Journal of Location Based Services
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t744398445

Challenges in data integration and interoperability in geovisual analytics

Ulanbek D. Turdukulov[a]; Connie A. Blok[a]; Barend Köbben[a]; Javier Morales[a]
[a] Department of Geo-Information Processing, International Institute for Geo-Information Science and Earth Observation (ITC), NL 7500AA Enschede, The Netherlands

Online publication date: 10 December 2010

To cite this Article Turdukulov, Ulanbek D. , Blok, Connie A. , Köbben, Barend and Morales, Javier(2010) 'Challenges in data integration and interoperability in geovisual analytics', Journal of Location Based Services, 4: 3, 166 — 182
To link to this Article: DOI: 10.1080/17489725.2010.532815
URL: http://dx.doi.org/10.1080/17489725.2010.532815

Journal of Location Based Services
Vol. 4, Nos. 3–4, September–December 2010, 166–182

Taylor & Francis
Taylor & Francis Group

Challenges in data integration and interoperability in geovisual analytics

Ulanbek D. Turdukulov, Connie A. Blok, Barend Köbben* and Javier Morales

Department of Geo-Information Processing, International Institute for Geo-Information Science and Earth Observation (ITC), P.O. Box 6 Enschede, NL 7500AA Enschede, The Netherlands

(*Received 26 April 2010; final version received 30 July 2010; accepted 14 October 2010*)

Geographic information technologies are evolving from stand-alone systems to a distributed model of independent web services. In parallel, voluminous geographic data are being collected with modern data acquisition techniques such as remote sensing and personal navigation devices. There is an urgent need for effective and efficient methods to integrate and explore relationships between remote sensing and trajectory datasets. When it comes to integration, one would commonly rely on a conventional chain of GIS operations to match trajectory locations to grid values: download grid data, georeference, match each trajectory record to a corresponding image cell, perform overlay, extract cell values for a given location and time and compose values into a resulting table. If one has to deal with large and dynamic spatio-temporal data sets, this approach is clearly unmanageable. We propose an alternative approach: a four-layered system architecture that utilises web services for the integration of trajectory and remote sensing data. We demonstrate how this integration service can be embedded into distributed components for manipulation, analysis and visualisation of geospatial trajectories, using Antarctic iceberg trajectories and wind data as a case study. The prototype can be accessed on the web. Future research will include optimisation and integration of more variables extracted from grid data sets, but the main focus will be on extension of the analytical component by implementing more mechanisms to discover patterns in the integrated data set, and on better visualisation.

Keywords: geovisual analytics; interoperability; spatio-temporal data integration; web services; Antarctic icebergs

1. Introduction

Voluminous geographic data are being collected with modern data acquisition techniques such as GPS, remote sensing, location-aware services and internet-based volunteered geographic information. In parallel, geographic information technologies, as all information technologies, are evolving from stand-alone systems into a distributed model of independent web services.

*Corresponding author. Email: kobben@itc.nl

ISSN 1748–9725 print/ISSN 1748–9733 online

DOI: 10.1080/17489725.2010.532815
http://www.informaworld.com

Certainly, both trends have affected the fastest growing collection of spatio-temporal information: remote sensing (RS) data. RS data providers and users are adopting new data exchange standards and protocols. Examples are the Open-source Project for a Network Data Access Protocol (OPeNDAP) and Web Coverage Services (WCS). These standards enable the creation of web services to exchange RS data and extract spatial and temporal subsets. As a result, large time-series are now offered as web services (see, e.g.: http://nomads.ncdc.noaa.gov/data.php).

Most likely, the second largest archive of spatio-temporal data consists of trajectories: traces of object movement in space and time. Trajectory data have dramatically increased due to a growing number of personal navigation devices (PNDs), miniaturising of tracking devices and their use in applications such as tracking of animals, goods and cars. Proliferation of Web2.0, the popularity of social networks with volunteered geographic information and the use of remote sensing for tracking objects also contributes to a growing volume of trajectory data.

