Online Spatial Data Analysis and Visualization System

Yun Lu, Mingjin Zhang, Tao Li, Yudong Guang, Naphtali Rishe

School of Computing and Information Sciences

Florida International University, Miami, Florida, 33199, USA

{yun,zhangm,taoli,yuguang,rishen}@cs.fiu.edu

ABSTRACT

With the exponential growth of the usage of web map services, the geo data analysis has become more and more popular. This paper develops an online spatial data analysis and visualization system, TerraFly GeoCloud, which facilitates end users to visualize and analyze spatial data, and to share the analysis results.

Built on the TerraFly Geo spatial database, TerraFly GeoCloud is an extra layer running upon the TerraFly map and can efficiently support many different visualization functions and spatial data analysis models. Furthermore, users can create unique URLs to visualize and share the analysis results. TerraFly GeoCloud also enables the MapQL technology to customize map visualization using SQL-like statements. The system is available at http://terrafly.fiu.edu/GeoCloud/.

1. INTRODUCTION

With the exponential growth of the World Wide Web, there are many domains, such as water management, crime mapping, disease analysis, and real estate, open to Geographic Information System (GIS) applications. The Web can provide a giant amount of information to a multitude of users, making GIS available to a wider range of public users than ever before. Web-based map services are the most important application of modern GIS systems. For example, Google Maps currently has more than 350 million users. There are also a rapidly growing number of geo-enabled applications which utilize web map services on traditional computing platforms as well as the emerging mobile devices.

However, due to the highly complex and dynamic nature of GIS systems, it is quite challenging for the end users to quickly understand and analyze the spatial data, and to efficiently share their own data and analysis results to others. First, typical geographic visualization tools are complicated and fussy with a lot of low-level details, thus they are difficult to use for spatial data analysis. Second, the analysis of large amount spatial data is very resource-consuming. Third, current spatial data visualization tools are not well integrated for map developers and it is difficult for end users to create the map applications on their own spatial datasets.

To address the above challenges, this paper presents TerraFly GeoCloud, an online spatial data analysis and visualization system, which allows end users to easily visualize and share various types of spatial data. First, TerraFly GeoCloud can accurately visualize and manipulate point and polygon spatial data with just a few clicks. Second, TerraFly GeoCloud employs an analysis engine to support the online analysis of spatial data, and the visualization of the analysis results. Many different spatial analysis functionalities are provided by the analysis engine. Third, based on the TerraFly map API, TerraFly GeoCloud offers a MapQL language with SQL-like statements to execute spatial queries, and render maps to visualize the customized query results.

Our TerraFly GeoCloud online spatial data analysis and visualization system is built upon the TerraFly system using TerraFly Maps API and JavaScript TerraFly API add-ons in a high performance cloud Environment. The function modules in the analysis engine are implemented using C and R language and python scripts. Comparing with current GIS applications, our system is more user-friendly and offers better usability in the analysis and visualization of spatial data. The system is available at http://terrafly.fiu.edu/GeoCloud/.

The rest of this paper is organized as follows: Section 2 presents the background and motivation; Sections 3 describes the architecture of TerraFly GeoCloud; Section 4 describes the visualization solutions in TerraFly GeoCloud; Section 5 presents a case study on the online spatial analysis; Section 6 discusses the related work; and finally Section 7 concludes the paper.

2. BACKGROUND

2.1 TerraFly

TerraFly is a system for querying and visualizing of geospatial data developed by High Performance Database Research Center (HPDRC) lab in Florida International University (FIU). This TerraFly system serves worldwide web map requests over 125 countries and regions, providing users with customized aerial photography, satellite imagery and various overlays, such as street names, roads, restaurants, services and demographic data [1].

TerraFly Application Programming Interface (API) allows rapid deployment of interactive Web applications and has been used to produce systems for disaster mitigation, ecology, real estate, tourism, and municipalities. TerraFly's Web-based client interface is accessible from anywhere via any standard Web browser, with no client software to install.

TerraFly allows users to virtually 'fly' over enormous geographic information simply via a web browser with a bunch of advanced functionalities and features such as user-friendly geospatial querying interface, map display with user-specific granularity, real-time data suppliers, demographic analysis, annotation, route dissemination via autopilots and application programming interface (API) for web sites, etc. [1][2].

TerraFly's server farm ingests geo-locates, cleanses, mosaics, and cross-references 40TB of base map data and user-specific data streams. The 40TB TerraFly data collection includes, among others, 1-meter aerial photography of almost the entire United States and 3-inch to 1-foot full-color recent imagery of major urban areas. TerraFly vector collection includes 400 million geo-located objects, 50 billion data fields, 40 million polylines, 120

million polygons, including: all US and Canada roads, the US Census demographic and socioeconomic datasets, 110 million parcels with property lines and ownership data, 15 million records of businesses with company stats and management roles and contacts, 2 million physicians with expertise detail, various public place databases (including the USGS GNIS and NGA GNS), Wikipedia, extensive global environmental data (including daily feeds from NASA and NOAA satellites and the USGS water gauges), and hundreds of other datasets [3].

