INSPECT : A Dynamic Visual Query System for Geospatial Information Exploration

Sang-Joon Lee[a,b], James K. Hahn[a], Alfred M. Powell Jr[b], Geoffrey Greene[b]
[a]Institute for Computer Graphics, The George Washington University
801 22nd St. NW, Washington, DC, USA 20052
[b]Boeing Autometric, 7700 Boston Boulevard, Springfield, VA, USA 22153

ABSTRACT

This paper presents a visual information exploration tool called *INSPECT*. *INSPECT* provides geospatial information analysts with an effective way to visually filter multidimensional data and explore the underlying information contained within it. In geospatial intelligence information analyses, it is necessary to query, visualize and understand the data combined with location information. These operations are not simple since they include complex database queries of both spatial and non-spatial data. Moreover, analysts need to repeatedly query and visualize data until they reach a desirable conclusion. Using *INSPECT*, analysts are able to experimentally query the database avoiding complex database schema and visualize the results in geospatial context with minimal effort. The tools available with *INSPECT* include see-through lens visualization, relationship visualization, time varying analysis, saved lens-filter sessions, a data reachback capability, and iterative visual exploration.

Keywords: information visualization, visual query, lens, filter, visual analysis

1. INTRODUCTION

Information visualization is a powerful tool for the exploration and manipulation of large multidimensional data sets. However, accessing the underlying database and visualizing the relationships inherent in them are difficult tasks. Even for experienced analysts, understanding relationships in the data require much time and effort. For example, consider the situation in which an analyst needs to know the locations of banks with the conditions: 1) the number of account holders in a bank is more than 100,000 in both Texas and New York, 2) the average balance of all the accounts in a bank is less than \$5,000 for New York and \$3,500 for Texas, and 3) whether there exists account holders who moved into those states within the last 3 months. The relatively straightforward constraints do not lead to simple data access nor does it show where the account holders moved. Analysts need to know how the data are associated in the database and must take into consideration the locations of those banks within a spatial bound. The former problem should be considered in the database domain and the latter should be considered in the spatial domain. Every time the database is re-queried the analyst is forced to look at the data in both of these domains. Being able to work in one visual domain that incorporates both the spatial and database domains for geospatial analyses is an important advantage.

The process of drawing a conclusion from data exploration is usually based on a number of hypotheses, each of which can be tested by experiments that result in visualizations. Furthermore, each hypothesis may lead to sub-hypotheses, so the number of experiments and visualizations can grow exponentially. Consequently, testing a single hypothesis may be a very complex task. Since it is difficult for users to manage all the experimental conditions that are significant for a particular session, the visual query system should make it intuitive to build queries, transform the queried data to visual objects, and evolve current visualizations into new visualizations that enhance overall understanding.

In this paper, a dynamic visual query system is proposed, *INSPECT*, that extends the idea of the see-through lens interface[8] and the movable filter interface[14,15]. The lens interface uses a see-through visualization concept: seeing visual objects inside the lens area in ways defined by the user[8]. The system initiates queries through the visual interface using movable see-through lenses over the spatial layer where data resides. In Figure 1, as the movable lens is located and sized, a filter is attached to the lens that automatically feeds the spatial conditions acquired from the lens location to the

