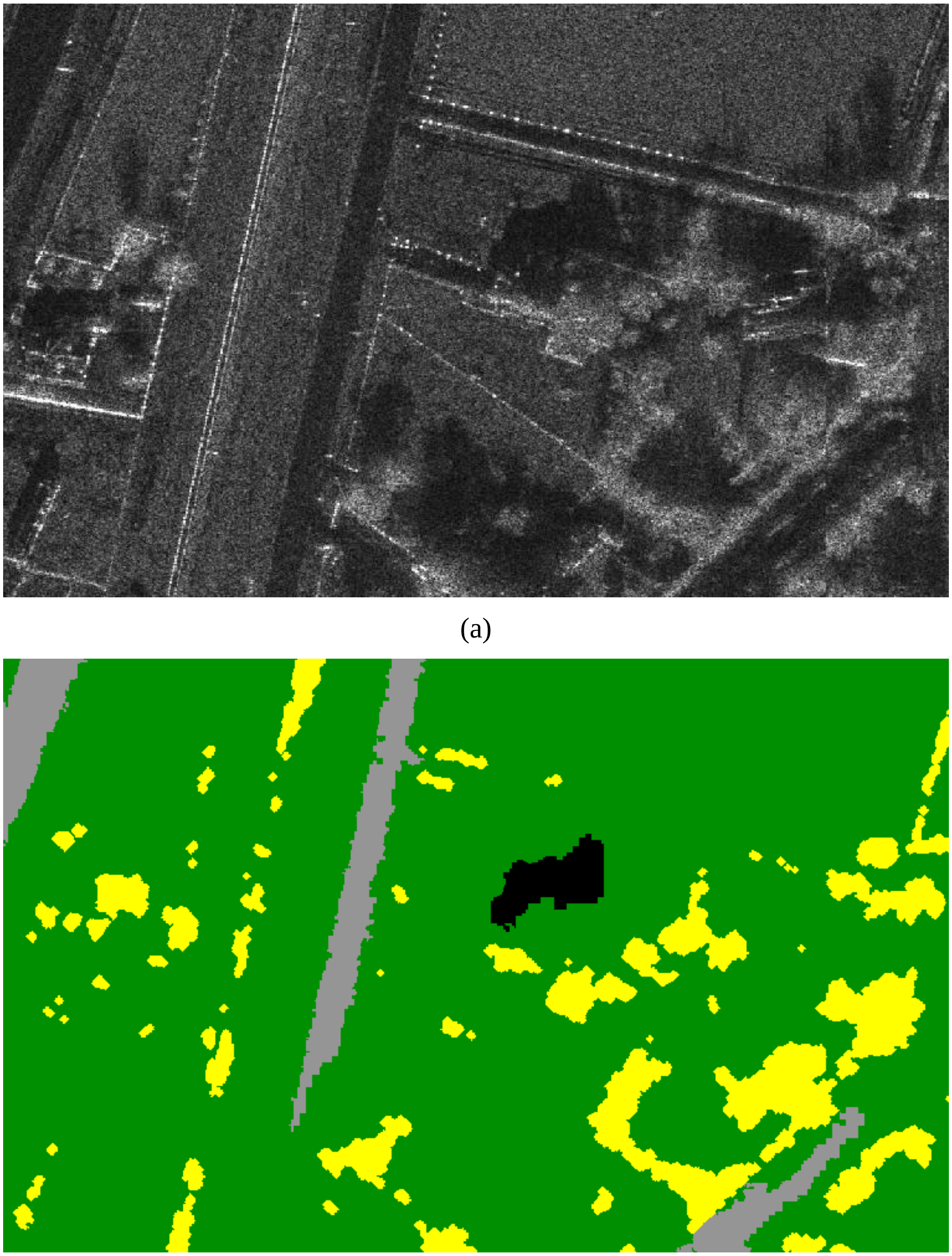
These initial measures allow us to compute a rough estimate of overall recognition accuracy. Of the (40+4) = 44 small buildings in the sample area, our graph search found four of them, yielding a producer's accuracy of 4/44 = 9%. Of the five buildings found by the algorithm in the sample area, four were correct, yielding a user's accuracy of 4/5 = 80%. These are obviously rough estimates, but given the very low producer's accuracy, investing additional effort to obtain a more precise accuracy estimate seems of dubious value.

To help better understand the results, we will review an example good match, and a few match failures.

4.2.1. Example Good Match

Figure 30 shows an example good match. The recognized tall feature, shown in yellow, corresponds to the portion of the house that is closest to the sensor. The matched shadow is indeed the shadow associated with this house. The adjacent road is a properly identified road feature. Figure 31 shows the SAR image and terrain cover for this example good match. For context, note that the close-up views shown parts (a) and (b) of Figure 31 correspond to the red outlines shown in Figure 24 and Figure 25, respectively. This same view is shown in Figure 27 and Figure 28; if you look closely you can see the match edges of Figure 30 as they appear in the context of the graph.

Figure 30. An example good match.



(a)

(b)

Figure 31. SAR image and terrain cover for the example good match.