2.2 Visualizing spatial data

Information visualization (or data visualization) techniques are able to present the data and patterns in a visual form that is intuitive and easily comprehendible, allow users to derive insights from the data, and support user interactions [4].
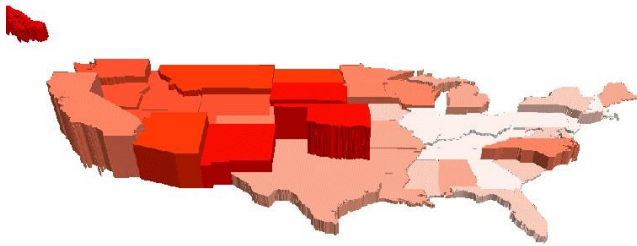
Figure 1: Population Total (height) vs. Density (color) of US

For example, Figure 1 shows the map of Native American population statistics which has the geographic spatial dimensions and several data dimensions. The figure displays both the total population and the population density on a map, and users can easily gain some insights on the data by a glance [5]. In addition, visualizing spatial data can also help end users interpret and understand spatial data mining results. They can get a better understanding on the discovered patterns.

Visualizing the objects in geo-spatial data is as important as the data itself. The visualization task becomes more challenging as both the data dimensionality and richness in the object representation increase. In TerraFly GeoCloud, we have devoted lots of effort to address the visualization challenge including the visualization of multi-dimensional data and the flexible user interaction.

TerraFly GeoCloud integrates spatial data mining and data visualization. The integration of spatial data mining and information visualization has been widely to discover hidden patterns. For spatial data mining to be effective, it is important to include the visualization techniques in the mining process and to generate the discovered patterns for a more comprehensive visual view [6].

2.3 Map Rendering

The process of rendering a map generally means taking raw geospatial data and making a visual map from it. Often it applies more specifically to the production of a raster image, or a set of raster tiles, but it can refer to the production of map outputs in vector-based formats. "3D rendering" is also possible when taking the map data as an input. The ability of rendering maps in new and interesting styles, or highlighting features of special interest, is one of the most exciting aspects in spatial data analysis and visualization.

TerraFly map render engine is a toolkit for rendering maps and is used to render the main map layers. It supports a variety of geospatial data formats and provides flexible styling options for designing many different kinds of maps, and the render speed is fast [7][8].

TerraFly map render engine is written in C++ and can be used as a web service. It uses the AGG library and offers anti-aliasing rendering with pixel accuracy. It can read different kind of file like PostGIS, TIFF rasters, .osm files, and other shape files. Packages are available for both Window and Linux [8].
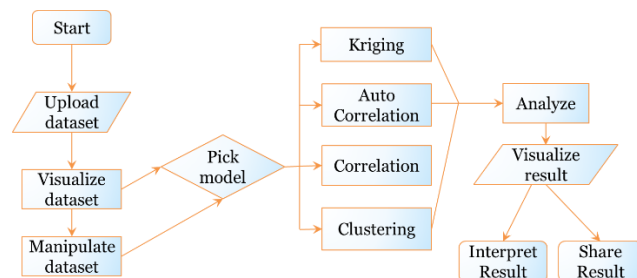
3. TerraFly GeoCloud

Figure 2 shows the system architecture of TerraFly GeoCloud. Based on the current TerraFly system including the Map API and all sorts of TerraFly data, we developed the TerraFly GeoCloud system to perform online spatial data analysis and visualization. In TerraFly GeoCloud, users can import and visualize various types of spatial data (data with geo-location information) on the TerraFly map, edit the data, perform spatial data analysis, and visualize and share the analysis results to others. Available spatial data sources in TerraFly GeoCloud include but not limited to demographic census, real estate, disaster, hydrology, retail, crime, and disease. In addition, the system supports MapQL, which is a technology to customize map visualization using SQL-like statements.
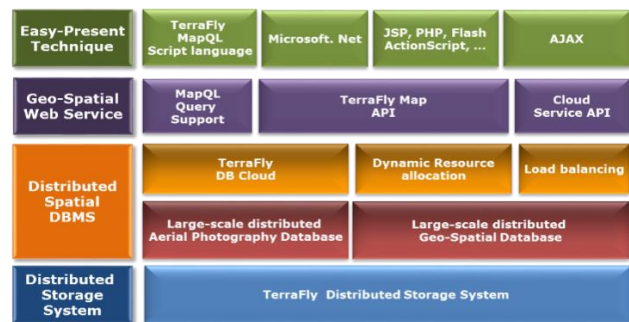

Figure 2: The Architecture of TerraFly GeoCloud

The spatial data analysis functions provided by TerraFly GeoCloud include spatial data visualization (visualizing the spatial data), spatial dependency and autocorrelation (checking for spatial dependencies), spatial clustering (grouping similar spatial objects), and Kriging (geo-statistical estimator for unobserved locations).

