A Computational Framework for Ontologically Storing and Analyzing Very Large Overhead Image Sets

Randy C. Brost
Sandia National Laboratories
Albuquerque, New Mexico
rcbrost@sandia.gov

William C. McLendon III
Sandia National Laboratories
Albuquerque, New Mexico
wcmclen@sandia.gov

Ojas Parekh
Sandia National Laboratories
Albuquerque, New Mexico
odparek@sandia.gov

Mark D. Rintoul
Sandia National Laboratories
Albuquerque, New Mexico
mdrinto@sandia.gov

David R. Strip
Sandia National Laboratories
Albuquerque, New Mexico
drstrip@sandia.gov

Diane Myung-kyung Woodbridge
Sandia National Laboratories
Albuquerque, New Mexico
dwoodbr@sandia.gov

ABSTRACT

We describe a computational approach to remote sensing image analysis that addresses many of the classic problems associated with storage, search, and query. This process starts by automatically annotating the fundamental objects in the image data set that will be used as a basis for an ontology, including both the objects (such as building, road, water, etc.) and their spatial and temporal relationships (is within 100 m of, is surrounded by, has changed in the past year, etc.). Data sets that can include multiple time slices of the same area are then processed using automated tools that reduce the images to the objects and relationships defined in an ontology based on the primitive objects, and this representation is stored in a geospatial-temporal semantic graph. Image searches are then defined in terms of the ontology (e.g. find a building greater than 10^3 m^2 that borders a body of water), and the graph is searched for such relationships. This approach also enables the incorporation of non-image data that is related to the ontology. We demonstrate through an initial implementation of the entire system on large data sets ($10^9 - 10^{11}$ pixels) that this system is robust against variations in different image collection parameters, provides a way for analysts to query data sets in a more natural way, and can greatly reduce the memory footprint of the search.

Categories and Subject Descriptors

H.4.2 [**Information systems applications**]: Types of Systems - Decision support; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalism and Methods - Semantic networks; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - Information filtering, Search process

General Terms

Algorithms, Design, Experimentation

Keywords

Geospatial-Temporal Data Analysis, Semantic Graph

1. INTRODUCTION

Analysis of overhead imagery is a key technology in commercial and national security enterprises. Modern imaging technology has both decreased the cost and increased the availability of high-resolution images. We are in an era where so much image data is acquired that in both commercial and defense domains, we are limited by the time it takes for human analysts to use these images to solve the problems of interest. Because of this bottleneck, computers have been used at many steps in the image analysis pipeline to help alleviate many of the problems associated with large image data sets. While this has been successful for many parts of the analysis process, many of these solutions simply work at adding efficiency to the classic method of having analysts look at a series of images for a specific pattern of objects, or perhaps looking at a series of multiple time slices of a particular scene for changes. Despite the expertise of human analysts, this approach is quickly becoming overwhelmed in the face of ever increasing amounts of data.

We propose here a very different approach to image analysis that creates a pipeline that is focused on augmenting the analyst's capabilities at a higher level. In our approach, the images are first processed and stored in terms of primitive objects that are used as the building blocks of an *ontology* that is defined by the interests of the analyst. Then, the specific questions that interest the analyst can then be addressed directly by the processed data. The answers are then returned to the analyst in terms of cues on the original images that matched the queries, sorted by the quality of the match. This model supports both an interactive sort of approach that could be furthered refined by modifying the query, or a batch approach, where very large image databases could be searched off-line and then return a list of hits to be examined later.

The key construct that enables this approach is a graph. More specifically, it is a *geospatial-temporal semantic graph*

(GTSG). The GTSG stores the information about the relevant ontology via its nodes and edges. The nodes hold all of the objects in the ontology along with a rich set of details about the objects that can be used as part of an analyst query. The spatial and temporal relationships between the objects, along with the details of these relationships, are stored in the edges. Then, an image query is not translated to a pixel-based query in order to search the images directly, but instead is framed as a more natural ontological query (find a building less than 1 year old that is within 50 meters of a large body of water and also is next to a parking lot) that is searched for as a pattern within the graph.

The GTSG approach as described in this paper has a number of potential advantages over traditional search strategies. These include:

- Allowing complex image searches to be described and executed in a more natural manner.
- Enabling specific types of change to be specified in search over multiple time slices.
- Greatly reducing the size of the data that needs to be searched by quickly eliminating all nodes and edges not related to the search query.
- Allowing data fusion of different types of images (such as visible and lidar) along with non-image data, such as GIS data and LiDAR.
- Reducing sensitivity to irrelevant image-to-image variation by decomposing the images into large objects within the ontology.

In the rest of the paper, we'll set the context for the work we have done by giving a high-level overview of some previous work by others. Then, we'll go into some of the details of our process and demonstrate its strengths and weaknesses by showing some example searches.

2. BACKGROUND

The notion of context being important in image analysis has been around for a long time. Olson mentioned it as one of the "essential elements of information" as far back as 1960 [10]. Even before that, the notion of doing feature detection via contextual analysis of components of the whole feature was emphasized by Chisnell and Cole [3], although that work was focused around the manual process of image analysis. The importance of context was put in a more formal structure by Biederman et al. who proposed specific types of contextual relationships between an object and the objects around it [1]. Since that time, much work has been done to try to study the effectiveness of context in image recognition [6], although most was not done in terms of an engineering workflow on large-scale examples. An excellent recent example of incorporating context into a workflow was given by O'Neil-Dunne as part of a process for distinguishing different types of land cover [11]. We will demonstrate that the use of the underlying graph structure in our process allows for a much more sophisticated use of context.

Representing change between images in a non-trivial way is also an important problem in computer science. Radke et al. give an excellent survey of the work that has been done, and present a taxonomy that includes a majority of the approaches [15]. There are a few previous examples of attempting to use the notion of semantic context to represent change. Gressin et al. describe an approach where pixels are hierarchically classified based on their similarity to surrounding regions, and this information is used to make decisions about relevant change on the whole image [7]. This is related to our approach, but is ultimately focused on classifying the pixels, and not amenable to semantic search. Bruzzone and Bovolo also approach the problem of change detection in high resolution images through semantic means, but the encoding is ultimately in a series of descriptors [2].