Object movements are influenced by various factors that impact and constraint their movements. To study these impacts, RS data play an important role. Thus, there is a need for effective and efficient methods to integrate and explore relationships between both datasets. Integration of related data sets provides opportunities to find answers to *Why?* questions, in addition to the well-known *Where?*, *What?* and *When?* questions to geographic data. Examples where data integration can be useful are: trajectories of hurricanes with other weather phenomena; iceberg movements with temperature images to study climate change; wild animal trajectories need to be integrated with vegetation conditions to investigate the cause of migrations; car movement data can be integrated with air pollution data to study impacts and route planners that integrate the weather data (similar to the recently published service at http://www.wunderground.com/roadtrip/ – but unfortunately in this example the integration mechanism is not known).

When it comes to integration of grid data with trajectories, one would commonly rely on a conventional chain of GIS operations to match trajectory locations to grid values: download grid data, georeference them, match each trajectory record to a corresponding image cell (or cells), perform overlay, extract cell values for a given location and time and compose values into a resulting table. Clearly, when dealing with large and dynamic spatio-temporal data sets, this approach is unmanageable.

The importance of geovisual analytics to explore large sets of geospatial data cannot be overemphasised (Andrienko *et al*. 2007, 2008). However, at present, web services are not widely used in geovisual analytics (an example can be found in Kramis *et al*. 2009), especially web services dealing with both RS and trajectory data. Since in the future more RS and trajectory data will be offered as web services, an important functionality of any geovisual analytics environment will be the ability to consume these services. The data integration issue in fact points to a larger set of infrastructural problems in geovisual analytics: the existing systems rely heavily on monolithic geospatial data processing systems and lack effective support for distributed and heterogeneous computing environments (Keim *et al*. 2006).

We emphasise that the objective of this article is not to present a fully developed geovisual analytics environment, but rather to demonstrate an example of utilising web services for trajectory and grid data integration and to show how this integrating service can be embedded into a distributed architecture for the manipulation and visualisation of trajectory data. Therefore, the remainder of this article is organised

as follows. We start in Section 2 with discussion of an overall system architecture and the role of integrating web services in this architecture. We continue in Section 3 by applying the proposed architecture and web services into a case study. We end in Section 4 with discussion and an outlook for a future work.

2. Overall system architecture

A distributed visual analytics architecture requires the coupling of various components. For instance, trajectory data are usually large; they need to be pre-processed, stored and manipulated somewhere, usually in the DataBase Management System (DBMS). A data integration layer should be able to access remote servers containing grid data. It also needs to communicate with the DBMS in order to receive input trajectory data, either interpolated or aggregated to certain spatio-temporal intervals. The visualisation layer has to be coupled with query mechanisms and computational methods (such as clustering, classification and association rule mining) to summarise data, accentuate structures and help users explore and understand patterns and relations. Considering the above information, we propose a four-layered system architecture (Figure 1) and below, we will discuss details of the layers of the system.

2.1. *Data integration layer*

The main purpose of the data integration layer is to access remote servers containing grid data and the trajectory data. There are two main architectures, one adopted by Unidata and the other proposed by the Open Geospatial Consortium (OGC), that have similar objectives of making spatial data freely and widely available through web services. However, the technologies of these organisations have been developed within their own realms.

Unidata has over 20 years experience; it includes almost all grid data providers and contains the largest grid data resources. In this article we choose the Unidata architecture as model to build up an integration system of trajectory and grid data,

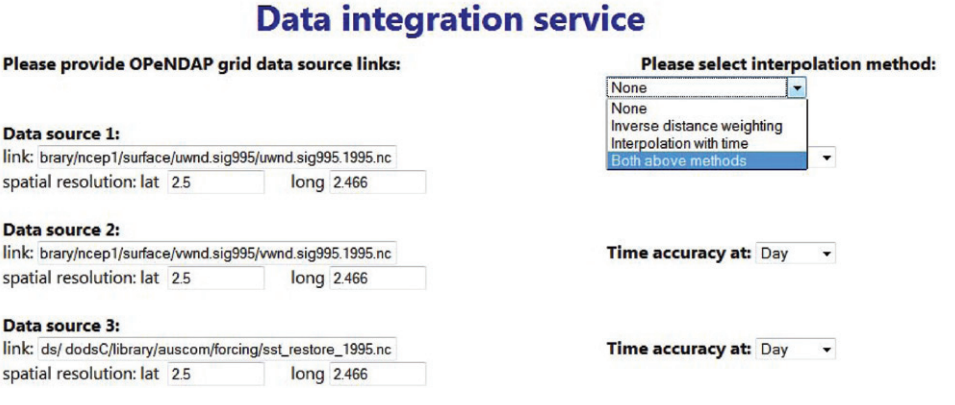

Figure 1. Proposed system architecture.

also because there are various protocols, supporting tools and client software available. OPeNDAP is the most popular protocol in this environment, with many available clients and servers.