Figure 3: The Workflow of TerraFly GeoCloud Analysis

Figure 3 shows the data analysis workflow of the TerraFly GeoCloud system. Users first *upload datasets* to the system, or view the available datasets in the system. They can then *visualize the data sets* with customized appearances. By *Manipulate dataset*, users can edit the dataset and perform pre-processing (e.g., adding more columns). Followed by pre-processing, users can choose proper spatial analysis functions and perform the analysis. After the analysis, they can visualize the results and are also able to share them with others.
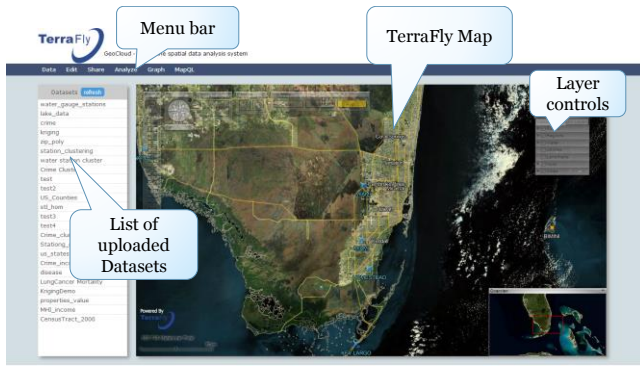
Figure 4: Interface of TerraFly GeoCloud

Figure 4 showed the interface of the TerraFly GeoCloud system. The top bar is the menu of all functions, including *Data, analysis, Graph, Share,* and *MapQL.* The left side shows the available datasets, including both the uploaded datasets from the user and the existing datasets in the system. The right map is the main map from TerraFly. This map is composed by TerraFly API, and it includes a detailed base map and diverse overlays which can present different kinds of geographical data.

TerraFly GeoCloud also provides MapQL spatial query and render tools. MapQL supports SQL-like statements to realize the spatial query, and after that, render the map according to users' inputs. MapQL tools can help users visualize their own data using a simple statement. This provides users with a better mechanism to easily visualize geographical data and analysis results.

4. Visualization in TerraFly GeoCloud

4.1 Spatial Data Visualization
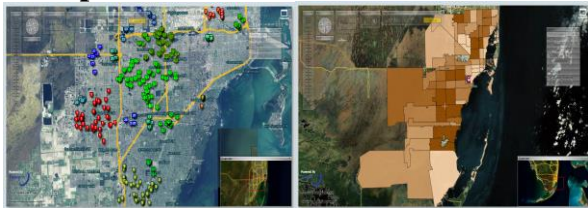
Figure 5: Spatial Data Visualization: Left subfigure: Point Data; Right subfigure: Polygon Data

For spatial data visualization, the system supports both point data and polygon data and users can choose color or color range of data for displaying. As shown in Figure 5, the point data is displayed on left, and the polygen data is displayed on the right. The data labels are shown on the base map as extra layers for point data, and the data polygons are shown on the base map for polygon data. Many different visualization choices are supported for both point data and polygon data. For point data, user can customize the icon style, icon color or color range, label value and so on. For polygon data, user can customize the fill color or color range, fill alpha, line color, line width, line alpha, label value and so on.

4.2 Spatial Data Mining Results Visualization

TerraFly GeoCloud integrates spatial data mining and data visualization. The spatial data mining results can be easily visualized. In addition, visualization can often be incorporated into the spatial mining process.

4.2.1 Spatial dependency and Auto-Correlation

Spatial dependency is the co-variation of properties within geographic space: characteristics at proximal locations that appear to be correlated, either positively or negatively. Spatial dependency leads to the spatial autocorrelation problem in statistics [9].

Spatial autocorrelation is more complex than one-dimensional autocorrelation because spatial correlation is multi-dimensional (i.e. 2 or 3 dimensions of space) and multi-directional. The TerraFly GeoCloud system provides auto-correlation analysis tools to discover spatial dependencies in a geographic space, including global and local clusters analysis where Moran's I measure is used [10].

Formally, Moran's I, the slope of the line, estimates the overall global degree of spatial autocorrelation as follows:

$$I = \frac{n}{\sum_i^n \sum_j^n w_{ij}} \times \frac{\sum_i^n \sum_j^n w_{ij}(y_i - \bar{y})(y_j - \bar{y})}{\sum_i^n (y_j - \bar{y})^2}$$

where w_{ij} is the weight, $w_{ij}=1$ if locations i and j are adjacent and zero otherwise $w_{ii}=0$ (a region is not adjacent to itself).y_i and $\bar{y}$ are the variable in the ith location and the mean of the variable, respectively. n is the total number of observations. Moran's I is used to test hypotheses concerning the correlation, ranging between −1.0 and +1.0.

Moran's I measures can be displayed as a checkerboard where a positive Moran's I measure indicates the clustering of similar values and a negative Moran's I measure indicate dissimilar values. TerraFly GeoCloud system provides auto-correlation analysis tools to check for spatial dependencies in a geographic space, including global and local clusters analysis
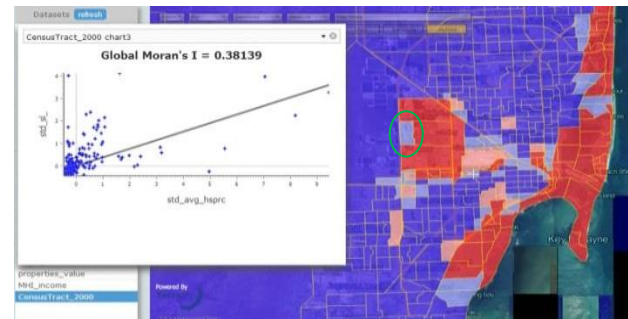
Figure 6: Average properties price by zip code in Miami

Local Moran's I is a local spatial autocorrelation statistic based on the Moran's I statistic. It was developed by Anselin as a local indicator of spatial association or LISA statistic [11]. The fact that Moran's I is a summation of individual cross products is exploited by the "Local indicators of spatial association" (LISA) to evaluate the clustering in those individual units by calculating Local Moran's I for each spatial unit and evaluating the statistical significance for each Ii. From the previous equation we then obtain:

$$I_i = z_i \sum_j^n w_{ij} z_j$$

where z_i are the deviations from the mean of y_i, and the weights are row standardized.