Perhaps the most commonly used competing technology to this work is that of geospatial databases. Spatial databases archive and support spatial queries on analyzed geospatial image data. Spatial queries include geometric questions regarding location or shape, or overlap/distance relationships among geometric data. While standards for representation and storage in geospatial databases are improving, the querying operations offered by many commercial or open-source databases have a limited capability for expressing complex geospatial relationships and geodetic feature representation [16]. For standardizing geospatial data model representations with respect to geospatial semantic properties, the Open Geospatial Consortium (OGC) released CityGML, an international standard for three-dimensional urban models [13, 9]. In contrast to the availability of several tools for editing and visualizing this new geospatial model, software for optimizing storage, analyzing, and mining the new data model still needs improvement.

Semantic graphs are a powerful way to encode spatiotemporal data. A semantic graph is a representation comprised of nodes and connecting edges, where both nodes and edges have associated attributes. Edges between nodes can encode geospatial and temporal relationships. Several studies have investigated mapping semantic graph structures and analyzing semantic data [5]. Many existing studies focused on either temporal or spatial semantic graphs. For instance, Passino et al focused on spatial relation analysis [14]. The OGC has proposed GeoSPARQL as a standard representation of the geospatial semantic graph model using Resource Description Framework (RDF). We considered using GeoSPARQL as the as our data model, but for this initial implementation, we chose not to. We did not find GeoSPARQL suitable for automated change detection in remote sensing data [8], and some of the subtleties associated with doing graph searches would be more difficult with GeoSPARQL. However, we are considering using it for future generations of the code.

While this work describes a detailed implementation and thorough testing of the concepts that were described by a subset of the previous authors [19], there has been some recent similar work done by Yue et al [20]. That work also demonstrates the notion of using semantics to identify features in images through their individual components. However, there were significant assumptions made in what labeled data was made available and they did not focus on the more generic notion of a general graph to describe the relationships between the objects. There is also no notion of time or change that work. The process we demonstrate here is almost completely automated and therefore applicable to wide-area search, and is also applicable to the problem of change detection across multiple images from different times.

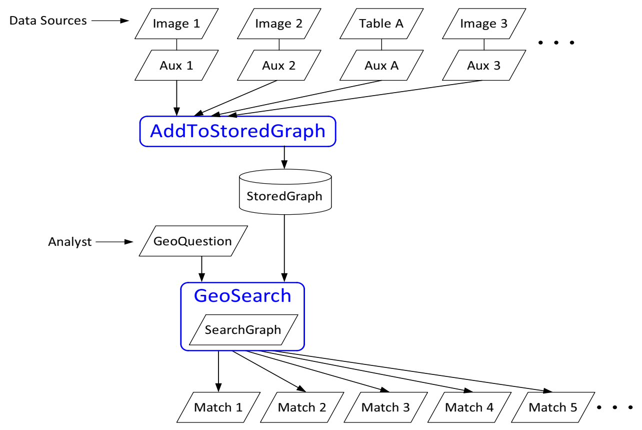


Figure 1: Basic computation flow.

3. REPRESENTATIONS AND ALGORITHMS

Our computation flow is based on constructing and searching a monolithic geospatio-temporal semantic graph, called a *StoredGraph*, and consists of two phases (see Figure 1): (i) a graph construction phase, embodied by the *AddToStoredGraph* program, and (ii) a graph search phase, embodied by the *GeoSearch* program. We implemented these programs in C++, employing an SQLite database to hold the StoredGraph via tables encoding the graph nodes and edges as well as associated geospatial and semantic attributes.

The AddToStoredGraph program accepts a land cover map and associated provenance image data; the land cover map assigns a land cover classification such as building, dirt, grass, road, water, etc. to each pixel of the image. The number and types of land cover classes are flexible. The land cover maps have associated pointers back to the raw image and height data used to construct them. In addition to land cover data, AddToStoredGraph is able to leverage geospatial point data, which is widely available through public databases, digital surface and elevation model data, and RGB spectral data. Each execution of AddToStoredGraph processes its input data and adds corresponding nodes, edges, and attributes to the StoredGraph, including pointers to the raw provenance files.

Objects that represent a contiguous land cover region from annotated image files are added to the StoredGraph through an optimized raster processing algorithm that keeps at least two lines of the image in memory at a time. This technique allows for a streaming algorithm to be used that uses much less memory while still allowing us to ascertain adjacency edges that correspond to contiguous regions within the image. Non-image point data is straightfoward to add to the graph. Distance edge calculations require a spatial indexing scheme to be done efficiently. We note that geospatial semantic graphs can allow different geospatial reference systems and encodings across different input data and manage geospatial encodings.

To complement the wide range of accepted data, a rich collection of basic node and edge types are supported; examples include nodes corresponding to geospatial points or regions and edges indicating region adjacency and distance between points or regions. In addition over fifty different attributes are currently used, reflecting geospatial, topological, semantic, and provence-related properties. Most attribute values are generated while the data is processed; however, more computationally demanding attributes such as internode distance for edges are computed on-demand at search time and chached for later re-use.

A natural approach to representing a geospatial-temporal graph is simply to construct a geospatial graph for each time slice and add temporal edges representing the correspondence between nodes across time slices. Although this is a complete representation, it is highly redundant; many nodes do not change from one time slice to the next and are unnecessarily replicated. We adopt a representation that avoids this redundancy. New nodes are only added to the graph if they have changed; the resulting temporal correspondence edges then imply a change condition. This representation is much more efficient and concentrates graph complexity where change occurs. In addition to the significant space savings, our representation allows us to perform change analysis by a simple topological analysis of the graph.