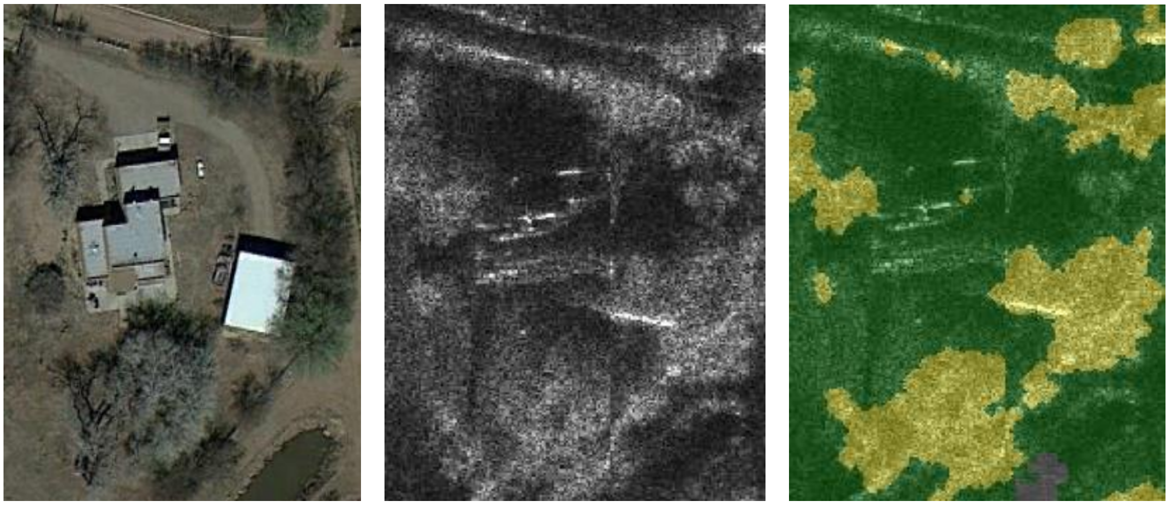
4.2.2. Example Match Failures

Figure 32 shows an example incorrect match, with the corresponding SAR image. The match includes Tall, Shadow, and Road features, but the shadow does not correspond to the tall feature. Instead, the shadow appears south of the tall feature, which would never be possible in SAR imagery looking north. This is a failure in the template design, which should be extended to include the geometric constraint that the shadow must appear on the north side of the tall feature.

But this is also a symptom of feature recognition failure, because the metal shed adjacent to the shadow should have resulted in a correct template match, and this was absent. The reason is that the shed did not produce a tall feature of adequate size and proximity to the shadow to generate a template match.

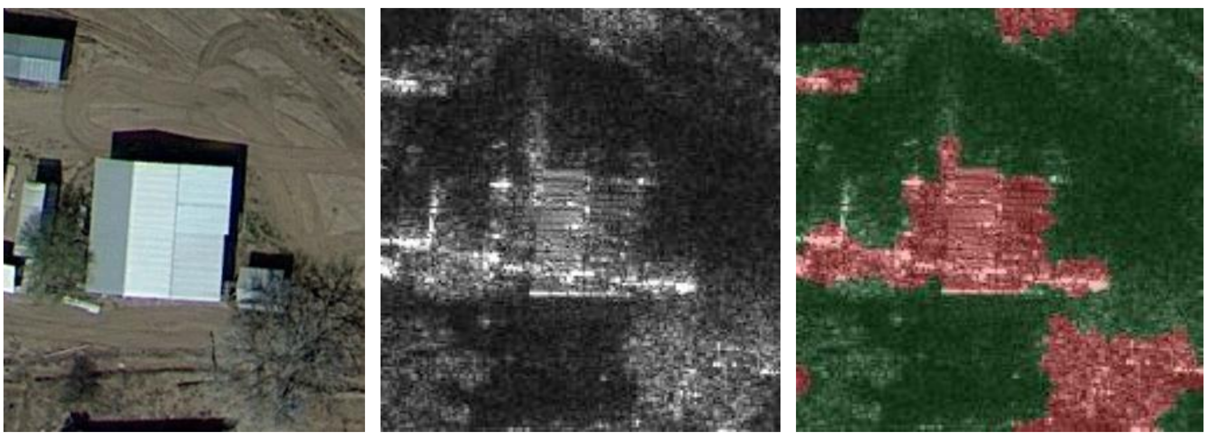
Figure 32. An example incorrect match.

Feature recognition failures were the primary cause of overall search failure. For our example goal of identifying small buildings, the principal recognition failures were failures to find the building and classify it as a Tall feature, or failure to find the Shadow adjacent to a Tall feature. Figure 33 and Figure 34 show example cases.



(a) (b) (c)

Figure 33. Example house that was missed by feature recognition.



(a) (b) (c)

Figure 34. Example house where shadow was missed.

It is worth noting that these failures are at least partly attributable to the feature recognition and graph search working at cross-purposes. Mr. O'Neil-Dunne views shadows as information loss, and thus designed his feature recognition code to return meaningful feature information in lieu of shadows whenever possible. Meanwhile, our search template requires a shadow to complete a match. These two perspectives are obviously in conflict.

One immediate response might be to adjust the feature recognition code to return shadows more often. However, this would discard useful information in some cases, a disadvantage. We will return to this topic in Section 5.3.