Figure 6 shows an example of spatial auto-correlation analysis on the average properties price by zip code data in Miami (polygon

data). Each dot here in the scatterplot corresponds to one zip code. The first and third quadrants of the plot represent positive associations (high-high and low-low), while the second and fourth quadrants represent associations (low-high, high-low). For example, the *green circle area* is in the low-high quadrants. The density of the quadrants represents the dominating local spatial process. The properties in Miami Beach are more expensive, and are in the high-high area.
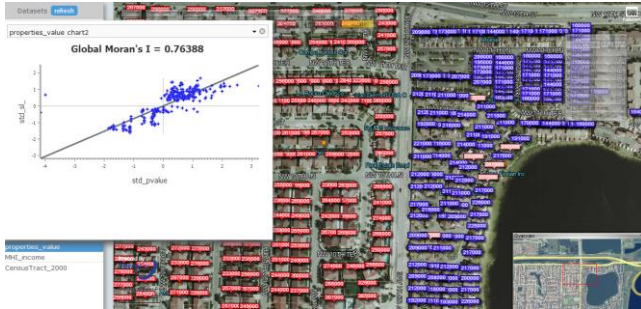

Figure 7: Properties value in Miami

Figure 7 presents the auto-correlation analysis results on the individual properties price in Miami (point data). Each dot here in the scatterplot corresponds to one property. As the figure shows, the properties near the big lake are cheaper, while the properties along the west are more expensive.

4.2.2 Spatial Data Clustering

The TerraFly GeoCloud system supports the DBSCAN (for density-based spatial clustering of applications with noise) data clustering algorithm [12]. It is a density-based clustering algorithm because it finds a number of clusters starting from the estimated density distribution of corresponding nodes.

DBSCAN requires two parameters as the input: *eps* and the minimum number of points required to form a cluster *minPts*. It starts with an arbitrary starting point that has not been visited so far. This point's neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started. Otherwise, the point is labeled as a noise point [12]. If a point is found to be a dense part of a cluster, its neighborhood is also part of that cluster. Hence, all points that are found within the neighborhood are added. This process continues until the density-connected cluster is completely identified. Then, a new unvisited point is retrieved and processed, leading to the discovery of a further cluster or noise points [13].

Figure 8 shows an example of DBSCAN clustering on the crime data in Miami. As shown in Figure 6, each point is an individual crime record marked on the place where the crime happened, and the number displayed in the label is the crime ID. By using the clustering algorithm, the crime records are grouped, and different clusters are represented by different colors on the map.

Figure 8: DBSCAN clustering on the crime data in Miami

4.2.3 Kriging

Kriging is a geo-statistical estimator that infers the value of a random field at an unobserved location (e.g. elevation as a function of geographic coordinates) from samples (see spatial analysis) [14].
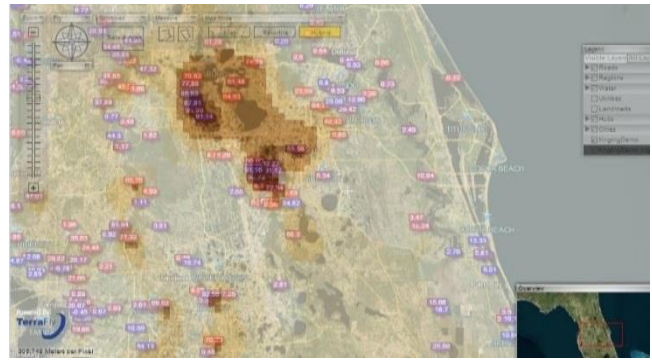

**Figure 9: Kriging data of the water level in Florida**

Figure 9 shows an example of Kriging. The data set is the water level from water stations in central Florida. Note that not all the water surfaces are measured by water stations. The Kriging results are estimates of the water levels and are shown by the yellow layer.

4.3 Customized Map Visualization (Supported by MapQL)

TerraFly GeoCloud also provides MapQL spatial query and render tools, which supports SQL-like statements to facilitate the spatial query and more importantly, render the map according users' requests. This is a better interface than API to facilitate developer and end user to use the TerraFly map as their wish. By using MapQL tools, users can easily create their own maps.

4.3.1 Implementation

The implementation of MapQL is shown in Figure 10. The input of the whole procedure is MapQL statements, and the output is map visualization rendered by the MapQL engine.
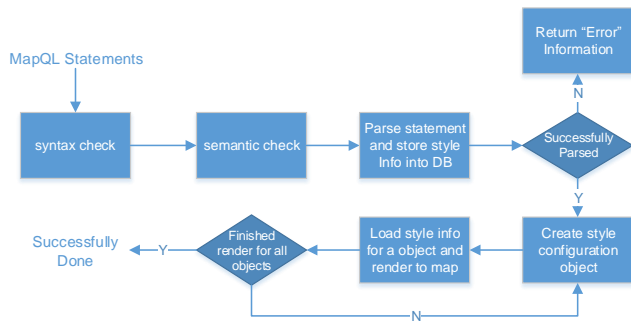
Figure 10: MapQL implementation

Shown in Figure 10, the first step is syntax check of the statements. Syntax check guarantees that the syntax conforms to the standard, such as the spelling-check of the reserved words. Semantic check ensures that the data source name and metadata which MapQL statements want to visit are correct. After the above two checks, system will parse the statements and store the parse results including the style information into a spatial database. The style information includes where to render and what to render. After all the style information is stored, system will create style configuration objects for render. The last step is for each object, load the style information form spatial database and render to the map according to the style information.