The OPeNDAP protocol (http://www.opendap.org) is based on HTTP. OPeNDAP includes standards for encapsulating structured data, annotating the data with attributes and adding semantics that describe the data. An OPeNDAP client (which could be even a browser although this gives limited functionality) sends requests to an OPeNDAP server, and receives various types of documents or binary data as a response (see Section 3.3 for URL examples of OPeNDAP requests). An OPeNDAP server can serve an arbitrarily large collection of data. Data on the server is often in HDF or NetCDF format. Compared to ordinary file transfer protocols (e.g. FTP), a major advantage of using OPeNDAP is the ability to retrieve subsets of files, and also the ability to aggregate data from several files in one transfer operation.

Using one of the OPeNDAP clients, Integrated Data Viewer (IDV, http://www.unidata.ucar.edu/) software, the data integration layer has been implemented. It is a web service that consists of three parts (Figure 2):

- First, a user interface offers HTML forms with: (a) a button to browse to a trajectory data source; (b) a URL to a grid data source on a remote server; (c) options to choose types of data on a remote server if the source contains several data types and (d) interpolation methods that can be used in case there are missing values in the RS data at the requested trajectory locations.
- Second, there is a the server side Python script that transforms user inputs and initialises IDV, a thick client to access remote grid data. An IDV client supports the OPeNDAP protocol, but it does not have default options for integration of trajectories.
- Third, an additional IDV script to control grid data manipulation using input trajectories: it finds RS cells that match the times and locations of any trajectory, extracts a small subset of the grid, interpolates if necessary and then extracts pixel values of that subset.

All the above-mentioned integration takes place on the server side: therefore, the actual load on the client side is minimal. A typical result of the data integration operation is a table that contains trajectory data, such as locations, times and other attributes and cell values extracted from the corresponding grid data source(s). If trajectory data – in addition to the location information – contains size (e.g. the

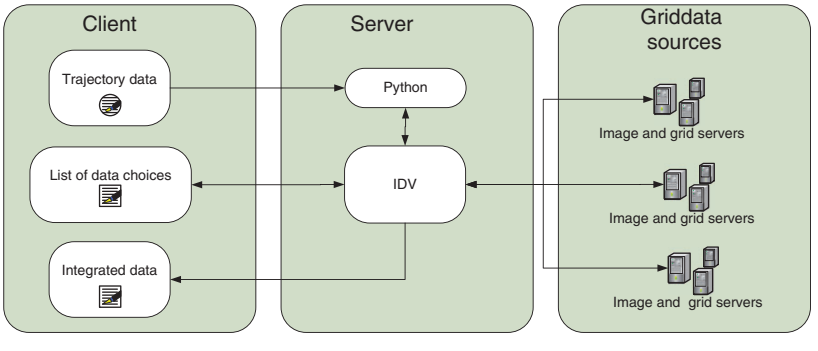

Figure 2. Data integration layer.

size of an iceberg), then grid cell values can also be aggregated, by extracting the minimum, maximum and/or average value of cells corresponding to the object's size.

2.2. *DBMS and analytics layer*

The DBMS and analytics layer for managing trajectories consists of three groups of database functions. The first group provides support for data pre-processing. Data pre-processing involves functions to clean a data set from inconsistencies (empty values, data duplicates, outliers) and to perform interpolation/aggregation of trajectories to certain spatio-temporal intervals. The second group of database functions extracts general and domain specific information about trajectories. Examples of general characteristics are travelled time, travelled distance, speed and direction. Domain-specific characteristics of a trajectory might contain information on events, such as appearances, disappearances and splits.