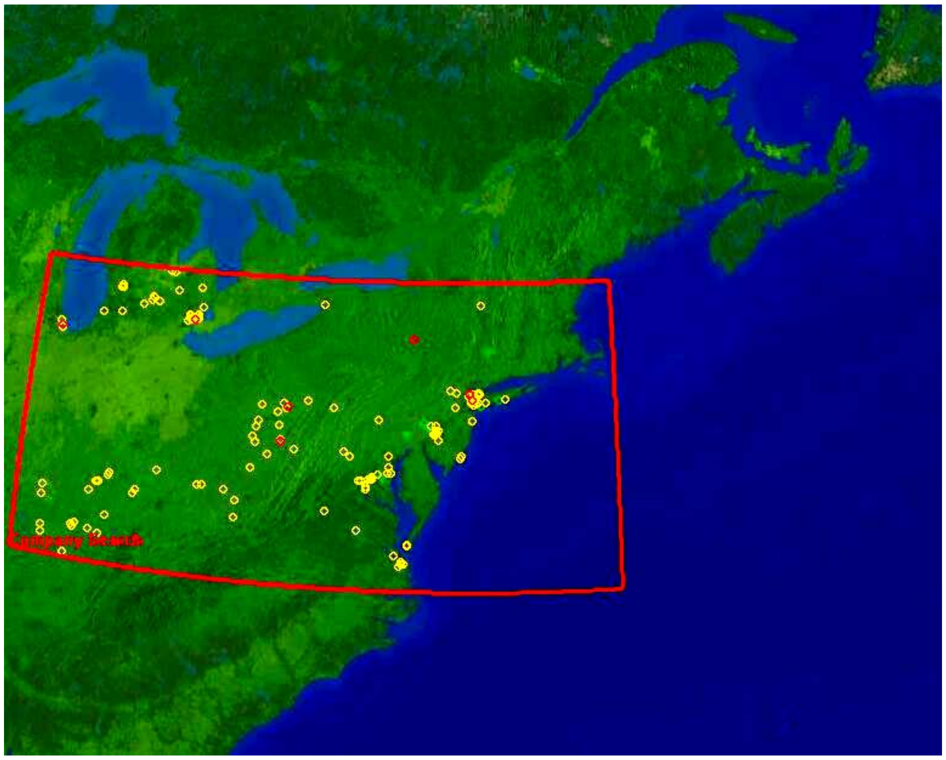


Figure 1: Example of See-through Visualization

database query. The filter queries data for the lens area and shows the visually transformed data attributes. The 'movable filter-lens visualization' in this system is one where we see the visual form of the filtered data residing inside the lens area and associated non-geospatial data in graphs or other visual displays. Unlike traditional visualization that brings the related data to a viewer for display, this system combines the interaction between the visualization and the database query to virtually couple the dataset to the geospatial visualization[14].

Once users select the filtering query for the lens, they do not have to expend any effort in manipulating the complex database schema to change regions of interest. Instead, they can work only in the visual domain by simply moving the lens to a new location. The filter re-queries the data automatically when the lens is placed in a new location. In addition, users can manipulate multiple lenses at the same time so that they can investigate several areas. Users can also produce a new lens-filter visualization simply by editing filter properties from the existing lenses or 'dragging' new filters into existing lenses.

The *INSPECT* system is built on top of a commercial geospatial data visualization and analysis tool, *EDGE*®[*]. This visualization environment provides robust capability in terms of combining imagery, maps, icons, tracks of aircraft/satellites, etc. This allows users to render visualizations in the best contextual view that can be built from the information included in the database. 'Contextual' in this sense means as close to actual locations and 'real world' views as possible.

2. RELATED WORK

Many researchers have developed approaches that provide users with intuitive graphical-interfaces to large databases[9,10]. Cooperative database interfaces[11,12], for example, present "approximate answers" when the original queries do not

[*] Registered mark of Boeing Autometric (http://www.autometric.com/products/index.cfm?content=edge_pf)

provide the desired answer. These approaches mainly improve user understanding about the data and refine erroneous queries. Our goal was to improve the overall interaction required for an analyst to arrive at definite conclusions. Related work that impacted our design is briefly discussed below.

VQE(Visual Query Environment)[1] presents a direct manipulation of visualization as well as visualization of multidimensional data. In this system, queries and visualizations are dynamically linked. VQE typically uses the model-view paradigm in the sense of software structure. This contrasts with the feed-forward sequence of a database query followed by the visualization of results that is more common in traditional systems. It is also possible to partially encapsulate users from typing in text-based query language for data filtering, aggregation, and navigation. Their filtering functionality, however, is not sophisticated enough for spatial data and has limitations when applied to a series of user hypotheses and visualizations.

Tioga-2[2] also provides features for the direct manipulation of databases. It has the advantage of having graphical representations of "programming" as well as data. Tioga-2 is based on a small set of primitive operations for data transformation and visualization. These primitives have been carefully chosen to have clear and simple semantics that can be easily composed. The primitives are the 1)"building blocks", 2) minimum language syntax, and 3) feedback according to incremental program modifications. The building blocks are quite similar to the "boxes-and-arrow" programming paradigm popularized by AVS[3], Data Explorer[4], and Khoros[5]. The advantage of using the "building block" approach is that it provides a commonly used way to manipulate data objects including filter functions. It, however, becomes complex when it is applied to geospatial data since it takes care of spatial changes and database changes separately. The system also cannot handle 'on-the-fly' changes of the visualizations and data queries that are necessary functions for most geospatial information analysis tools.