5. DISCUSSION

The search results obtained in this study are both poor and unexciting. They are poor because for the conceptual target we had in mind (small buildings), the producer's accuracy was only 9%. They are unexciting because even if we had achieved 100% accuracy, finding small buildings is not a result of great strategic importance. Nonetheless, we learned important information through this effort, which we will recount here.

First, let's review the successes of this work. Initial static feature recognition code was developed, which produced good performance results for water features. The resulting features were used in a semantic graph construction and search, providing a first end-to-end semantic graph analysis of SAR data.

Further, we gained some insight into the resources and time required to complete such an analysis. To sum up the time information reported above, SAR data collection took roughly two hours to cover a 42 km^2 area. Feature recognition for this data set required 2.0 hours, graph construction required 2.3 hours (exploiting parallelism), and graph search required 0.3 hours. Thus if we imagine a pipeline running this computation serially using the software and hardware described in this report, we would expect that search results for a two-hour data collection could be available within 4.6 hours lag time. Note that with required pre-processing complete, additional subsequent graph searches would run much faster — roughly two minutes for a similar template.

This process has not been optimized for speed, and there are multiple opportunities for run time improvement. Thus we can view this time measurement as an upper bound on what might be achieved for a computation with equivalent semantic performance. That said, we know that semantic performance must be improved, and this will likely add computation. So the results reported here give an initial reference regarding run time, without a clear indication of future performance.

We can make a similar summary of data size. The ensemble of input image patches consumed 1,050 MB of disk; the intermediate terrain cover map was 976 MB, and the files required to store the semantic graph were 541 MB. Thus the files required to support automated search required an additional 150% of the disk space required to store the SAR image data. This additional storage can be thought of as analogous to the disk space required to store fast lookup indexes in SQL databases. As with run time, we would expect that additional work would reduce required storage for equivalent semantics.

5.1. Why is SAR Feature Recognition So Difficult?

The performance of the feature recognition algorithm was a key factor limiting the overall recognition capability of this system. The difficulty in feature recognition resulted from intrinsic aspects of SAR image data that make automated image analysis challenging. In this section we will explore these aspects, to help better understand why these recognition problems are hard, and to inform possible future work to do better. We will review aspects of SAR phenomenology, the lack of a consistent representations, and orthorectification issues.

5.1.1. Phenomenology

The phenomenology of SAR brings many well-known difficulties to image interpretation. One example is range layover, where reflective features on tall objects can appear displaced relative to their ground locations, due to the grazing angle used in image capture. Another example is multi-path reflection, where the radar beam bounces from one object to another before returning to the antenna; this results in a range error that causes displaced signal information in the image.

These effects are well-known, and are the subject of on-going research in SAR image analysis (Short 2005; Wolff 2012; Lamont-Smith2011; Andre, Hill, and Moate 2008; Krishnan, Yarman, and Yazici 2010). In the following sections we will discuss other perspectives on this issue.

5.1.2. No Consistent Representation of Objects

A more fundamental problem is that SAR images produce widely different renderings of seemingly similar items. In discussions regarding this issue, Mr. O'Neil-Dunne put it well: "Features don't have a consistent representation in SAR data. This makes automatic recognition difficult."

When viewed from this perspective, it is easy to find supporting examples when closely studying SAR images. Here we will consider houses as an exemplary case. Figure 35 shows a house and its corresponding SAR image. This example shows a case where the house roof produces a clear signature that is both bright and delineates the full feature outline. The corrugated metal roof on this house produced an easily recognizable, house-like signature.

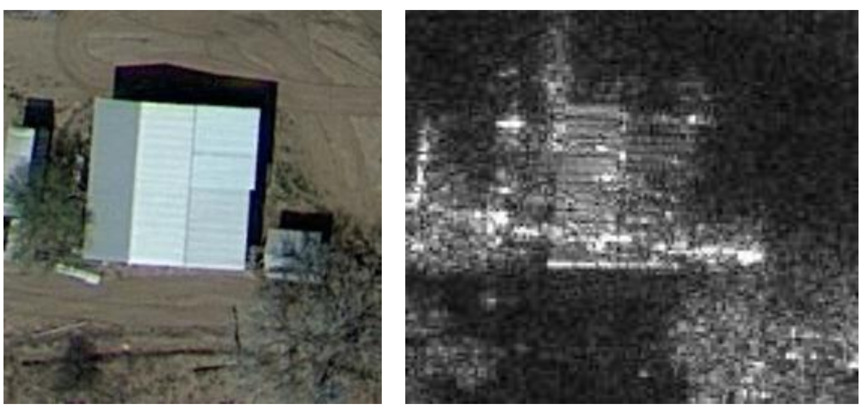


Figure 35. Example house with a clear SAR signature.

But this easily-recognized signature is not seen in many cases. Figure 36 shows several examples. House D on the right has a parapet roof, which produces a faint outline in the SAR image corresponding to the house shape. House C has a corrugated roof similar to the example in Figure 35, but in this case the roof area is not seen, but instead a bright highlight is observed along the roof's southernmost edge. House E exhibits a faint, partial outline. House A on the left does not seem to appear at all; the authors do not see how this house might be recognized even by a human. Meanwhile, label B indicates a bright outline in the SAR image that appears very similar to the outlines of houses D and E; however, this is not a house at all, but rather a closed fence for containing livestock.