We implemented the MapQL tools using C++. For the last step which is rendering the objects to the map visualization, we employed the TerraFly map render engine [8].

For example, if we want to query the house prices near Florida International University, we use MapQL statements like this:

```
SELECT
    '/var/www/cgi-bin/house.png' AS T_ICON_PATH,
r.price AS T_LABEL,
    '15' AS T_LABEL_SIZE,
r.geo AS GEO
FROM
realtor_20121116 r
WHERE
ST_Distance(r.geo,  GeomFromText('POINT(-80.376283 25.757228)')) <
0.03;
```

There are four reseverd words in the statements, *T_ICON_PATH* , *T_LABEL*, *T_LABEL_SIZE* , and *GEO*. We use *T_ICON_PATH* to store the customized icon. Here we choose a local png file as icon. *T_LABEL* denotes that icon label that will be shown on the map, . *T_LABEL_SIZE* is the pixel size of the label; and *GEO* is the spatial search geometry.

The statement goes through the syntax check first. If there is incorrect usage of reserved words or wrong spelling of the syntax, it will be corrected or   Error information will be sent to users. For example, if the spelling of "*select*" is not correct, Error information will be sent to user. Semantic check makes sure that the data source name *realtor_20121116 and* metadata *r. price* and *r.geo* are exist and available.

After the checks, the system parsed the statements. The SQL part will return corresponding results including the locations and names of nearby objects, the MapQL part will collect the style information like icon path and icon label style. Both of them are stored into a spatial database. The system then created style configuration objects for query results. The last step is rendering all the objects on the map visualizations. The style information

needed includes icon picture and label size, and the data information includes label value and location (Lat, Long).
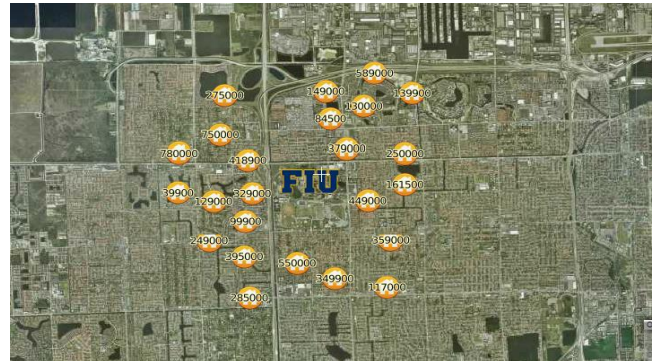

Figure 11: Query data near the point

Figure 11 shows the result of this query. Please be noticed that the unit of the distance function in all the demos is Lat-Long.

4.3.2 Other Samples

Figure 12 shows all the hotels along a certain street within a certain distance and also displays the different stars of the hotels. The MapQL statement for this query is listed below:

```
SELECT
    CASE
        WHEN star >= 1 and star < 2 THEN '/var/www/cgi-bin/hotel_1star.png'
        WHEN star >= 2 and star < 3 THEN '/var/www/cgi-bin/hotel_2stars.png'
        WHEN star >= 3 and star < 4 THEN '/var/www/cgi-bin/hotel_3stars.png'
        WHEN star >= 4 and star < 5 THEN '/var/www/cgi-bin/hotel_2stars.png'
        WHEN star >= 5 THEN '/var/www/cgi-bin/hotel_2stars.png'
        ELSE '/var/www/cgi-bin/hotel_0star.png'
    END AS T_ICON_PATH,
h.geo AS GEO
FROM
osm_fl o
LEFT JOIN
hotel_all h
ON
ST_Distance(o.geo, h.geo) < 0.05
WHERE
    o.name = 'Florida Turnpike';
```
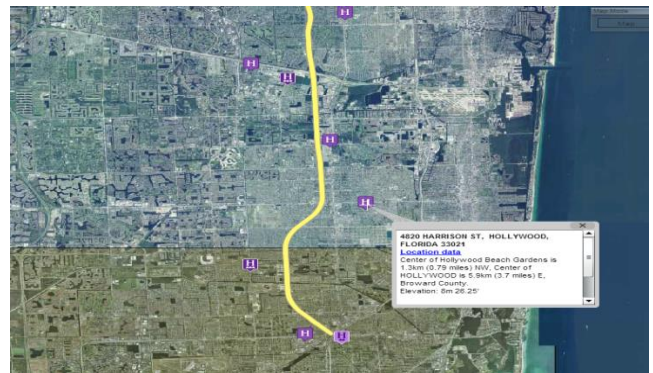