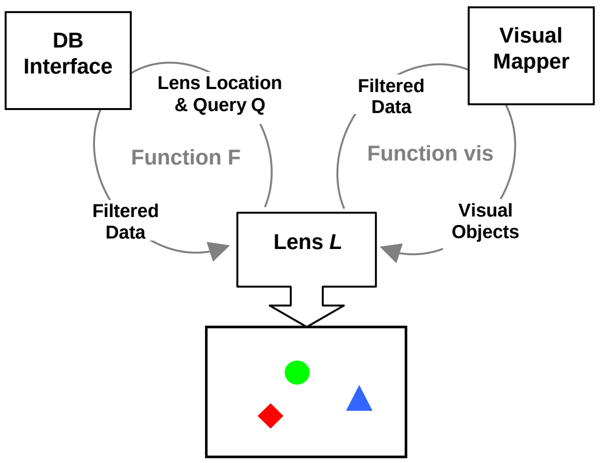


Figure 2: Flow of Visual Query

Polaris[6] presents an interface for the exploration of multidimensional databases that extends the Pivot Table[7] interface to directly generate a rich and expressive set of graphical displays. Polaris builds tables using an algebraic formalism involving the fields of the database. Each table consists of layers and panes, and each pane may present different graphics so that users can have a more effective way of comparing multi-dimensional data than the Pivot Table. This system has a unique feature of visual interaction on databases, but its functionality is limited to the applications that could be optimized using table-based visualization.

The *INSPECT* tool design was an attempt to alleviate some of the shortcomings in previous visual data query approaches to arrive at a tool more applicable for geospatial information analysis. To address some of these issues, the visual interface was designed to provide contextual views of the world while the lens-filter paradigm was used to support relational database structures. A brief explanation of the *INSPECT* components follows.

3. LENS VISUALIZATION

The *INSPECT* lens bounds the region of interest. The lens position and size are determined interactively by the user's spatial interest in the display. Within the spatial frame of the lens, information is visualized based on the queried data. To analyze information inside of the region of interest, users may manipulate data by choosing one of several supporting functions. The possible set of data to be visualized in a lens region may be large and is presumed to exist in the form of database tables. Users must manipulate the data via queries attached to the lens to turn it into useful information for the problem at hand. When the data is visualized, it is sometimes hard for users to avoid clutter in the visualization due to the enormous number of visual objects to which data items are mapped. Cluttered visualizations are not optimal for examining hypotheses and results in information overload. Filters attached to lenses are referred to as visual queries in this paper. They can be a good solution to reduce or avoid possible information overload by providing limits on the data to be used. In following sections, we will cover basic data structures from simple queries to complex queries that are the basis of filter functions. Also, a short discussion of how *INSPECT* supports a variety of analyses with the combined lens-filter visualization will be presented.

3.1 Visual Query

The visual query process encapsulates complex data schema to improve user understanding and reduce the user interface workload (Figure 2). In geospatial information systems, most user queries or actions are performed in reference to location information. Several different queries over the same location are possible and may be combined with other filters to produce a complex resultant visualization. Single queries over the several different regions are also possible. For the interactive lens-filter manipulation, effective data structures are essential for optimized performance[16]. However, every problem cannot be optimized using *INSPECT*'s generalized approach to the relational query.

Query Unit

The basic query unit is a tuple of the query language word 'select', whose arguments may be column names, table names, and table conditions.

$$Q_{simple} = (\textit{select}, \text{column name, table name, condition(s)})$$

Q_{simple} is a basic query unit and '*select*' is a key word of SQL (Standard Query Language).

One assumes that the complex query is basically built by concatenating multiple queries with logical operations (e.g. **AND**, **OR**, or **NOT**). A complex query function could be of the form,

$$Q_{\text{complex}} = Q_{\text{simple},1} \ \mathbf{LogOp} \ Q_{\text{simple},2} \ \ldots \ \mathbf{LogOp} \ Q_{simple,n}$$

where **LogOp** is a logical operation. Therefore, the general query Q could be written as in pseudo-BNF of

$$Q \rightarrow Q \mid Q + \mathbf{LogOp} + Q.$$

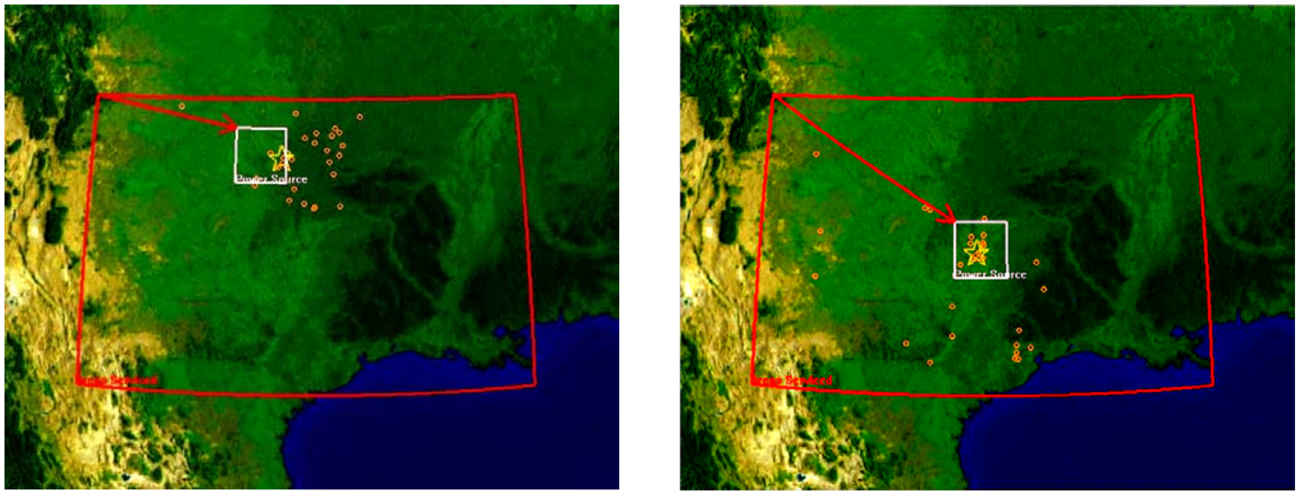
Visualization with a basic filter function