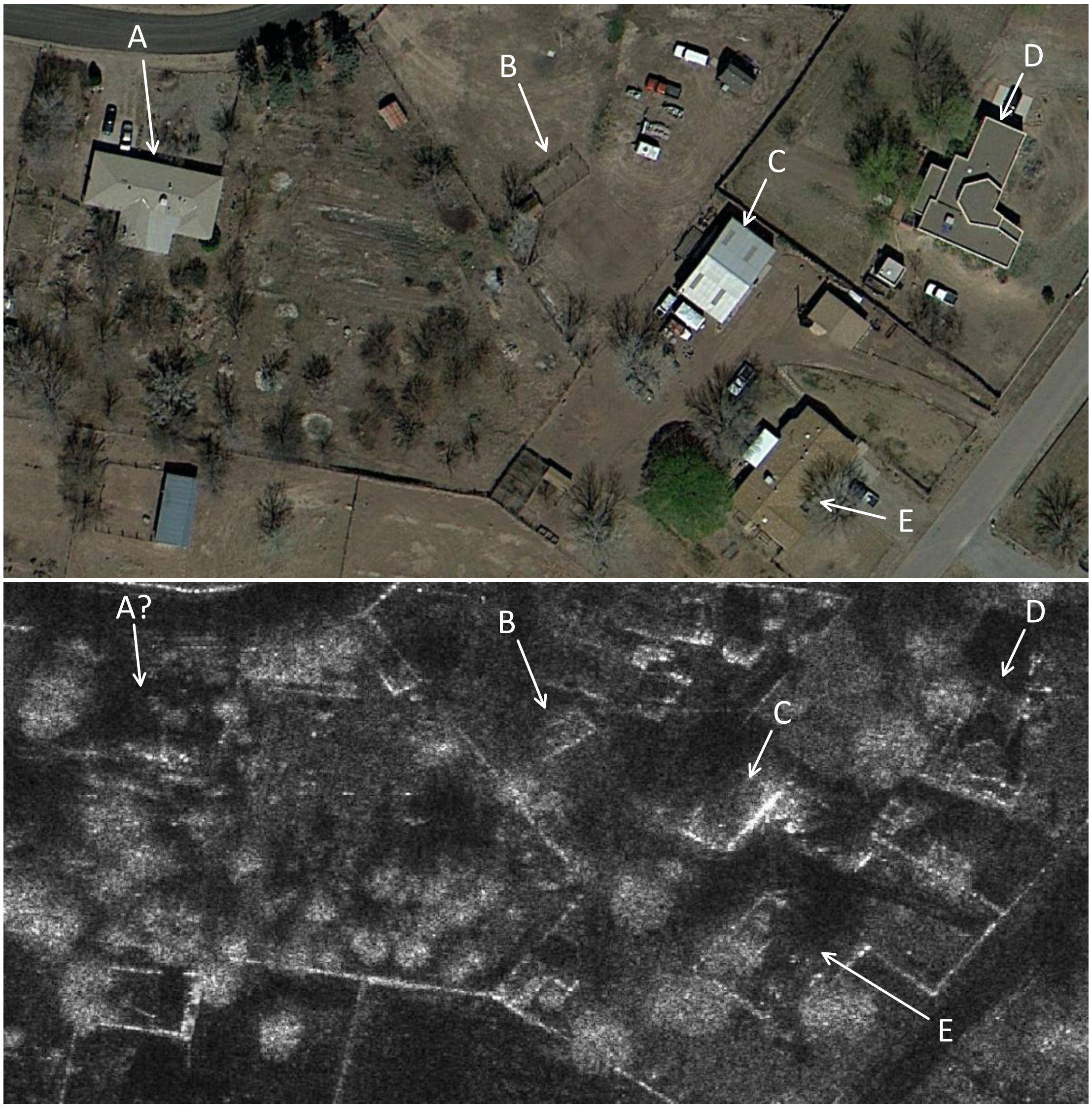


Figure 36. Example houses of varying recognition difficulty.

The examples of Figure 35 and Figure 36 support Mr. O'Neil-Dunne's observation that features do not exhibit a consistent representation in SAR image data. Examples selected from a basic category of "small building" appear as bright rectangles corresponding to their area (Figure 35), faint outlines (Figure 36, D), partial faint outlines (E), bright linear highlights (C), or seemingly nothing at all (A). Meanwhile, non-building features can produce SAR signatures remarkably like some buildings (Figure 36, B). It would be difficult, if not impossible, to correctly recognize some of the features in Figure 36 without the accompanying optical imagery. Confronting this variability clearly creates a challenge for automated SAR static feature recognition.

5.1.3. Georectification Issues

Another problem that caused difficulty for UVM when developing the feature recognition algorithm was the presence of feature dislocations resulting from orthorectification issues. Figure 37 shows a lateral dislocation occurring at the stitch boundary between two SAR image patches. This caused misalignment of several road features, and complete disconnection of the highway median and the bright linear feature alongside the highway which is a railroad. These errors confound algorithms that exploit large-scale feature properties, such as tests designed to identify long contiguous stretches of roadway.