Figure 12: Query data along the line

Figure 13 shows the traffic of Santiago where the colder the color is, the faster the traffic is, the warmer the color is, and the worse the traffic is. The MapQL statement is listed below:

```
SELECT
    CASE
        WHEN speed >= 50 THEN 'color(155, 188, 255)'
        WHEN speed >= 40 and speed < 50 THEN 'color(233, 236, 255)'
        WHEN speed >= 30 and speed < 40 THEN 'color(255, 225, 198)'
        WHEN speed >= 20 and speed < 30 THEN 'color(255, 189, 111)'
        WHEN speed >= 10 and speed < 20 THEN 'color(255, 146, 29)'
        WHEN speed >= 5 and speed < 10 THEN 'color(255, 69, 0)'
        WHEN speed >= 0 and speed < 5 THEN 'color("red")'
else 'color("grey")'
    END AS T_FILLED_COLOR,
    '3' AS T_THICKNESS,
GEO
FROM santiago_traffic;
```
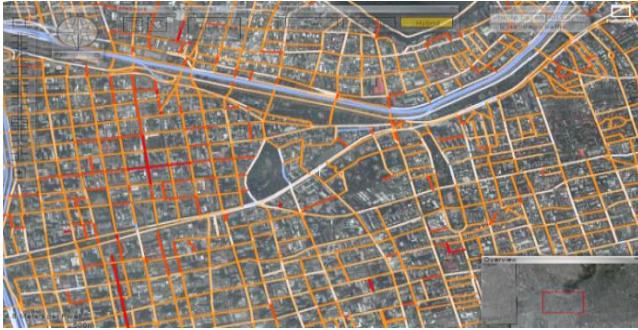
Figure 13: Traffic of Santiago

Figure 14 shows the different average incomes with in different zip codes. In this demo, users can customize the color and style of the map layers, different color stand for different average incomes. And the MapQL statement is listed below:

```
SELECT
u.geo AS GEO,
u.zip AS T LABEL,
    '0.7' AS T OPACITY,
    '15' AS T LABEL SIZE,
'color("blue")' AS T BORDER COLOR,
    CASE
        WHEN avg(i.income) < 30000 THEN 'color(155, 188, 255)'
        WHEN avg(i.income) >= 30000 and avg(i.income) < 50000 THEN 'color(233, 236, 255)'
        WHEN avg(i.income) >= 50000 and avg(i.income) < 70000 THEN 'color(255, 225, 198)'
        WHEN avg(i.income) >= 70000 and avg(i.income) < 90000 THEN 'color(255, 189, 111)'
        WHEN avg(i.income) >= 90000 and avg(i.income) < 110000 THEN 'color(255, 146, 29)'
        WHEN avg(i.income) >= 110000 and avg(i.income) < 130000 THEN 'color(255, 69, 0)'
        WHEN avg(i.income) >= 130000 THEN 'color("red")'
else 'color("grey")'
    END AS T FILLED COLOR
FROM
us zip u left join income i
ON
ST Within(i.geo, u.geo)='t'
GROUP BY
u.geo, u.zip;
```
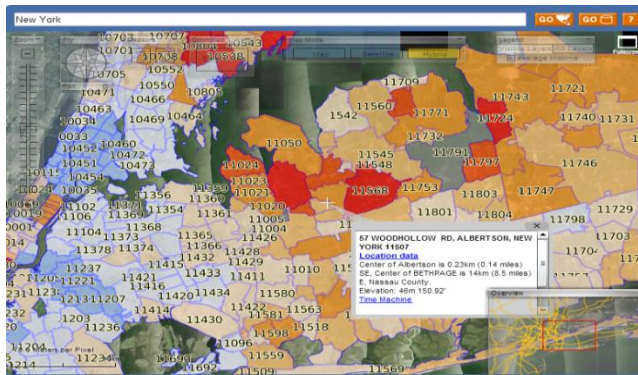

Figure 14: Income at New York

All these examples demonstrate that in TerraFly GeoCloud, users can easily create different map applications using simple SQL-like statements.

5. A Case Study

In this section, we present a case study on using TerraFly GeoCloud for spatial data analysis and visualization. As discussed in 4.2.1, we know the results of auto correlation can be shown in a scatter diagram, where the first and third quadrants of the plot represent positive associations, while the second and fourth quadrants represent negative associations. The second quadrant

stands for low-high which means the value of the object is low and the values of surrounding objects are high.

A lay user whose name is Erik who has some knowledge about the database and data analysis wanted to invest a house property in Miami with a good appreciation potential. By using TerraFly GeoCloud, he may obtain some ideas about where to buy. He believes that if a property itself has low price and the surrounding properties have higher values, then the property may have good appreciation potential, and is a good choice for investment. He wants to first identify such properties and then do a field trip with his friends and the realtor agent.
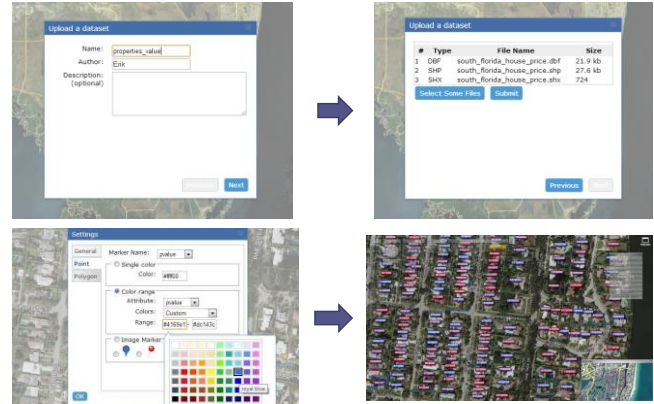