The *INSPECT* lens presents the visual representation of various single or multiple filter combinations that are derived from basic query units. However, the basic query unit can be considered a simple function of the form:

$$L = (\ \mathbf{vis}\ (F)\ ,\ \mathbf{ROI})$$

where L is a lens, **vis** is a function of displayed data objects from the filter function F which represents query Q, and the 'Region of Interest' (**ROI**) specified by the lens location. The filter function F is performed by either a simple or complex query, Q_{simple} or $Q_{complex}$, or a set of several simple or complex queries.

ROI is not a parameter in the **vis** function because it is related to the user action of specifying the lens region.
This provides a clear separation of actions to delineate location information from other literal and non-literal data actions and simplifies the query structures. This also reduces the possibility that the user will simultaneously change the data



(a) (b)

Figure 3: Relationship visualization : star shape icon shows location of a power plant inside the small pink lens and small red circles represent spots serviced by the power plant. See the different serviced spots from different power plant in (a) and (b)

region and filter functions to produce an ambiguous situation. The separation of the query building process from the identification of the bounding region improves the efficiency of the *INSPECT* tool.

3.2 Overlapping Multiple Lenses

If users are familiar with the data characteristics, they can build a complex query without seeing what each query unit shows. This presumes users have gained detailed database knowledge from seeing many basic queries. Because of the complex structure of database, its size and its large number of dimensions, users need to gain confidence about how their basic queries work and what they show. By applying queries one by one, they get a sense about the dataset and its spatial characteristics.

By overlapping the lens visualizations, users can examine how logical combinations of each filter affect the visualization in the spatial frame. Suppose that *Lens1* shows all the rental car offices that rent full size pickup trucks and *Lens2* shows all the credit card transaction locations for a specific account under investigation. When the lenses are overlapped with a logical **AND** combination of queries through the lens visualization operation, the locations where full size pickup trucks were rented to particular credit card holders will be shown. To specify the logical relationship between overlapped lenses (lens layers), a logical operation is added in the form of a binary relationship to a lens

$$L = (\ \mathbf{vis}\ (F)\ ,\ \mathbf{ROI}\ ,\ \mathbf{LogOp})$$

where **LogOp** is one of the logical operations (**AND**, **OR**, **NOT**).

For example, consider two lenses:

$$L_1 = (\ \mathbf{vis}\ (F_1),\ \mathbf{ROI}_1,\ \mathbf{AND})$$
$$L_2 = (\ \mathbf{vis}\ (F_2),\ \mathbf{ROI}_2,\ \{\ \mathbf{AND}\ |\ \mathbf{OR}\ |\ \mathbf{NOT}\ \}\ )$$

The output from **vis** (F) is a set of locations within the **ROI**, where the logical operations are applied between the output sets from the **vis** function. If F_1 and F_2 have the same column name, table name, condition variable but different data range, the system detects it and then executes a new single query that joins the two conditions with a lens logical operation. Suppose that Lens1, L_1, shows “all the population centers with populations between 10,000 and 50,000” and Lens2, L_2, shows “all the population centers with populations between 20,000 and 100,000”. The **AND** operation produces a filter where “all the population centers with populations between 20,000 and 50,000” are shown.

3.3 Relationship Visualization

Finding semantic relationships between various data using visualization is an important task for analytical data explorations. However, it has been a difficult process for analysts to explore relationships while experimentally producing new visual cues. *INSPECT* attempts to solve this difficult problem.

In most cases, users need to transfer the filtered data from one lens to the other lens to figure out, for example, a cause and affect relationship.