Figure 37. Image patch registration errors.

This sort of problem is well-known in the field of aerial image processing, with several papers addressing the problem of stitching together images taken from different viewpoints. Examples include (Afek and Brand 1998; Leprince, et al 2007; Thomas, et al 2008; Westerteiger, et al 2012).

Proper georectification of SAR imagery is challenging, but is important for supporting downstream analysis processes that consider more than just a single image patch. These include feature recognition algorithms as described above, and also semantic graph analysis. Orthorectification is needed to support registration for two primary reasons:

- Reason #1: Within the SAR data modality, registration across patches is necessary for proper modeling and identification of large-scale and linked features (roads are but one of many examples).
- Reason #2: To allow reasoning across multiple data modalities, SAR data must be registered with other data types: GIS road network data, phone book information, RGB-IR image data, LIDAR data, etc.

Note that his second reason implies a stronger requirement — the need for proper geospatial location of image features. Expressing SAR feature data in terms of geospatial coordinates would enable (a) registering adjacent SAR image patches (as seen above), (b) registering SAR data collected from different view angles to a common coordinate system, and (c) registering SAR image data with other data and information sources.

5.2. Lessons Learned

A key outcome of this work is confirmation of an obvious constraint regarding geospatial semantic graphs: search capability is limited by both the quality and semantic depth of feature recognition. Related work studying geospatial semantic graphs has focused on this issue, aiming to produce representations that have increased robustness to data errors, and allowing an increase in semantic depth by exploiting provenance back-pointers to obtain focused probes into raw data to improve discrimination capability. This work is in progress.

Since recognition performance limits semantic graph search performance, an important corollary is that for SAR data, strong search performance will only be enabled by substantial improvements in feature recognition capability. Given the difficulties inherent in SAR data (see above), what strategies might we pursue? This is the topic of the next section.

5.3. Possible Next Steps

Reviewing this work suggests several ideas for future improvement. Here we will discuss five of these ideas: utilizing other data sources, georectification, developing a hybrid shadow representation, applying SAR data recognition to find perimeter fences, and addressing effects due to image tiling.

5.3.1. Utilizing Other Data Sources

As we have seen, SAR data includes several artifacts that make recognition difficult. Further, Mr. O'Neil-Dunne's direct field experience suggests that SAR has not historically been used as a sole source of static feature information. Instead, it has been used successfully in conjunction with other data sources which provide useful context.

This seems an idea worth exploring. Static feature recognition is a particular task that might benefit greatly from ancillary data sources. For almost all scenes in the world, it seems likely that prior data exists and is available. These data might take the form of optical imagery, topographic height information, or road network information. All of these data sources might be used to provide context to enhance the information gained from SAR image data.

As one simple example, consider the recognition error seen in Figure 23, where the large shadow on the north side of the hill is misclassified as water. Based on the SAR pixels alone, this shadow looked very similar to a lake. However, we know based on other information that this is in fact a shadow on the back side of a large hill. In reality this cannot be a lake, because the underlying surface is sloped. Ordinary topographic information could easily reveal this to a computer analysis, because the hill would be clearly represented. Recognition algorithms that consider this information would be able to readily decide that the region must be a shadow, because the alternative classification of standing water would be physically impossible.

Similar progress might be made using road network information. For the vast majority of anticipated SAR images, roads in the scene are likely already mapped in digital form. Why not make use of this information, instead of struggling to find all roads from scratch? Given recognition results informed by GIS road models, regions of change to the road network seen through SAR might be readily recognizable, and even more significant when detected as a change from prior data models.

Further, other sensors might provide information that complement SAR's special capabilities. For example, LiDAR data provides excellent (x, y, z) surface models, which could obviate the need for difficult recognition of buildings and other durable structures using SAR. At the same time, SAR might be very effective at detecting changes to these structures. In addition, LiDAR could provide both the terrain elevation and key feature position measurements necessary to enable registration and georectification of SAR image data, needed for several reasons described above. The necessary LiDAR data could be captured infrequently under preferred conditions, followed by regular re-visits by a SAR sensor.

All of these suggestions fall under the approach of exploiting each information source to gain the best information it provides.

If we choose to explore this approach, note that optical imagery, LiDAR, topographic, and GIS road network data are already available for the region covered by our test mosaic SAR image.

5.3.2. Georectification

Section 5.1.3 above explains why georectification is an important step allowing analysis of data across multiple SAR images, or across multiple data modalities. Section 5.1.3 makes the case for this result, but we mention it again here as an important possible next step.

5.3.3. Hybrid Shadow Representation

As noted above, the feature recognition and graph search procedures employed in this report worked at cross-purposes. The feature recognition code aimed to minimize shadows, since labeling a ground feature as Shadow amounted to admitting complete information loss. Meanwhile, the graph search desired shadows as confirmation of the substance of a tall object.

This case brings to light a fundamental issue surrounding shadows, both for SAR and optical image data. We'll explore this first discussing visible optical sensing, and then return to SAR.