The last group of backend functions supports the discovery of patterns, i.e. patterns that show collective movement behaviours. Pattern extraction helps users to understand the dataset, since patterns reveal some of the motion laws that are hidden in the complexity of a trajectory representation (Giannotti and Pedreschi 2008). From a geovisual analytics perspective, this last group of database functions is of great importance.

There are many methods that can be applied to detect interesting patterns in trajectory data. Detection can be based on similarity of motion parameters, i.e. similarity in route, lifespan, speed and direction (Pelekis *et al*. 2007). Other approaches that have been proposed are a two-dimensional matrix of trajectory vectors and use of clustering methods (Rinzivillo *et al*. 2008), or applying a spatial network constraint as parameter to calculate similarity in trajectories (Tiakas *et al*. 2009).

Laube and Imfeld (2002) introduced the analysis concept RElative MOtion (REMO). The analysis is based on the comparison of motion attributes of point objects (e.g. speed, change of speed or motion direction) over space and time, and also relates one object's motion to the motion of all others. The observation data (ID, location and time) is transformed into a matrix which features a time axis, an object axis and motion parameters. The matrix is matched with formalised patterns to reveal basic searchable relative movement patterns (Figure 3). REMO patterns can be derived by finding interrelations among objects that are clustered on the time-axis, across the objects or combination of both (Figure 3d).

The above-described characteristics make REMO an appropriate method to discover patterns among freely moving objects, where the algorithms that search for similarity in routes often do not converge. Also, in the REMO analysis, trajectory patterns can not only be identified, but also quantified based on user-defined parameters. Therefore, various external factors, e.g. from grid data sources, can be incorporated in the analysis of the behaviour of freely moving objects.

Both for pre-processing and analysis of trajectory patterns, we propose a PostgreSQL/PostGIS DBMS. PostGIS supports spatial data types (points, lines, polygons), coordinate systems and transformations; it also supports spatial operations: distance, buffer and topology that is needed for the analysis of trajectories. PostGIS can also export to various formats (text, KML, GML), so that results can be visualised easily.

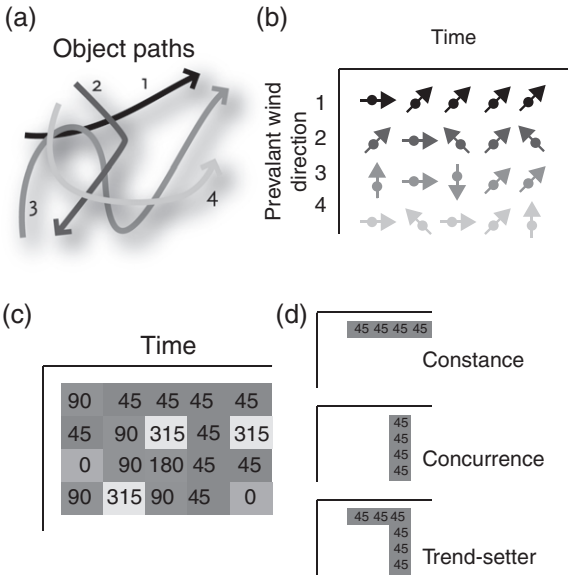

Figure 3. Steps in finding pattern based on relative motion analysis (Laube *et al*. 2005). Reprinted with kind permission from Springer Science + Business Media: Developments in spatial data handling, Finding remo – detecting relative motion, 2005, P. Laube, M. van Kreveld and S. Imfeld.

2.3. *Visualisation layer*

In a service-oriented, distributed architecture such as we propose, the visualisation layer should be loosely coupled to the other layers of the system. There are many options to implement a distributed visualisation layer. Here we will describe two examples of web visualisation layers: first, a browser client using AJAX and TimeMap JavaScript and second, using standardised OGC Open Web Services interfaces and Scalable Vector Graphics (SVG).

2.3.1. *AJAX*

The proliferation of the Web2.0 leads to highly interactive browser-based web applications. In particular, the extensive use of Asynchronous JavaScript and XML (AJAX) influenced this development. AJAX is a technology which is based on several standards provided by the World Wide Web Consortium (W3C); it allows updating of the contents of a web page without having to reload the entire page. An example of software that enables spatio-temporal visualisation in this context is TimeMap JavaScript; it provides options to view and interact with a map and a timeline, explore data interactively, filter data and view details (see Section 3.5).