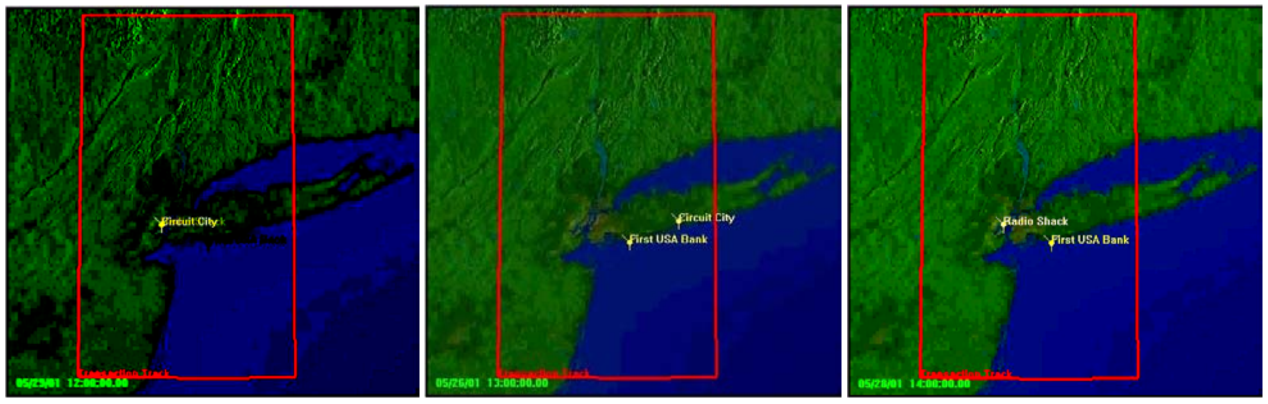
$$L1 = (\ \mathbf{vis}\ (F_1),\ \mathbf{ROI}_1,\ \{\ \mathbf{AND} \mid \mathbf{OR} \mid \mathbf{NOT}\ \}\ )$$
$$L2 = (\ \mathbf{vis}\ (F_2 \cdot F_1)),\ \mathbf{ROI}_2,\ \{\ \mathbf{AND} \mid \mathbf{OR} \mid \mathbf{NOT}\ \}\ )$$

With *INSPECT*, users may set queries hierarchically for each lens visualization. This provides users with a visual cue that a lens is displaying information dependent upon what the other lens contains. The *INSPECT* system dynamically visualizes this relationship with simple drag and drop user actions on filters that have already been set. Filter F_2 in L_2 in the above equation filters the data that F_1 filtered first. As seen in Figure 3, the small lens shows a power plant as star-shape icons and the larger lens shows areas that are served by the power plant (circles). In this cause and effect relationship setting, users may relocate the lens region or resize the lens area to see new results. Once the lens relocation is completed, the appropriate query is automatically re-applied, and the visual display updated in both lenses.

3.4 Data Reachback

For some analysis, users may need to know more about the non-spatial attributes of data before they actually filter and visualize them. If analysts know the quantitative range, or statistical attributes of values, they can avoid laborious trial and error set proper filter criteria. *INSPECT* helps overcome the difficulty associated with not knowing the data ranges a priori.

INSPECT can show data using graphs such as bar charts, pie charts, scattered plots, etc. Users can interactively select some data values in the graph and *INSPECT* can then show the spatial attributes of the data values inside lenses. For instance, in a scatter plot, a dot may represent the price and amount of copper produced in a copper mine. Users can select a certain range within the data distribution, interactively using the scatter plot and see the geographic distribution of the selected mines or check if they are present in a lens area. This feature directly helps analysts explore the relationship between the statistical attributes of data values and their spatial distribution. Conversely, *INSPECT* can show the statistical attributes of the geographic location shown in the lens visualization.



(a) (b) (c)

Figure 4: Time varying visualization by applying lenses (a) transaction at May 26, 2001 12:00PM (b) transaction at May 26, 2001 1:00PM (c) transaction at May 26, 2001 2:00PM

3.5 Supporting Tools

The following section briefly discusses three important supporting tools in *INSPECT*: time varying visualization, levels of detail, and saving user sessions or lens-filters.

Time Varying Visualization

INSPECT also visualizes data in one extended domain, time. Users can attach a clock to each lens visualization and set 'windows' of time. The lenses then display objects as a function of time. Because of the relatively expensive querying of the database server, the system should not perform the repeated work of querying and displaying data at small increments of time. In addition, another potential problem is icons may last for a relatively short periods of time compared to the entire time period of the query. To solve this problem, *INSPECT* buffers visual objects at every discrete query time and then plays them (Figure 4). The user's perception of time varying change was enhanced by adding a fade-in and fade-out, respectively, before and after an icon's appearance as an event. This precludes missing an 'event' in the data. The common coloration of paper with time was chosen as the paradigm for coloring an icon before and after an event: white shortly before the event, actual color very near event time, and yellow (aging) that fades to black after the event time.