In our personal experience, we encounter shadows all the time. The shadows provide us with important information; when a shadow is present, it implies that something is nearby that blocks the primary source of incident light. This might be a tall building, a tree, a passing cloud, or a pterodactyl about to attack us. In all of these cases, the shadow provides important information about our surroundings.

Yet at the same time, we can still generally see within the shadow, and make out important features. In our outdoor experience this is enabled by scattered light from the sky, and the shadow penumbra that results from the finite diameter of the sun. This ability to see into shadows and glean important information is essential to our survival.

Inspection of the sample SAR data reveals that something similar is going on with SAR shadows. First, shadows provide a very handy cue for image interpretation. Tall objects with distinct shapes, such as trees in an open field, are more easily recognized and understood by considering the shape of their shadows. At the same time, shadows can include important image information. Figure 38 shows an example. Notice the fence line, delineated by the series of bright dots. There are SAR shadows from nearby trees that are cast over the fence, but the fence line is still visible within the shadows! This is important if we want to know whether the fence is continuous or has a gap. We are not certain why this occurs, but will conjecture that this image witness to the fence continuation through the shadow is the result of penumbra effects due to the finite length of the SAR synthetic aperture, much like the penumbra due to the finite diameter of the sun.

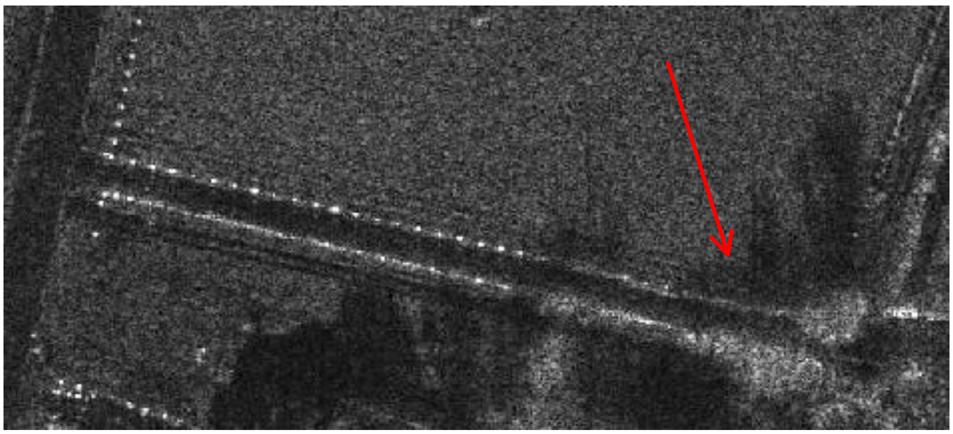


Figure 38. Example shadow which still contains useful information.

These observations raise a dilemma for those designing feature recognition and semantic graph algorithms: Should we design our codes to find and exploit shadows, or minimize categorization as shadows, so that the information within the shadows can be utilized?

We propose to have the best of both choices, by introducing a representation of hybrid shadows. These are land cover classes that retain the usual category choices, but with the modifier "in shadow."

From the feature recognition perspective, this might make things easier, because (1) it eliminates the need to make an arbitrary decision in resolving the dilemma described above, and (2) it might reduce brittleness resulting from the selection of a shadow threshold for image analysis. To facilitate robust recognition, it may help to loosen the expectation of precision of the shadow boundary, since these may be poorly defined.

From the semantic graph perspective, this will certainly improve capability. Given returned feature classes with "in shadow" modifiers, separate graph nodes would be constructed for regions in and out of shadow. However, at search time these nodes would be merged appropriately. Representations that desire shadows for adjacency reasons (as in the template explored in this work) would have shadow information available. Representations seeking information ignoring shadow information (such as asking whether the Figure 38 fence is continuous) can simply merge adjacent nodes whose attributes agree in all semantic characteristics except the shadow modifier.

5.3.4. Using SAR to Find Perimeter Fences

We have seen how some features, such as small buildings, are easy to see in optical images but can be difficult to see in SAR. Some features are the opposite, proving difficult to see in optical images but easily recognized in SAR image data.

The rural area in the sample SAR data set is full of livestock fences, and these are clearly seen throughout the SAR imagery, including several figures above. These fences are much less apparent in the optical images.

In some cases perimeter fences can be quite difficult to see, yet are very important. Consider the example shown in Figure 39. This nice-looking facility has a beautiful skylight and surrounding recreational areas that suggest it might be a school. The availability of parking for teaching and administration staff supports this hypothesis.