Since we do not store a complete geospatial semantic graph for each time slice, we endow each node with a compact temporal signature that encodes what is known about the existence of the node over time, given data observations seen so far. This chronology attribute differentiates our representation from a pure geospatial graph; the attribute consists of four fields: $(t_{\text{last.absent}}, t_{\text{first.seen}}][t_{\text{last.seen}}, t_{\text{first.absent}})$. Time $t_{\text{last.absent}}$ 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 $t_{\text{first.seen}}$ is the time of the earliest observation when the node was seen. Time $t_{\text{last.seen}}$ is the time of the latest observation when the node was seen. Finally, time $t_{\text{first.absent}}$ 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.

A crucial primitive operation in constructing our geospatial-temporal graphs is determining when a geospatial feature has truly changed. There are an abundance of environmental and sensor-related factors that could cause an unchanged feature to have small variations in shape across images; these include sensor differences, atmospheric conditions, small geolocation errors, and so on. For the results shown in Section 4, we determine whether a newly-observed land cover region has changed by computing the geometric difference between it and the corresponding region(s) from the prior time slice, and then performing a geometric shrink operation to eliminate spurious differences attributable to noise. Once we have determined whether change has occurred, we then modify the StoredGraph by either adding a new node and connecting change arc (edge) in the case that the node has changed, or simply updating the chronology fields if the node has not changed.

The GeoSearch program accepts a *GeoQuestion*, which defines a query. It is through the GeoQuestion that the analyst constructs the search query via a primitive ontology based on the regions that have been previously identified in the land cover map and by the relationships that have been assigned to specific pairs of nodes. While the ontology is necessarily constrained to the types of regions, their properties, and the relationships between them, a surprisingly rich description of search objects can be constructed in this way. These search objects are described as a graph that can then be searched for within the StoredGraph.

Using the GeoQuestion, the GeoSearch program constructs a *SearchGraph* in memory, and then applies a selected graph search algorithm to find matches which are rendered on output. When constructing the SearchGraph, the program identifies qualifying nodes by first applying SQL queries to fetch node candidates, then checking lazy-evaluated constraints, invoking detailed analysis of the raw provenance data when necessary. Edges are then constructed for SearchGraph node pairs satisfying GeoQuestion constraints.

A benefit of this approach is that the leaner SearchGraph is decoupled from the monolithic and persistent StoredGraph. While the StoredGraph has rich attributes and is expressed in the semantics of the data, the SearchGraph is compact and expressed in the semantics of the query. Thus SearchGraph construction accomplishes two purposes: It builds a lightweight in-memory graph containing only graph elements relevant to the current question, and it converts the semantics from the data semantics to the question semantics.

The query definition also specifies a search algorithm. A natural search approach is via the subgraph isomorphism problem; however, this problem is NP-complete [4, 17], although there is some hope that more tractable implementations are possible on structured graphs such as geospatial graphs. Our present work demonstrates the effectiveness of fast and simple search approaches that are alternatives to full-fledged subgraph isomorphism. There were two primary search options used in the results presented in this paper: star-graph search and connected-component search.

A *star-graph search* query is defined with respect to a hub node of a given type and potentially heterogeneous spoke nodes connected to the hub node. The star graph algorithm also includes the notion of a spoke-to-spoke constraint, which is a constraint which must be satisfied across certain spokes. This is required to solve some problems of interest, due to actual inter-spoke requirements. A *connected-component search* finds connected components of nodes in the SearchGraph, whose edges are controlled by the user-defined query specification. For example, a user may specify a connected-component search including edges only where two nodes are within a specified distance threshold.

4. RESULTS

4.1 Data

We applied the GTSG approach to solve problems for three geospatial regions including Anne Arundel County, MD, Philadelphia, PA, and Washington, DC. Using raw optical images (Figure 2), height maps measured by LiDAR (Figure 3) and GIS data, land cover maps were generated [12] (Figure 4). Anne Arundel County and Philladelpia data include one time slice from 2007 and 2008 respectively, while Washington data have two time slices from 2006 and 2011. For Washington data, land cover maps were derived from raw data optical images captured in 2006 and 2011, augmented by LiDAR from 2008 and additional GIS data.

Depending on the region, image pixel resolutions varied between 0.1 and 1.0 m, while the height map pixels were 0.3 - 1.0 m, and land cover pixels were 0.3 - 1.0 m. The size of each optical image, height map and land cover map is defined in Table 1. The land cover classification was primarily done using rule sets in eCognition, and the accuracy was verified to be approximately 95% in all of the data sets. This level of accuracy was good enough for effective testing of the methodology while still having enough error to test the robustness of the algorithms. A more complete discussion of the process and the accuracy analysis can be found in [12].

Our input data also included a text file containing a list of 100 hospitals in the Washington area, including their name and (latitude, longitude) coordinates, along with other attributes [18].

4.2 Geospatial Search

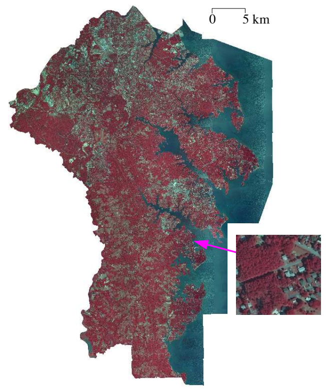
Anne Arundel County, Philadelphia, and Washington data in Table 1 were input to the graph construction program (AddToStoredGraph), which produced a StoredGraph for each region. The resulting graphs included a node for each land cover region, with associated attributes such as land cover type, bounding box, centroid, area, eccentricity, etc. The resulting StoredGraphs contained a total of over 3.6 million nodes and 8.1 million adjacency edges. GeoSearch added distance edges on demand.

We discuss the results below in detail in order for the strenghts and limitations of the approach to be understood more clearly through the examples. We also present our results in terms of traditional language of true/false positives/negatives, but this requires further explanation. First, we are assuming that false results are the result of the query parameters either not matching what should have been a positive hit, or matching a result that should not have been a positive hit. The false results are not a result of a flaw in the graph search algorithms. While guaranteeing the program is bug-free is outside of the scope of this project, we did confirm that that the results returned as positive (both false and negative) did correspond correctly to the GeoQuestion. Second, while it is straightforward to use public resources to confirm our false positives, it is somewhat more difficult to confirm the false negatives. Still, we spent considerable time and effort investigating both the images and relevant on-line resources to understand where there might be false negatives. We also note that the emphasis in this paper is to demonstrate the technique on the examples, and that none of the authors are subject matter experts regarding the facilities that were given in the examples.