Level of Detail and Demand-on-Detail

If a visualization maintains the same size and shape of visual objects, while zooming in or out, users will see clutter between the visual objects. The clutter impedes user recognition and can excessively magnify visual objects in the limited space of the screen without adding value in the display. For example, as a user zooms their view they expect to see more details of the visualized information rather than just magnified visual objects. *INSPECT* displays the details of predefined iconsbased on the level of zoom and the number of objects shown within the lens. This technique provides a method for de-cluttering the lens visualization while providing added information to the user.

INSPECT also supports interactive drill-down functionality for visual objects. Without zooming, users can see details about each visual object by choosing it. The information is derived from the database table the picked object is associated with in the database.

Saving session data and Lens-filter library

INSPECT provides functionality for saving data, filters, and screen images that are produced during the exploration process. *INSPECT* saves the filter and lens structures and the resultant visualization so that later users can revisit saved sessions and restart the exploration.
INSPECT can save entire sessions as well as store useful lenses and filters into a library. With the library of filters and lenses, analysts can easily construct similar or new analyses, perhaps on different datasets. This library also can be shared by a group of analysts in a collaborative distributed environment.

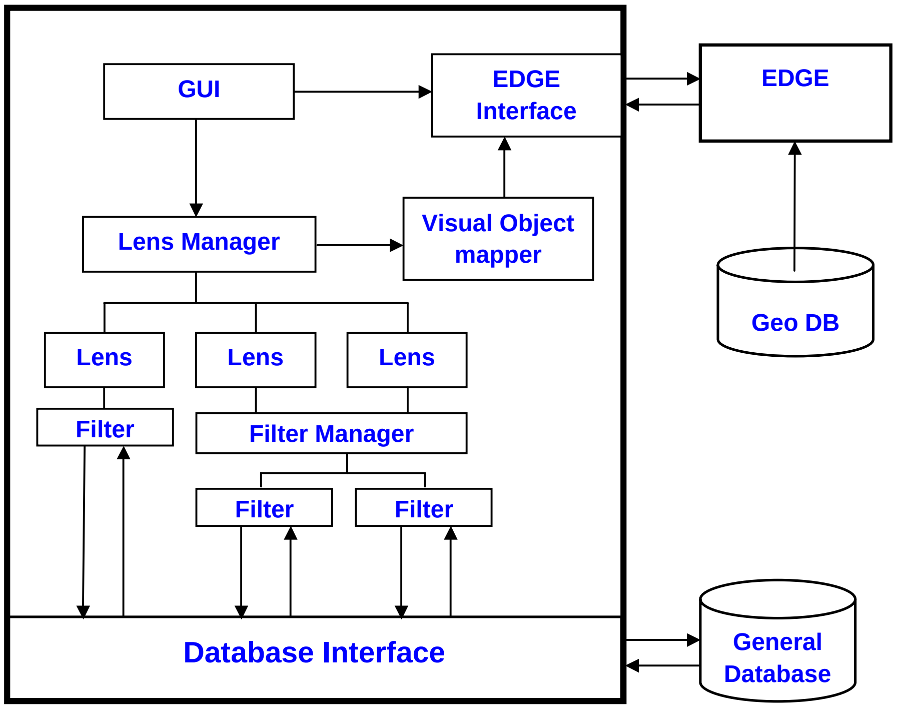


Figure 5 : INSPECT system design

4. SYSTEM

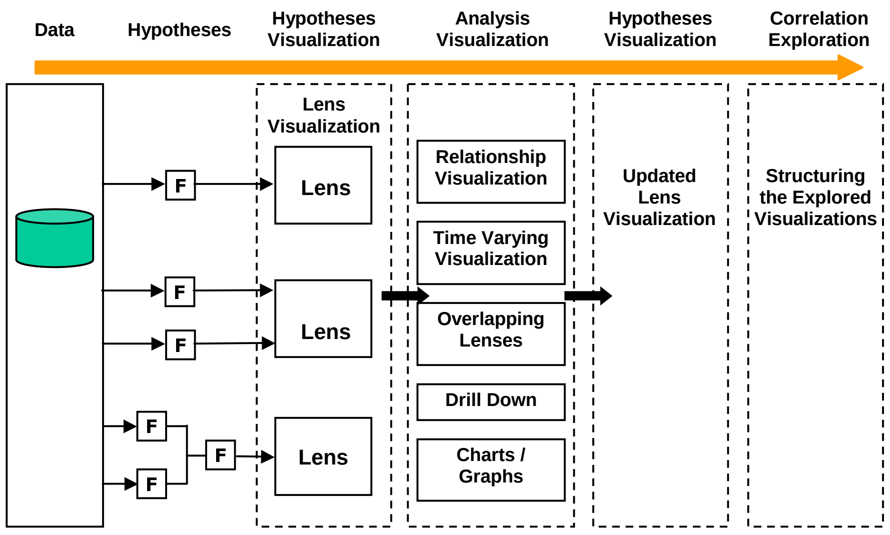
INSPECT is linked to a commercial geospatial data analysis tool, EDGE™. As seen in Figure 5, *INSPECT*'s main function can be seen as a manager for filters and lenses. The database interface for *INSPECT* is designed to work with most major relational database systems. The general database built for this research has 400,000 data entries and the geospatial database provides EDGE™ with geospatial data such as imagery, maps, terrain data, weather data, and geographic information. EDGE fuses this information into a contextual view of the location or region of interest using the information available to it. This helps users understand the geospatial aspects of problems without having to keep separate maps or other information as a crutch.