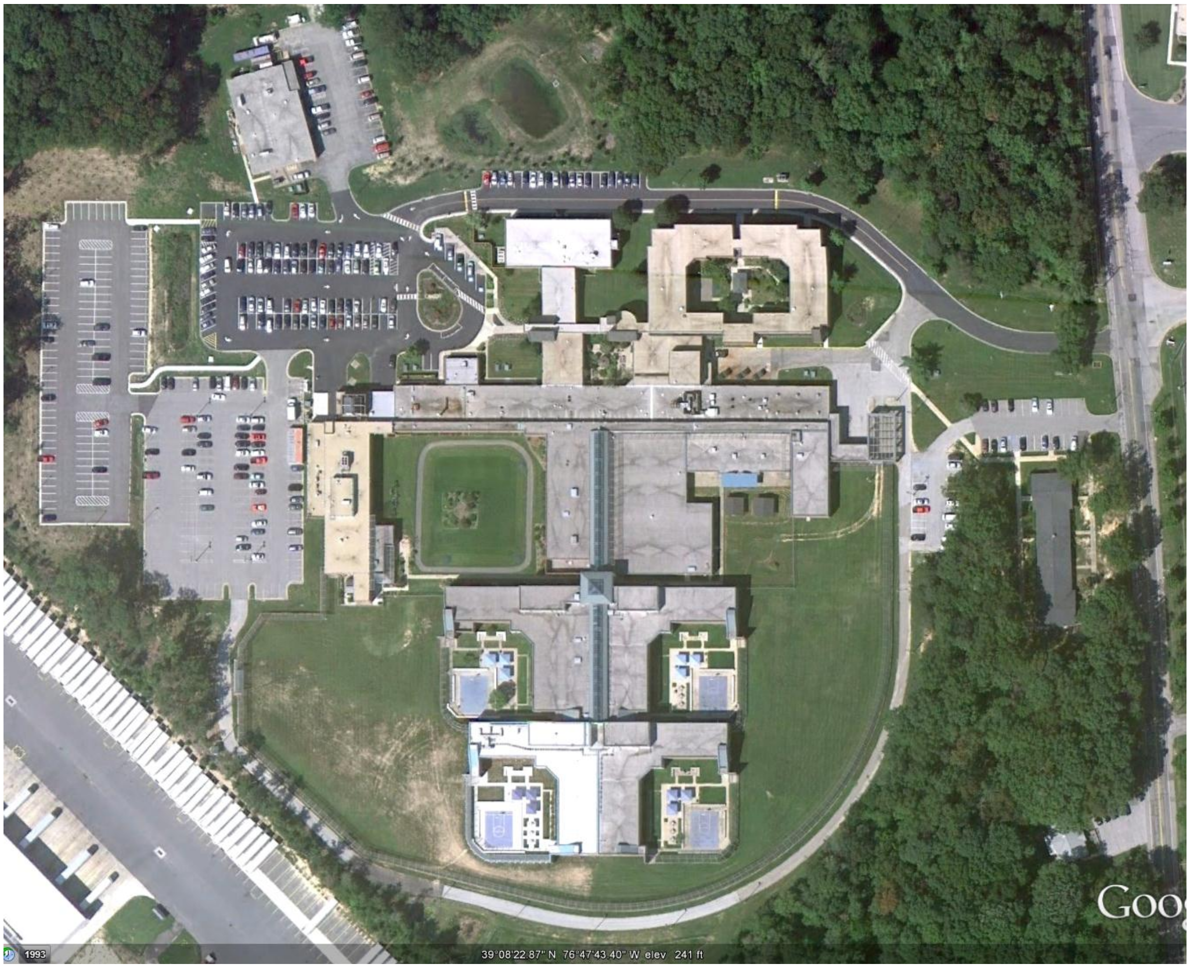


Figure 39. What is this place?

And yet there's something strange about this place. Very close inspection of the optical image reveals that it is surrounded by two layers of fence — one surrounding the individual recreation areas, and a second surrounding the entire facility. (These are so faint they might be invisible in your rendering of this image.) There is a large gap between the inner and outer fences, and further inspection reveals that the outer fence is quite large, and bends over at the top toward the interior of the facility. Odd. What is this place?

It turns out that this facility is a hospital. The Clifton Perkins Hospital, which is a maximum-security forensic hospital for dangerous mental health patients.

Notice that in studying this facility, it was the hard-to-see fence that raised our interest. This observation suggests that using SAR in conjunction with other sensing modalities might be quite useful at detecting secured facilities, which might merit further scrutiny. Geospatial semantic graphs might provide one effective method for representing the data from multiple modalities in a uniform representation facilitating search — imagine a template seeking a building complex found in optical data, in proximity to or surrounded by a perimeter fence found by SAR data.

5.3.5. Eliminating Effects Due to Image Tiling

We have seen how image tiling can be very useful in improving computation speed. Breaking the large data mosaic into tiles was very effective at speeding up both feature recognition and graph construction. If we desire fast computation and low latency, it seems that breaking data into smaller chunks and dividing the computation among different cores will play a key role.

But breaking the image into tiles creates artificial boundaries, and these have been seen to produce adverse effects. There are many examples noticeable at a zoomed-in level, but it's most convenient to point to two large scale effects, readily visible in Figure 23. Tile boundary effects played a role in the terrain categorization errors highlighted by both arrows. The river section misclassified as shadow would likely have been correctly merged into adjacent water regions, but tile boundaries prevented this (compare with Figure 14). Similarly, tile boundaries prevented the large shadow north of the hill from being classified as one big contiguous lake. This would have been an error, but it shows how tile boundaries can interfere with image processing, yielding unexpected results. Similar problems affect semantic graph analysis; we'll skip the details here.

These observations underscore the importance of understanding the effect of image tiling strategies, and preferably designing algorithms that are either immune to tile effects, or which correct problems due to tile boundaries during some sort of healing post-processing.