Figure 15: Data Set Upload and Visualization

To perform the task, first, Erik checked the average property prices by zip code in Miami which is shown in Figure 6. He found the *green circled area* in the low-high quadrants, which means that the average price of properties of this area is lower than the surrounding areas. Then, Erik wanted to obtain more insights on the property price in this area. He uploaded a detailed spatial data set named as *south_florida_house_price* into the TerraFly GeoCloud system as shown in Figure 15. He customized the label color range as the properties price changes. And then, he chose different areas in the *green circled area* in Figure 6 to perform the auto-correlation analysis.

Figure 16: Properties in Miami

Finally, he found an area shown in Figure 16, where there are some good properties in the low-high quadrants (in yellow circles) with good locations. And one interesting observation is, lots of properties along the road *Gratigny Pkwy* has lower prices. He was then very excited and wanted to do a query to find all the cheap properties with good appreciation potential along the *Gratigny Pkwy*. Erik composed the MapQL statements like:

```
SELECT
    CASE
        WHEN h.pvalue >= 400000 THEN '/var/www/cgi-bin/redhouse.png'
        WHEN h.pvalue >= 200000 and h.pvalue < 400000 THEN '/var/www/cgi-bin/bluehouse.png'
        WHEN h.pvalue >= 100000 and h.pvalue < 200000 THEN '/var/www/cgi-bin/greenhouse.png'
```

```
        ELSE '/var/www/cgi-bin/darkhouse.png'
    END AS T ICON PATH,
h.geo AS GEO
FROM
osm fl o
LEFT JOIN
south_florida_house_price h
ON
ST Distance(o.geo, h.geo) < 0.05
WHERE
    o.name = 'Gratigny Pkwy' AND
    h.std pvalue<0 AND
    h.std sl pvalue>0;
```


Figure 17: MapQL results

The Figure 17 presents the final results of the MapQL statements. Finally, Erik sent the URL of the map visualization out by email, waiting for the response of his friends and the realtor agent.
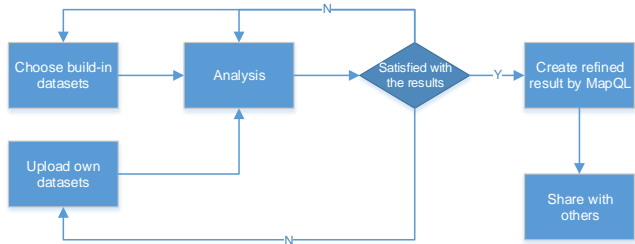

Figure 18: The flow path of Erik case

Figure 18 illustrates the whole workflow of the case study. In summary, Erik first viewed the system build-in datasets, conducted the data analysis, and then he identified properties of interest. He then composed MapQL statements to create his own map visualization to share with his friends. The case study demonstrates that TerraFly GeoCloud supports the integration of spatial data analysis and visualization and also offers user-friendly mechanisms for customized map visualization.

6. Related work and products

In the geospatial discipline, web-based GIS services can significantly reduce the data volume and required computing resources at the end-user side [16][17]. To the best of our knowledge, TerraFly GeoCloud is one of the first systems to study the integration of online visualization of spatial data, data analysis modules and visualization customization language.

Various GIS analysis tools are developed and visualization customization languages have been studied in the literature. ArcGIS is a complete, cloud-based, collaborative content management system for working with geographic information. But systems like ArcGIS and Geoda focus on the content management and share, not online analysis [18][19]. Azavea has many functions such as optimal Location find, Crime analysis, data aggregation and visualization. It is good at visualization, but has very limited analysis functions [20].

Various types of solutions have been studied in the literature to address the problem of visualization of spatial analysis [19]. However, on one hand, good analysis visualization tools like Geoda and ArcGIS do not have online functions. To use them, users have to download and install the software tools, and download the datasets. On the other hand, good online GIS systems like Azavea, SKE, and GISCloud have limited analysis functions. Furthermore, none of above products provides a simple and convenient way like MapQL to let user create their own map visualization [21][22].

The related products are summarized in Table 1. Our work is complementary to the existing works and our system also integrates the data mining and visualization.

Table 1: GIS Visualization Products

Name	Website	Product features description	Comments
ArcGIS Online	http://www.arcgis.com	http://www.arcgis.com ArcGIS Online is a complete, cloud-based, collaborative content management system for working with geographic information.	No online Analysis, focus on the content management and share.
Azavea	http://www.azavea.com/ products/	optimal Location find, Crime analsis, data aggregated and visualized	Good visualization. Very limited Analysis functions
SKE	http://www.skeinc.com/ GeoPortal.html	Spatial data Viewer	Focus on the spatial data viewer.
GISCloud	http://www.giscloud.com	with few analysis (Buffer , Range , Area , Comparison , Hotspot , Coverage , Spatial Selection)	Very limited simple analysis.
GeoIQ	http://www.geoiq.com/ http://geocommons.com/	filtering, buffers, spatial aggregation and predictive	Focus on GIS, very good Visualization and interactive operation. Very limited and simple analysis: currently provide predictive(Pearsons Correlation).