5. CONCLUSION

With several features of *INSPECT*, users could enhance their analysis work in several aspects summarized in the following.

Effective filter-lens paradigm: Users could separate their manipulation of the data in the spatial and database domains. Once users set a filter for data in the region of interest, they could concentrate their attention to what happens in the spatial domain by simply interacting with the lenses. The lenses, and their associated functions or tools, become the key to the iterative hypothesis testing that leads to well-defined decisions.

Relationship Visualization: Relationships can be viewed through lens dependencies, lens-filter combinations and graphs. Since the dependent and independent lenses are separately sized and located, multiple relationship combinations for different regions of interest can be shown. The lens regions may or may not overlap depending on the relationship. This combination of features provides a very robust view into potential geospatial relationships. For non-literal relationships, like scatter plots, the graphs serve a similar function and allow direct association of graph features with other variables or the spatial domain.



Data reachback interface: Users can also view relationships in both non-spatial data(graphs) and spatially aligned data (modeled data in the form of a gridded field, like the forecast weather) using the combined function interaction between the spatial and database domains. Interaction on either the geospatial domain or the database domain visualization dynamically affects the other. As a result, users get more flexibility from both visual feedbacks[17,18]. This capability is not traditionally available in GIS type systems.

Iterative visualization: Users can easily modify filter functions and freely apply the visualizations to regions of interest. As shown in Figures 6 and 7, the user may apply any sequence of *INSPECT* tools (graphs, lens-filter combinations, etc) to iteratively arrive at a final solution. The behavior that allows users to iteratively apply different functions while tracking the analysis steps is a unique capability of *INSPECT*.

Time varying visualization: The lens visualization extends to the time dimension. Users can view change over time within one or more open lenses situated at different locations. This is advantageous to problems with a temporal dimension.

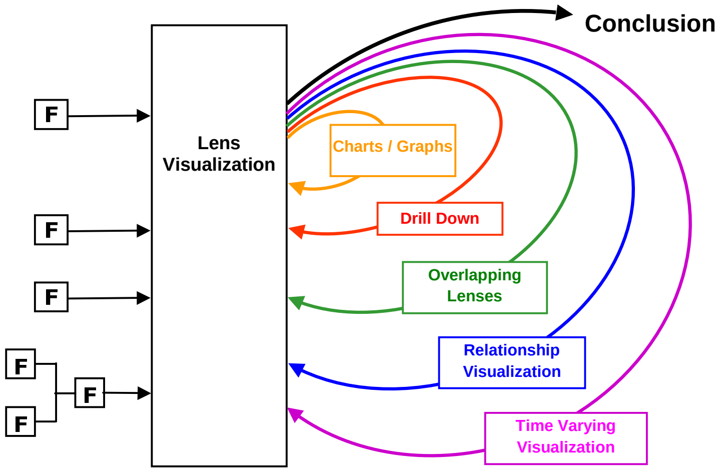


Figure 7 : Iterative visualization with INSPECT ('*F*' denotes filter associated with lenses)

Saving sessions and the lens-filter library: Users can consider a filter-lens visualization to be an encapsulation of their hypothesis testing which can consist of data queries, and visualizations. The tracking and saving of all functions, filters and lenses allows for their reuse later without having to develop them over again for the next problem. This feature is a significant time saver for the user.

6. FUTURE WORK

INSPECT helps perform visual data mining of geospatial data with minimal database actions and allows the application of the visualization scheme to virtually any location. This means that the system supports an iterative visualization procedure: a cyclic process that allows users to modify parameters, visualize results, and get feedback from the visualization in order to further modify the results, if necessary, to achieve the desired solution. Until users achieve the desired solution, they may make a number of visualizations that have their own significant meaning for the exploration process. In future work, we intend to consider a topological visualization as part of the exploration process. The topological functions will help identify chronological order and the degree of uncertainty for filter and lens visualizations at each exploration stage.
Other features under consideration are more complex relationship visualizations, bi-directional data reachback functionality with enhanced statistical tool supports, and more sophisticated levels of detail.