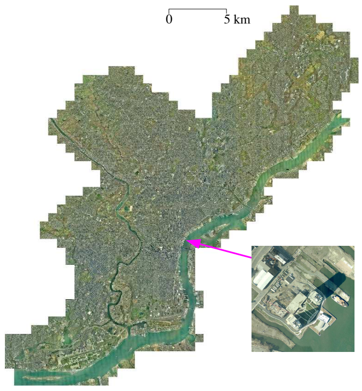
4.2.1 Power Plant Search

We defined a star-graph GeoQuestion query to search for fossil-fueled electricity generation plants as described in Figure 5. Our search definition invoked raw data provenance analysis to fetch height data, and also RGB spectral data from the imagery to differentiate black coal piles from ordinary bare earth.

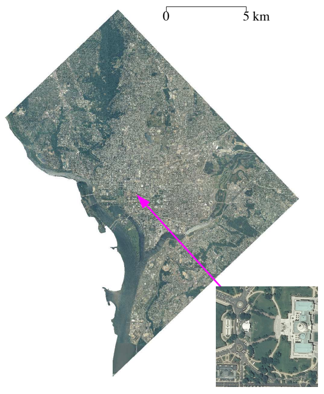
Out of 3.6 million primitive features, a total of 15 matches were returned from this search. The power plant search found two in Anne Arundel County, eight in Philadelphia and five in Washington. Six of these were actual power plants including the two in Anne Arundel County shown in Figure 6. Table 2 shows true positives, false negatives, and false positives of each region. All of the nine false positives and one false negative can be explained by difficulty differentiating transformer pads from parking lots with tall lighting structures; an improvement in the query regarding



(a) Anne Arundel County, MD

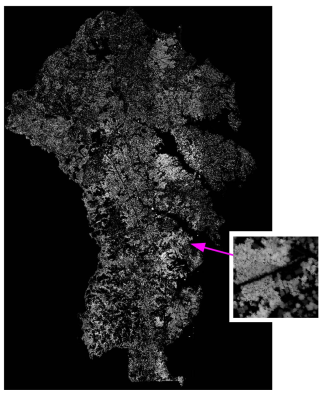


(b) Philadelphia, PA

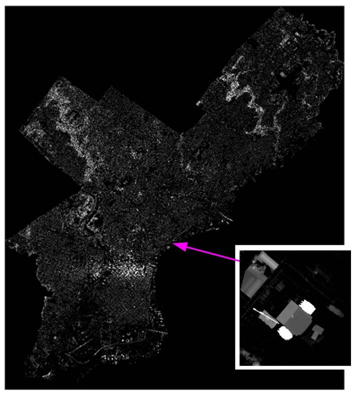


(c) Washington, DC

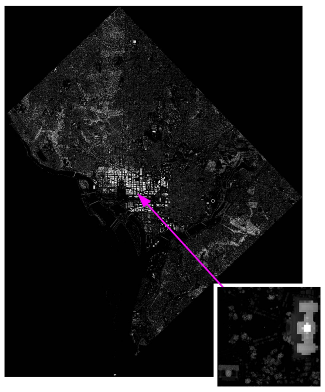
Figure 2: Raw optical imageries.



(a) Anne Arundel County, MD

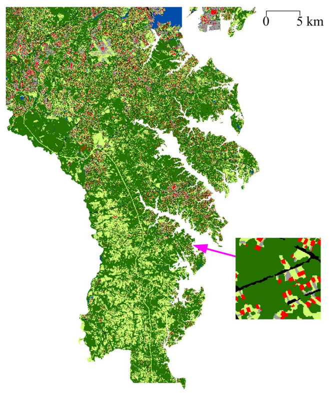


(b) Philadelphia, PA

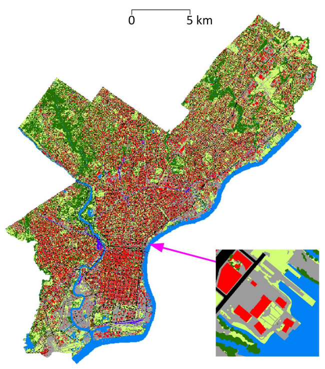


(c) Washington, DC

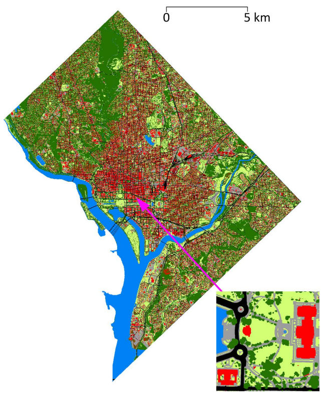
Figure 3: Height maps (Lightness corresponds to height).



(a) Anne Arundel County, MD



(b) Philadelphia, PA



(c) Washington, DC

Figure 4: Land cover maps generated by using raw optical images (Figure 2), height maps (Figure 3) and GIS data. (Brown: Dirt, Light green: Grass/shrub, Dark green: Trees, Blue: Water, Red: Buildings, Black: Roads, Grey: Other paved)

Table 1: Data set image sizes.