2.3.2. *Open Web Services*

The standardised OGC Open Web Services interfaces can also be used for visualisation purposes. We have developed a prototype called TimeMapper, a Web

Map Service implementation that serialises spatio-temporal data from a database backend into Scalable Vector Graphics. The SVG is used in a web browser to show animated maps with a built-in advanced user-interface. This interface allows the user to interact with both the spatial and the temporal dimensions of the data. The potential and limitations of the TimeMapper WMS were explored in a test-case for Antarctic iceberg data that can be tried out at: http://geoserver.itc.nl/TimeMapper/ (Becker *et al*. 2009).

3. Case study: visualising iceberg movement

In this section an implementation of a distributed visualisation system is described. As a case study, we used Antarctic iceberg trajectory data. We explain how we pre-processed input iceberg trajectories and interpolated a subset to daily intervals. Then we show the integration of the interpolated trajectory data with metocean (wind) parameters. We describe how we implemented the REMO algorithm (Laube and Imfeld 2002) that allows to look for iceberg trajectories with a certain wind direction profile. Finally, we describe the views and the functionality implemented in a browser client that is running on TimeMap JavaScript.

3.1. *Case study data*

The icebergs data are tracked based on visual comparison of various images. The US National Ice Center (NIC) provides records of icebergs movement since 1978. This dataset consists of 301 icebergs with 15,737 records (at the time of writing of this article) that can be freely downloaded from NIC's website (National Ice Center 2009). The dataset contains some basic information such as iceberg name, iceberg position (in latitude and longitude) time of recording (year and day), iceberg size (in nautical miles) and name of the satellite that is used to record a particular iceberg. This dataset has an irregular temporal resolution: for two consecutive records of the position of an iceberg, the difference may be 1–5 days, but there are also cases of 30 or even 50 days.

The iceberg dataset of the NIC has been used by researchers with various objectives. Schmittner *et al*. (2002) describes iceberg information as invaluable for climate related investigations. Climatologists use trends in iceberg numbers and events – such as calvings or splits, appearances and disappearances – as climate change indicators. They noticed that calving might result from a warming effect that influences the strength of ice shelves. Iceberg events are strongly influenced by external factors, such as sea surface temperature (SST), winds and ocean currents (Benn *et al*. 2007). Many scientists try to investigate the events that occur on icebergs by defining a model of iceberg movement with respect to these external factors and the life history of icebergs, taking into account factors like calving sites and approximate calving rates (Bigg *et al*. 1997).

To conduct studies like the ones mentioned above, iceberg datasets need to be integrated with external factors, i.e. metocean data. There are many possible factors that influence iceberg behaviours, but in this article, we limit those to wind direction and SST. But before performing any integration, iceberg trajectories need to be made consistent.

3.2. *DBMS support for pre-processing of iceberg trajectories*

Iceberg trajectories are based on results of visual tracking and manual updating of iceberg positions. Since both activities are error-prone, the data set contains various errors. For instance, quadrant A (one out of four areas distinguished for data collection purposes and naming of icebergs) has 5061 tuples in a spreadsheet, referring to 95 icebergs. In this dataset, the following typical errors occur:

- Data duplicates with the same value for iceberg name, date of recording, latitude, longitude and size (219 tuples).
- Data duplicates with the same value for iceberg name and date of recording, but different latitude and longitude values (18 tuples).
- Data with null values for position and time recordings (1 tuple).
- Data with 0 (zero) values for longitude (4 tuples) and data with 0 (zero) values for latitude (3 tuples).
- Incidental exchange of '+' and '−' signs for longitude (15 tuples); incidental exchange of '+' and '−' signs for latitude (7 tuples).

These errors need to be treated before any analytical operations are performed. Below we show several data pre-processing steps related to iceberg trajectories.

3.2.1. *Outliers management*

Our initial observation of iceberg records shows that various outliers exist as a result of:

(i) Typing errors which caused the incidental exchange of plus and minus signs.
(ii) Switch of values between longitude and latitude.
(iii) Typing error which caused additional digits in longitudes.
(iv) None of the above, assumed to be true outliers, like icebergs that move unrealistically far in a short period of time.