7. CONCLUSIONS AND FUTURE WORK

Web map services become increasingly widely used for various commercial and personal purposes. GIS application needs to be able to easily analyze and visualize spatial data and satisfy the increasing demand of information sharing. This paper presents a solution, TerraFly GeoCloud, an online spatial data analysis and visualization system, to address the challenges. TerraFly GeoCloud is built upon the TerraFly Geo spatial database, to offer a convenient way to analyze geo spatial data, visualize the results, and share the result by a unique URL. Our system also allows users to customize their own spatial data visualization using a SQL-like MapQL language rather than writing codes with Map API.

In our future work, we will research and develop an extra layer between end users who have limit knowledge in writing SQL statements and the MapQL, a query composing interfaces for the MapQL statements, to facilitate lay users to create their own map visualizations. Also, we will improve the scale of TerraFly GeoCloud, conduct large-scale experiments and employ

distributed computing as additional mechanisms for optimizing the system. In addition, we will explore how to apply the principle of MapQL to other applications that share similar characteristics with web GIS services.

8. ACKNOWLEDGMENTS

9. REFERENCES

[1] Rishe, N., Chen, S. C., Prabakar, N., Weiss, M. A., Sun, W., Selivonenko, A., & Davis-Chu, D. (2001, April). TerraFly: A high-performance web-based digital library system for spatial data access. In The 17th IEEE International Conference on Data Engineering (ICDE), Heidelberg, Germany (pp. 17-19).

[2] Rishe, N., Sun, Y., Chekmasov, M., Selivonenko, A., & Graham, S. (2004, December). System architecture for 3D terrafly online GIS. In Multimedia Software Engineering, 2004. Proceedings. IEEE Sixth International Symposium on (pp. 273-276). IEEE.

[3] Rishe, N., Gutierrez, M., Selivonenko, A., & Graham, S. (2005). TerraFly: A tool for visualizing and dispensing geospatial data. Imaging Notes, 20(2), 22-23.

[4] Spence, R., & Press, A. (2000). Information visualization.

[5] Old, L. J. (2002, July). Information Cartography: Using GIS for visualizing non-spatial data. In Proceedings, ESRI International Users' Conference, San Diego, CA.

[6] Yi Zhang and Tao Li. DClusterE: A Framework for Evaluating and Understanding Document Clustering Using Visualization. ACM Transactions on Intelligent Systems and Technology, 3(2):24, 2012.

[7] Teng, W., Rishe, N., & Rui, H. (2006, May). Enhancing access and use of NASA satellite data via TerraFly. In Proceedings of the ASPRS 2006 Annual Conference.

[8] Wang, H. (2011). A Large-scale Dynamic Vector and Raster Data Visualization Geographic Information System Based on Parallel Map Tiling.

[9] De Knegt, H. J., Van Langevelde, F., Coughenour, M. B., Skidmore, A. K., De Boer, W. F., Heitkönig, I. M. A., ... &Prins, H. H. T. (2010). Spatial autocorrelation and the scaling of species-environment relationships. Ecology, 91(8), 2455-2465.

[10] Li,Hongfei; Calder, Catherine A, "Beyond Moran's I: Testing for Spatial Dependence Based on the Spatial Autoregressive Model". Geographical AnalysisCressie, Noel (2007).

[11] Anselin, L. (1995). Local indicators of spatial association—LISA. Geographical analysis, 27(2), 93-115.

[12] Ester, M., Kriegel, H. P., Sander, J., &Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. ACM SIGKDD.

[13] Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (1998). Density-based clustering in spatial databases: The algorithm gdbscan and its applications. Data Mining and Knowledge Discovery, 2(2), 169-194.

[14] Stein, M. L. (1999). Interpolation of spatial data: some theory for kriging. Springer Verlag.

[15] Bilodeau, M. L., Meyer, F., & Schmitt, M. (Eds.). (2005). Space: Contributions in Honor of Georges Matheron in the Fields of Geostatistics, Random Sets, and Mathematical Morphology (Vol. 183). Springer Science+ Business Media.

[16] Xiaoyan Li, Sharing geoscience algorithms in a Web service-oriented environment, Computers & Geosciences Volume 36, Issue 8, August 2010

[17] Fotheringham, S., & Rogerson, P. (Eds.). (2004). Spatial analysis and GIS. CRC Press.

[18] Johnston, K., Ver Hoef, J. M., Krivoruchko, K., & Lucas, N. (2001). Using ArcGIS geostatistical analyst (Vol. 380). Redlands: Esri.

[19] Anselin, L., Syabri, I., & Kho, Y. (2006). GeoDa: An introduction to spatial data analysis. Geographical analysis, 38(1), 5-22.

[20] Boyer, D., Cheetham, R., & Johnson, M. L. (2011). Using GIS to Manage Philadelphia's Archival Photographs. American Archivist, 74(2), 652-663.

[21] Hearnshaw, H. M., & Unwin, D. J. (1994). Visualization in geographical information systems. John Wiley & Sons Ltd.

[22] Boyer, D. (2010). From internet to iPhone: providing mobile geographic access to Philadelphia's historic photographs and other special collections. The Reference Librarian, 52(1-2), 47-56.