ACKNOWLEDGEMENTS

This study was supported and monitored by the Advanced Research and Development Activity (ARDA) and the National Imagery and Mapping Agency (NIMA) under Contract Number NMA201-01-C-0005. The views, opinions, and findings contained in this report are those of the author(s) and should not be construed as an official Department of Defense position, policy, or decision unless so designated by other official documentation.
Boeing Autometric Inc. and The George Washington University would also like to thank Phillip Zuzolo, Steve Hoffert, and Abayomi Sylvester for administratively managing the project and helping to build the database used to test the *INSPECT* functionality.

REFERENCES

1. M. Derthick, J. Kolojejchick and S. F. Roth. “An Interactive Visualization Environment for Data Exploration.” In *Proc. of Knowledge Discovery in Databases*, pp. 2-9, 1997.
2. A. Aiken, J. Chen, M. Stonebraker, and A. Woodruff, “Tioga-2: A Direct Manipulation Database Visualization Environment”, In *Proc. of the 12th International Conference on Data Engineering*, pp. 208-217, 1996.
3. C. Upson et al., “The application visualization system”, *IEEE Computer Graphics and Applications* **9(4)**, pp.30-42, July 1989,.
4. B. Lucas, G.D. Abram, N.S. Collins, D.A. Epstein, et al., “An architecture for a scientif ic visualization system”, In *Proc. of the IEEE Visualization Conference*, pp. 107-114, October 1992.
5. J. Rasure and M. Young, “An open environment for image processing software development” In *Proc. of* the *SPIE Symposium on Electronic Image Processing*, pp. 300-310, February 1992.
6. C. Stolte and P. Hanrahan. “Polaris: A System for Query, Analysis and Visualization of Multi-dimensional Relational Databases”, In *Proc. of the IEEE Symposium on Information Visualization*, pp. 5-14, 2000.
7. Microsoft Excel – User’s Guide, Microsoft, Redmond, WA 1995.
8. E. A. Bier, M. Stone, K. Pier, W. Buxton and T. DeRose. “Toolglass and Magic Lenses: The See-Through Interface”, In *SIGGRAPH ’93 Proceedings*, pp. 73-80, August 1993.
9. A. Motro, “Flex: A Tolerant and Cooperative User Interface to Databases”, *IEEE Trans. On Knowledge and Data Engineering*, **Vol. 2, No.2**, pp. 231-246, 1990.
10. D.A. Keim and V. Lum, “Gradi: A Graphical Database Interface for a Multimedia DBMS”, *Proc. Int’l Workshop on Interfaces to Database Systems*, in *Lecture Notes in Computer Science*, Springer, London, pp. 95-112, 1992.
11. S.J. Kaplan, “Cooperative Responses from a Portable Natural Language Query System”, *Artificial Intelligence*, **Vol. 19**, pp. 165-187, 1982.
12. A.K. Joshi, S.J. Kaplan, and R.M. Lee, “Approximate Responses from a Database Query System: Applications of Inferencing in Natural Language”, *Proc. 5th Int’l Joint Conf. on Artificial Intelligence*, pp. 211-212, 1977.

13. A Silberschatz, H. F. Korth, S. Sudarshan. *Database Systems : Concepts, Management, and Applications 3rd Ed*. McGraw Hill 1998.
14. K. Fishkin and M. C. Stone, “Enhanced Dynamic Queries via Movable Filters”, *Proc. CHI*, pp. 415-420, 1995.
15. K. Fishkin, M. C. Stone, and E. A. Bier, “The movable filters as a user interface tool”, *Proc. CHI*, pp. 306-312, 1994.
16. J. Favre and J. K. Hahn, "An Object Oriented Design for the Visualization of Multi-Variable Data Objects", In *Proc. of the IEEE Visualization Conference*, Washington, DC, pp 107-114, October 1994.
17. A.M. Powell, Jr and P.A. Zuzolo, “The Benefits of Visual Multi-Source, Multi-Resolution Data Analysis and Fusion”, In the *American Meteorological Society’s (AMS) 16th Conference on Interactive Information and Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology*, pp 132-135, January 2000.
18. P.A. Zuzolo and A.M. Powell, Jr, “New Technologies for Demonstrating Environmental Impacts to Society”, In the *American Meteorological Society’s Second Symposium on Environmental Applications*, pp 167-171 2000.