All the above types of outliers could be handled. True outliers (icebergs that move more than 80 km/day (Robe and Maier 1980)) have been removed. Speed was calculated based on distance travelled between two consecutive records of the same iceberg. A function to handle the typing errors compares the consistency of current longitude (or latitude) value with its previous record (Algorithm 1).

Algorithm 1: Manage outliers
Ensure: Table to store the clean data set from outliers (*outlier free*) has been created
Require: The first record has to hold the correct value
1: Retrieve two consecutive records (r_1 and r_2) of the same iceberg and calculate speed between r_1 and r_2
2: **if** speed > 80 **then**
3: It is an outlier, perform check:
4: Create temporary record r_{temp}
5: $dif \leftarrow r_1.longitude - r_2.longitude$
6: **if** $dif >= 10$ **then**
7: Possible typo, i.e., should be 8 instead of 80:
8: $r_{temp}.longitude \leftarrow r_2.longitude/10$

9: **else if** ($r_1.longitude > 0$ AND $r_2.longitude < 0$) OR ($r_1.longitude < 0$ AND $r_2.longitude > 0$) **then**
10: Possible typo, i.e., change of sign:
11: $r_{temp}.longitude \leftarrow r_2.longitude*(-1)$
12: **end if**
13: Write back newly corrected record from temporary record:
14: $r_2.longitude \leftarrow r_{temp}.longitude$
15: Check newly corrected record: Calculate speed between r_1 and r_2
16: **if** speed < 80 **then**
17: Insert r_2 into *outlier free*
18: **end if**
19: **else**
20: Insert r_2 into *outlier free*
21: **end if**

3.2.2. *Data duplicate management*

Data duplicates are defined based on four attributes: iceberg name, longitude, latitude and date. The iceberg data set shows there are two types of duplicates, i.e. full and partial. A record is defined as full duplicate if another record exists with similar values for all four attributes. If the similarity is found only on iceberg name and date, it is defined as a partial duplicate. Hence there are two steps needed for handling data duplicates:

- The first step is dealing with full duplicates. This is done by taking only one milestone for each group of milestones that have full duplication.
- The second step is applied for partial duplicates. A similar approach as in the previous step is followed, but only an average longitude and latitude value of all duplicates is taken. This step is able to treat partial duplication; but it is prone to the presence of outliers in the group. This limitation can be avoided by handling the outliers before the duplicates.

3.2.3. *Interpolation*

Iceberg trajectories have an irregular temporal resolution: for two consecutive icebergs records the difference is usually 1–5 days, but there are also cases with gaps of 30–50 days. For the analysis of trajectories, we need to interpolate or aggregate the trajectories to certain spatio-temporal intervals. There are many interpolation methods available ranging from linear interpolation to explicit application dependent models of object movements. In this article we show an example of interpolation by applying a simple linear interpolation. Linear interpolation is defined using the following equations (Nanni *et al*. 2005):

$$(x - x_1)(t_2 - t_1) - (x_2 - x_1)(t - t_1) = 0 \quad (1)$$

$$(y - y_1)(t_2 - t_1) - (y_2 - y_1)(t - t_1) = 0 \quad (2)$$

In these equations, x and y refer to a record that needs interpolation. It is done by considering x_1 as a coordinate of the previous record and x_2 refers to the coordinate

value of the next record of an x. Note that values in x and/or y are in projected coordinates (Antarctic Polar Stereographic Coverage, SRID =3031) of the respective longitude and latitude of an iceberg record.

3.3. *Integration of iceberg trajectories and metocean variables*

Once we know positions of all icebergs, we can start the integration process using IDV and additional Jython scripting functionality of IDV. As mentioned earlier, the idea behind the integration of grid and trajectory data is to find the value of a grid cell (or cells) at the same location and time as stored in the trajectory records. Therefore, location (latitude, longitude) and time of the moving objects need to be read out first.

Then the corresponding grid values have to be extracted, but the temporal resolution that can be obtained for metocean data may differ, so this influences the process. In the prototype, we therefore gave users three options to select temporal accuracy: at intervals based on seconds, hours and days. If a temporal accuracy in hours is selected, trajectory data will be integrated with grid data of the same hour and day (if available). If for the grid data there is more than one value for the same hour recorded, the first value will be used.