	Anne Arundel County, MD	Philadelphia, PA	Washington, DC	
	2007	2008	2006	2011
Optical Image	2.485×10^9 pixels	1.015×10^{11} pixels	1.462×10^9 pixels	4.340×10^8 pixels
	7,331 MB	7,669 MB	3,420 MB	721 MB
Height Map	1.470×10^9 pixels	8.980×10^9 pixels	-	4.330×10^8 pixels
	5,745 MB	2,084 MB	-	141 MB
Land Cover Map	6.661×10^9 pixels	8.982×10^9 pixels	1.271×10^9 pixels	1.462×10^9 pixels
	123 MB	8,775 MB	32 MB	35 MB
	1,204,087 nodes	1,140,821 nodes	705,942 nodes	1,326,212 nodes

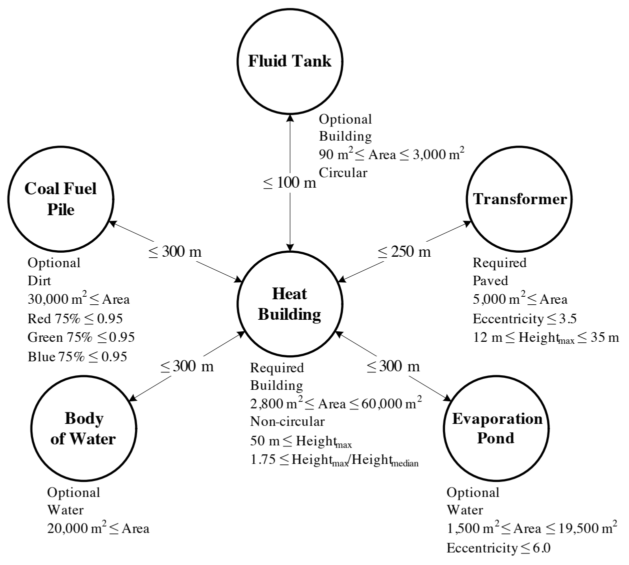


Figure 5: Power plant search definition.

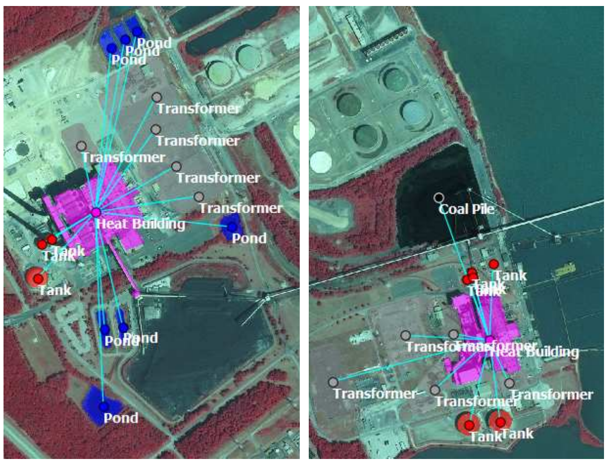


Figure 6: Power plant search outputs in Anne Arundel County, MD.

Table 2: Power plant search output evaluation.

	Anne Arundel, MD	Philadelphia, PA	Washington, DC
TP	2	2	2
FP	0	6	3
FN	0	2	0

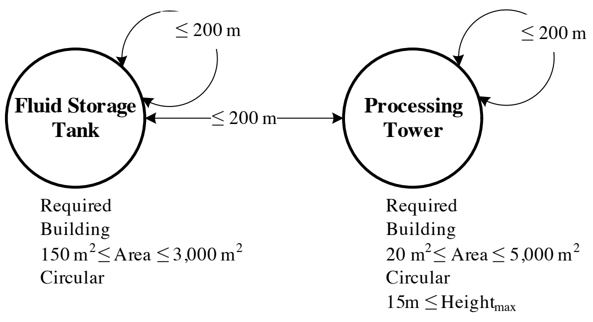


Figure 7: Refinery search definition.

this one issue would likely reduce the number of false positives to zero. The second false negative was missed because its height values of the heat building stacks were under the threshold limit.

While running this search, a total of 2,683 SearchGraph nodes were generated. This vast reduction from the 3.6 million primitive features is attributable to the constraints defining candidate nodes. Yet the number of SearchGraph nodes is still much larger than the number of returned matches; this narrowing is attributable to the graph topology indicating suitable adjacency between plant components. For example, there are 113 candidate heat building nodes, yet only 15 returned matches.

4.2.2 *Refinery Search*

We defined a connected component GeoQuestion query to search for groups of fluid storage tanks and processing tower nodes in Figure 7. Because we were interested in large refineries, we required matches to include a minimum of 10 tanks and 10 towers.

This search successfully found the two largest refinery complexes in Philadelphia, and returned nothing for Anne Arundel and Washington, where there are no large refiner-

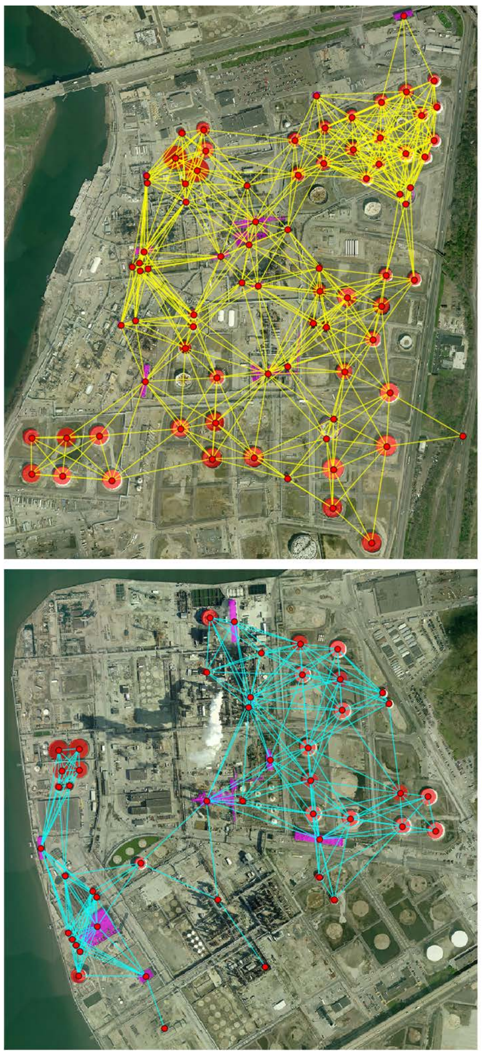


Figure 8: Detailed view of refinery search outputs.

ies. The two refinery complexes in Philadelphia are about 400 meters apart, managed by the same company. Table 3 shows true positives, false negatives, and false positives of each region. The refinery search successfully found all refineries in three regions without any false positives or false negatives. Figure 8 shows a detailed view of the returned refinery complexes.