5.4. Conclusion

This report has described work toward developing geospatial semantic graphs capable of temporal reasoning, and applying semantic graph analysis techniques to SAR data.

Effort in the first area has yielded a design for an efficient graph representation encoding both geospatial and temporal concepts, and which appears promising for solving a wide variety of spatial, temporal, and spatiotemporal search queries.

Effort in the second area has demonstrated feature recognition, geospatial semantic graph construction, and search based on input SAR data. While recognition performance needs substantial improvement, this work represents a successful initial step demonstrating feasibility and illuminating important focus areas for future work.

REFERENCES

(Afek and Brand 1998) Y. Afek and A. Brand, Mosaicking of Orthorectified Aerial Images, *Photogrammetric Engineering and Remote Sensing* **64**(2), pp. 115-125, February 1998.

(Andre, Hill, and Moate 2008) D. B. Andre, R. D. Hill, and C. P. Moate, Multipath Simulation and Removal from SAR Imagery, *Proceedings of SPIE 6970, Algorithms for Synthetic Aperture Radar Imagery XV*, April 2008.

(Congalton and Green 2009) R. Congalton and K. Green, *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices, 2nd Edition*, CRC Press/Taylor & Francis, 2009.

(eCognition 2012) eCognition, *eCognition Developer 8.7.2 Reference Book*, Trimble, 2012.

(Evans and Hudak 2007) J.S. Evans and A.T. Hudak, A Multiscale Curvature Algorithm for Classifying Discrete Return LiDAR in Forested Environments, *IEEE Transactions on Geoscience and Remote Sensing* **45**(4), pp. 1029-1038, April 2007.

(Kluckner and Bischof 2010) S. Kluckner and H. Bischof, Large-Scale Aerial Image Interpretation Using A Redundant Semantic Classification, in N. Paparoditis, M. Pierrot-Deseilligny, C. Mallet, and O. Tournaire, Eds., *IAPRS, Vol. XXXVIII, Part 3B*, pp. 66-71, September 2010.

(Krishnan, Yarman, and Yazici 2010) V. Krishnan, C. E. Yarman, and B. Yazici, SAR Imaging Exploiting Multi-Path, *Proceedings of the IEEE National Radar Conference*, pp. 1423-1427, June 2010.

(Lamont-Smith 2011) T. Lamont-Smith, Range-Layover to Height Mapping in High Grazing Angle Synthetic Aperture Radar for Non-Interferometric 3-D Image Formation, *IET Radar, Sonar, and Navigation* **5**(8), pp. 871-876, October 2011.

(Leprince, et al 2007) S. Leprince, S. Barbot, F. Ayoub, and J.-P. Avouac, Automatic and Precise Orthorectification, Coregistration, and Subpixel Correlation of Satellite Images, Application to Ground Deformation Measurements, *IEEE Transactions on Geoscience and Remote Sensing* **45**(6), pp. 1529-1558, June 2007.

(O’Neil-Dunne, et al 2012) J. P.M. O'Neil-Dunne, S. W. MacFaden, A. R. Royar, and K. C. Pelletier, An Object-Based System for LiDAR Data Fusion and Feature Extraction, *Geocarto International*, pp. 1-16, Taylor and Francis Group, 2012.

(Short 2005) N. M. Short, Remote Sensing Tutorial,
http://www.fas.org/irp/imint/docs/rst/Front/overview.html.
Section on range layover: http://www.fas.org/irp/imint/docs/rst/Sect8/Sect8_4.html.

(Strip and Watson 2011) D. Strip and J.-P. Watson, Building an Enterprise Solution to Complex Target Recognition Through Semantic Graphs, Sandia external presentation, 2011.

(Thomas, et al 2008) U. Thomas, F. Kurz, D. Rosenbaum, R. Mueller, and P. Reinartz, GPU-Based Orthorectifcation of Digital Airborne Camera Images in Real Time, *Proceedings of the XXI ISPRS Congress*, 2008.

(Westerteiger, et al 2012) R. Westerteiger, F. Chen, A. Gerndt, B. Hamanny, and H. Hagen, Remote GPU-Accelerated Online Pre-Processing of Raster Maps for Terrain Rendering, *Proceedings of Ninth Workshop on Virtual Reality and Augmented Reality of the GI Expert's Group (VRAR 2012)*, Bonn, Germany, 2012.

(Wolff 2012) C. Wolff, Radar Tutorial: Synthetic Aperture Radar, http://www.radartutorial.eu/20.airborne/ab07.en.html.

DISTRIBUTION

1	MS0519	Steve Castillo	5346
1	MS0519	Kristina Czuchlewski	5346
1	MS0519	Ana Martinez	5344
1	MS0519	Laura McNamara	5346
1	MS0532	Bryan Burns	5340
1	MS0532	William Hensley	5344
1	MS0532	James Hudgens	5340
1	MS0532	Cameron Musgrove	5344
1	MS0576	John Feddema	5511
1	MS0576	Kathy Simonson	5511
1	MS1326	Ojas Parekh	1465
1	MS1326	Mark D. Rintoul	1465
1	MS1326	Jean-Paul Watson	1465
1	MS0359	D. Chavez, LDRD Office	1911
1	MS0899	Technical Library	9536 (electronic copy)