A grid often has missing cell values, so interpolation methods are necessary to estimate values of cells that lie between the known cell values. Two methods for grid data interpolation are included in the data integration layer: an inverse distance weighting function to estimate values on the same date (spatial interpolation) and a function to interpolate between grid values of a cell before and after the time of a missing value (temporal interpolation). All the above-mentioned functions have been realised in the user interface presented in Figure 4.

As the icebergs are large, they may overlap with a number of grid cells having different values. Therefore, using only the values of the cell that coincides with the centre of an iceberg may not give a good indication of influencing factors on moving objects. Therefore, minimum, maximum and average values of grid cells corresponding to the area of an iceberg are also acquired.

As an example, we used a subset of 289 iceberg trajectory records for quadrant A corresponding to 99 distinct days in 1995. After performing the linear interpolation to daily intervals, number of records grew from 289 to 1461. As a grid data sources, we used three variables: the U (east-west) and V (north-south) components of wind direction and SST for the year 1995. Grid data sources are hosted by the Oceans and Climate Digital Library Portal of the Tasmanian Partnership for Advanced Computing (http://ngportal.sf.utas.edu.au/gridsphere/gridsphere) as follows:

- Grid data source link to U component of wind in 1995: dods://ngportal.sf.utas.edu.au:1/thredds/dodsC/library/ncep1/surface/uwnd.sig995/uwnd.sig995.1995.nc
- Grid data source link to V component of wind in 1995: dods://ngportal.sf.utas.edu.au:1/thredds/dodsC/library/ncep1/surface/vwnd.sig995/vwnd.sig995.1995.nc

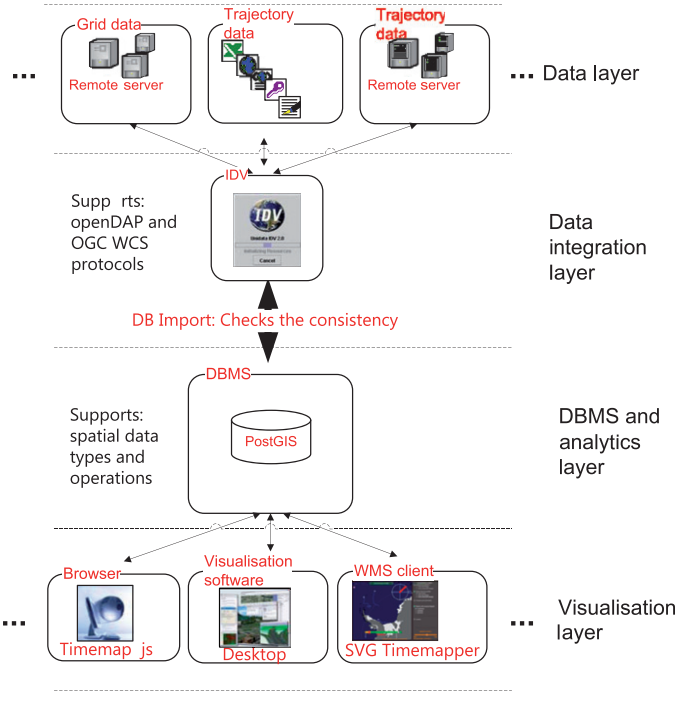

Figure 4. User interface of the data integration service.

- Grid data source link to sea surface temperature in 1995: dods://ngportal.sf.utas.edu.au:1/thredds/dodsC/library/auscom/forcing/sst_restore_1995.nc

The integration layer was build on a personal computer (Intel Core 2 Duo CPU 2.26 GHz) with 2024 megabytes allocated for Java memory to run the IDV client. The processing time was 3437 s (about 57 min) in total with a time consumption of about 1.2 s to process each record (out of which 0.75 was to actually download each integrated subset). This is a slow performance and certainly there are ways to optimise the integration component of the system (see Section 4 for a more elaborate discussion).