Across all three regions, the refinery SearchGraph contains 22,282 total nodes, while the refinery search returned only two refineries comprised of 129 nodes total. This provides a striking demonstration of the power of exploiting graph topology to constrain geospatial search.

4.3 Geospatial-Temporal Search

We applied GTSGs to analyze changes in the Washington, DC area. We constructed a StoredGraph by running our graph construction program on first the 2006 land cover map, then the hospital text file, and then the 2011 land cover map. The resulting StoredGraph thus represented geospatial objects across Washington, including change relationships from 2006 to 2011, as well as relationships to known hospital locations. This StoredGraph has over 1.3 million

Table 3: Refinery search output evaluation.

	Anne Arundel, MD	**Philadelphia, PA**	**Washington, DC**
TP	0	2	0
FP	0	0	0
FN	0	0	0

nodes, over 3 million adjacency edges, and over 2 million change edges. Distance edges are added to the graph on demand at search time.

We can use the StoredGraph to perform direct change diagnosis. Because change relationships are built into the StoredGraph, a number of change analysis problems can be solved by applying graph topology constraints at search time, during the phase when the SearchGraph is constructed. For example, a node corresponding to a completely new building would appear at the head of one or more change arcs, if the nodes at the tail of the change arcs are all non-building land cover types. This implies that the building now occupies space where no building existed before. Similarly, a merged building is indicated when a building node is at the head of multiple change arcs, every change arc has a building node at its tail, $Area_{\text{removed}} = 0$ for all arcs, and the total area of these prior buildings equals the area of the new building (within a tolerance). When these conditions are met, we conclude that the entire footprint of the current building region is accounted for by previously existing buildings, and we declare it a merged building. As a third example, a building extension is indicated when a change arc exists from a prior building, the added area $Area_{\text{added}}$ is significant, the removed area $Area_{\text{removed}} \approx 0$, and this is not a merged building.

In the examples that follow, this direct change diagnosis is employed to identify significant construction across the entire city of Washington. Change categories for new, extended, and changed buildings are grouped into a single "Constructed" category. Merged buildings are excluded, thus removing numerous examples of spurious insignificant change.

4.3.1 Construction Search Near Hospitals

We applied GTSGs to investigate the question "Which hospitals are near the greatest construction activity?" This might support a study investigating whether construction-generated particulates were related to hospital contamination or lung infections.

We defined a star graph query with a hospital as the hub, constructed nodes as the only spoke type, and a maximum distance for SearchGraph edge creation of 1 km. The search query is defined in Figure 9. Hospital nodes are from a text file and building nodes are generated from the input land cover map. Constructed nodes are inferred from graph change arc topology representing change across time slices. Out of over 1.3 million nodes, a total of 73 matches were returned from this search. Figure 10 shows the match with the largest area sum; this is the match with the greatest nearby construction, measured in terms of constructed building plan area. The nearby construction is illustrated by the purple regions, and has a total building plan area of over $297,000\text{m}^2$. We note that while this serves as a good example of the capabilities of the program, most of the change represented in the

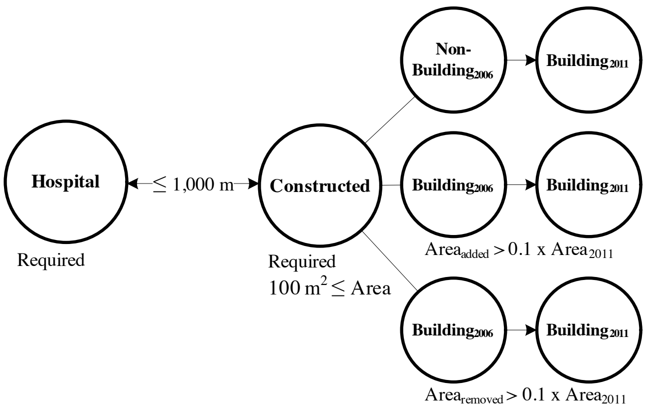


Figure 9: Construction near hospitals search definition.

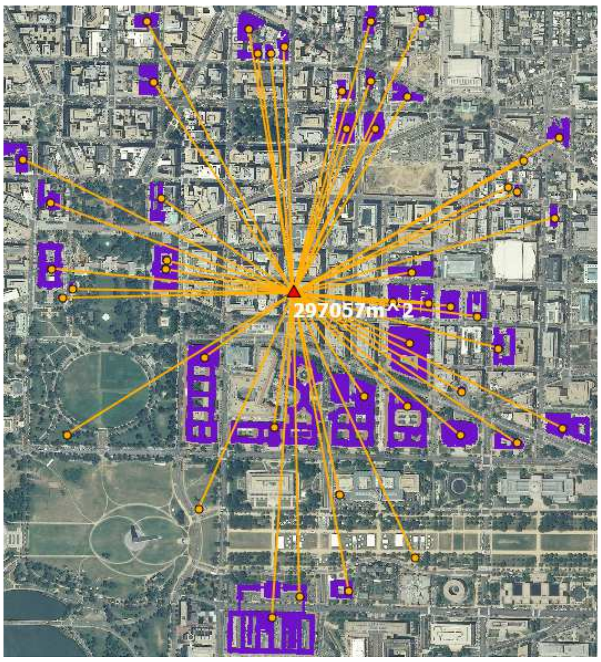


Figure 10: Construction near hospitals search output with the largest area sum.

example was probably not real change, but a manifestation of an error in the building image processing identification program that did not correctly identify the building atria in one of the time slices.

The SearchGraph contains 2,949 total nodes and 4,151 edges between hospital nodes and construction nodes, while the StoredGraph includes 1.3 million nodes. The search returned 73 outputs comprised of 4,224 nodes and 4,151 edges total. The reason we have more nodes in the search output is because a construction node can be close to multiple hospital nodes.

4.3.2 New Building Complexes

We also used our program to identify complexes of new similar buildings throughout the city, using the connected component graph search definition in Figure 11. We defined a graph query where constructed nodes were the only node type in the SearchGraph, and SearchGraph edges were constructed between all constructed node pairs with similar area ($Area_1/Area_2 \in [0.\overline{66}, 1.5]$) and similar eccentricity ($Eccentricity_1/Eccentricity_2 \in [0.\overline{66}, 1.5]$). We constrained the search to require more than 5 buildings within the distance limit for each connected component.

The search returned 27 matches. Based on the input land cover, 26 matches correctly identified new building complexes, while one match was a complex of unrelated buildings, indicating a limitation in the building similarity constraints. Figures 12a and Figure 12b show two examples. In Figure 12a, a neighborhood of new houses appears over what was bare earth in 2006. In Figure 12b, a new complex of large houses appears in the place of a complex of smaller houses that were torn down.

The SearchGraph contained 2,876 total nodes and the search returned outputs comprised of 436 nodes.

5. DISCUSSION

The execution time using Intel Core and Xeon systems running at 2.50 - 3.40 GHz with 8.0 - 32.0 GB main memory for the searches in Section 4 is listed in Table 4. The results given are for a single-threaded implementation. It is important to note that the current work has primarily focused on a demonstration of the utility of the method, and we put only a small amount of effort into optimizing the calculations. We would anticipate a formal engineering implementation of the method would be much faster.

While we have not yet performed a rigorous performance evaluation, we can make several subjective remarks about the robustness of the approach. First, because the method utilizes graph topology and object properties rather than more rigid shape-based templates, the recognition system is tolerant of wide variation in the item searched. For example, consider the two refineries found by the method, shown in Figure 8. These two refineries have widely varying component shapes, layout, and even component content. Yet our algorithm had no difficulty finding these very different sites, because it focused on the generic properties of the primitive objects, which, when taken as an ensemble, produced a strong indication of a refinery.

The refinery example also demonstrated a different sort of robustness that our process enables, that is a variation

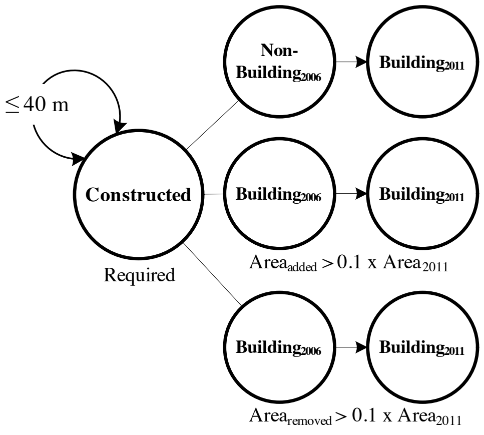
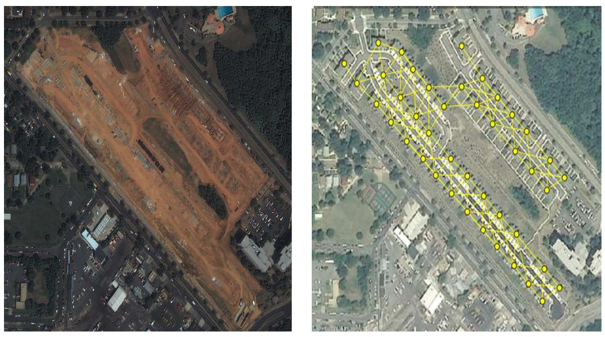


Figure 11: New building complexes search definition.

Table 4: Execution time of the spatial and spatio-temporal searches (Unit: Hours).

	Power Plant	**Refinery**	**Construction near Hospitals**	**New Building Complex**
Anne Arundel	8.7	4.6	-	-
Philadelphia	2.9	2.7	-	-
Washington	3.4	1.6	1.9	1.7


(a) A complex of new buildings.
Left: 2006 (DigitalGlobe), Right: 2011.

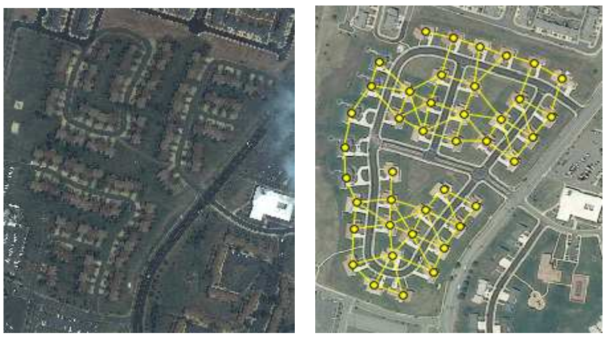

(b) A complex of new buildings,
replacing a previous complex
Left: 2006 (DigitalGlobe), Right: 2011.

Figure 12: New building complexes search output.

in match topology. It has been pointed out that many of our search examples use somewhat arbitrary numbers for what determines a match, such as the minimum of 10 tanks and towers that define our refinery. This is certainly true, but the flexibility in our match algorithms enables searches where the target is not necessarily precisely defined (such as a refinery) or the person performing the analysis doesn't know the precise specifications of the target. A wider parameter space will return more matches, including potentially more false positives, but it enables an initial search that can potentially be further refined after the initial results are examined.

The method also exhibits robustness to some preprocessing errors. For example, in Figure 6 the large power plant was affected by two errors in preprocessing. First, one coal pile was missed because preprocessing erroneously assigned this area a land cover category of grass. Second, the tall stacks of the heat building were omitted from the building footprint for some reason. Nonetheless, the power plant was successfully found, because other attributes and elements were sufficient to provide strong clues. So while the method can still be foiled by some preprocessing errors, it does not require perfect preprocessing to achieve high performance.

This also is an example of another aspect of our approach that promotes robustness. Our goal is to automatically draw a user's attention to an area of potential interest, not to recognize and explain all components of a facility. Thus the presence of refinery components in Figure 8 that remain unmarked is not a problem; after pointing out this facility to the user, they can apply their human expertise to study the site details to determine if it is in fact a refinery, and to thoroughly analyze and label all of the site components.

In this sense one can view our system as analogous to the familiar web search engine; based on simple keyword criteria, the search engine returns a set of plausible candidates, which we then quickly review and choose a subset to read in detail. For this reason we are not overly concerned about the small number of false positives and false negatives returned in the power plant search; false positives can be very quickly reviewed and discarded. False negatives can be reduced by improving search query designs and feature recognition. So while we do intend to focus future work on improving our ability to distinguish transformer pads from parking lots to drive the power plant false positives to zero, we view the current result as successful.

The use of GTSGs for detecting change also was effective, but it is important for us to emphasize that its effectiveness was more strongly dependent on the some of the subtleties associated with good preprocessing than single time-slice geospatial search. The combination of node-based analysis, shrinking to eliminate noise artifacts, change magnitude quantification, and the use of graph topology for change diagnosis provided an effective system for accepting significant change and rejecting spurious change that would have otherwise been prevalent across the data set. The change magnitude attributes and graph topology analysis provide a means of adjusting the criteria for significant change on a per-query basis, or choosing to focus on one type of change, such as returning only razed buildings or extended buildings.

It is worth noting that none of the steps in the process, from segmentation to land cover assignment to graph construction, were particularly complex in their approach. Yet the pieces fit together into an overall system that seemed fairly robust in identifying significant change, despite the presence of sensor noise and other uncertainties. This is a key feature in an approach that has the goal of being used for large data streams that will not necessarily be collected in a "perfect" manner.

Our graph-based model of change allows the analysis of change seen across multiple time steps, enabling search for stories such as the sequential initiation, progress, and completion of construction projects. However, we did not have suitable data available spanning multiple time steps, so exploration of this capability remains for future work. When we do so, we anticipate the need for new graph search algorithms which exploit temporal sequencing constraints to solve interesting problems, such as the analysis of multiple related events that must occur with consistent ordering. The graph is a natural representation for solving such problems, and we have already begun investigation into this interesting area.

While there are many approaches to image search, including those that use contextual information, it is important to point out that this approach fundamentally captures geospatial and temporal changes in a simple intuitive way. Yet while easy to understand, the sophistication of the query is almost limitless. Using the same fundamental structure, queries can be constructed that include complex topological notions of spatial connectedness (or disconnectedness) that at the same time include references to change and even activity. This is something beyond the capability of other approaches.

6. ACKNOWLEDGMENTS

We gratefully thank Jarlath O'Neil-Dunne and his group at the University of Vermont for their excellent work preparing the raw data and land cover maps used in our experiments. We also thank David Perkins, David Stracuzzi, Cindy Phillips, and Michelle Carroll for interesting technical discussions and helpful suggestions, and David Perkins in particular for suggesting the spoke-to-spoke constraint aspect of the star graph search algorithm. This work was funded by the DOE/NNSA Office of Nonproliferation Research and Development (NA-22) and the Sandia National Laboratories LDRD program. 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.

7. REFERENCES

[1] I. Biederman et al. Perceiving real-world scenes. *Science*, 177(43):77–80, 1972.

[2] L. Bruzzone and F. Bovolo. A semantic-based multilevel approach to change detection in very high geometrical resolution multitemporal images. In *2011 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 229–232, July 2011.

[3] T. C. Chisnell and G. E. Cole. Industrial components — a photo interpretation key on industry. *Photogrammetric Engineering*, 24:590âĂŞ602, March 1958.

[4] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[5] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. pages 662–673, 2002.

[6] C. Galleguillos and S. Belongie. Context based object categorization: A critical survey. *Comput. Vis. Image Underst.*, 114(6):712–722, June 2010.

[7] A. Gressin, N. Vincent, C. Mallet, and N. Paparoditis. Semantic approach in image change detection. In J. Blanc-Talon, A. Kasinski, W. Philips, D. Popescu, and P. Scheunders, editors, *Advanced Concepts for Intelligent Vision Systems*, volume 8192 of *Lecture Notes in Computer Science*, pages 450–459. Springer International Publishing, 2013.

[8] I. Herman. State of the semantic web. Key Note Speech in Semantic Days, April 2007.

[9] L. Malamboa and M. Hahnb. Lidar assisted citygml creation. *AGSE 2010*, page 13, 2010.

[10] C. E. Olson. Elements of photographic interpretation common to several sensors. *Photogrammetric Engineering*, 26(4):651–656, 1960.

[11] J. O'Neil-Dunne, J.P.M., S. MacFaden, and K. Pelletier. Incorporating contextual information into object-based image analysis workflows. In *ASPRS 2011 Annual Conference*, May 2011.

[12] J. P. O'Neil-Dunne, S. W. MacFaden, A. R. Royar, and K. C. Pelletier. An object-based system for lidar data fusion and feature extraction. In *Geocarto International*, volume 28, pages 227–242. Taylor and Francis Group, 2012.

[13] Open Geosptial Consortium. http://www.opengeospatial.org/.

[14] G. Passino, I. Patras, and E. Izquierdo. Aspect coherence for graph-based semantic image labelling. *Computer Vision, IET*, 4(3):183 –194, September 2010.

[15] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *Image Processing, IEEE Transactions on*, 14(3):294–307, March 2005.

[16] S. Ray, B. Simion, and A. D. Brown. Jackpine: A benchmark to evaluate spatial database performance. In *Data Engineering (ICDE), 2011 IEEE 27th International Conference on*, pages 1139–1150. IEEE, 2011.

[17] J. R. Ullmann. An algorithm for subgraph isomorphism. *Journal of the ACM*, 23(1):31–42, January 1976.

[18] United States Geological Survey. Hospitals in the district of columbia. http://geonames.usgs.gov/pls/gnispublic.

[19] J.-P. Watson and R. Brost. Change detection and temporal reasoning in geospatial semantic graphs, May 2012.

[20] P. Yue, L. Di, Y. Wei, and W. Han. Intelligent services for discovery of complex geospatial features from remote sensing imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 83:151–164, 2013.