3.4. *DBMS support for visual analysis of iceberg trajectories*

The option to search for patterns is based on the REMO concept, that employs a classification technique to find similarity in behaviour of iceberg movement. The search for patterns is based on two key features (Laube and Imfeld 2002, Laube *et al.* 2005):

(i) Transformation of the trajectory data into a matrix.
(ii) Pattern detection.

The matrix is formed by iceberg names (object ID's) in the rows and regular time steps in the columns, and the matrix cells are filled with classified motion parameter

values. As an example, Figure 3(d) shows a matrix that contains classified movement direction of some icebergs from $time_1$ to $time_5$.

There are three types of patterns that can be extracted from the matrix using search templates:

(i) Constancy, results from a search template that spans over time, i.e. for one object over several time steps (Figure 3(d), upper).
(ii) Concurrence, results from a search template that moves across objects, i.e. spanning several objects at one time (Figure 3(d), middle).
(iii) Conformity, results from a search template that combines the search over times and across objects, i.e. several objects at several times (Figure 3(d), lower).

For this article we used only wind speed and direction (calculated from the integrated U and V components of wind in the previous step) as parameters to detect similarity in iceberg movement. Before integration with the grid data, the iceberg data were interpolated to form records having daily intervals. After integration and calculation of speed and direction, these derived attributes were classified. Two types of data classification have been applied. Wind direction classes were based on eight cardinal directions (as commonly used). Wind speed classes were based on a quantile classification to get an equal distribution of values in the predefined number of classes.

In general, the most interesting (spatio-temporal) pattern that can be detected with the REMO algorithm is conformity, since its output will be objects with the same behaviour over a number of user-defined, consecutive time steps. It helps in finding causes and relationships, particularly in integrated data: answers to question like, for example: 'How much does wind direction and/or wind speed influence iceberg movements?'. To test the procedure, a relative motion matrix was created based on a user-defined range of consecutive time steps for wind direction (as example). The REMO algorithm running on the server was quick (less than 5 s for 1461 records) and provided the ID's of icebergs having the same wind directions with an array of five or more consecutive time steps as output.

3.5. *Visualisation of iceberg trajectories*

To visualise the integrated iceberg data from the DBMS in a clients browser, we chose PHP as middle ware to read the data in the database, restored them in KML files, and then send the files to the client side through an Apache server. Both PHP and Apache are open source software. KML can be used in many applications (e.g. Google Maps) and the files are dynamically generated, i.e. if changes are made in the database, new KML files are created and transferred to the client.

At the client side, the intention was to create a prototype that enables users to obtain overview, zoom , filter and see details on demand: the first task is mentioned in the Visual Information Seeking Mantra of Shneiderman (Shneiderman 1996). We chose TimeMap JavaScript (http://code.google.com/p/timemap/) for the graphical user interface. It is an open source JavaScript library that enables integration of the Google Maps API (http://code.google.com/apis/maps/) and

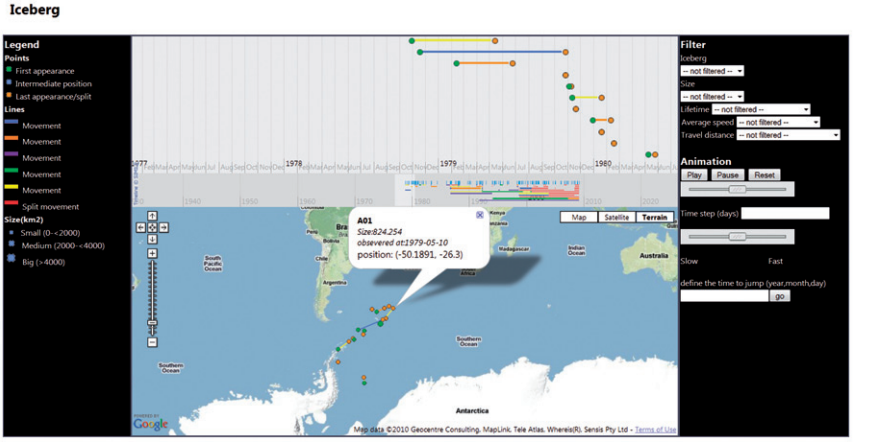

Figure 5. Client side interface for visualisation of iceberg trajectories on overview level. Visible temporal range is from October 1979 to October 1980.

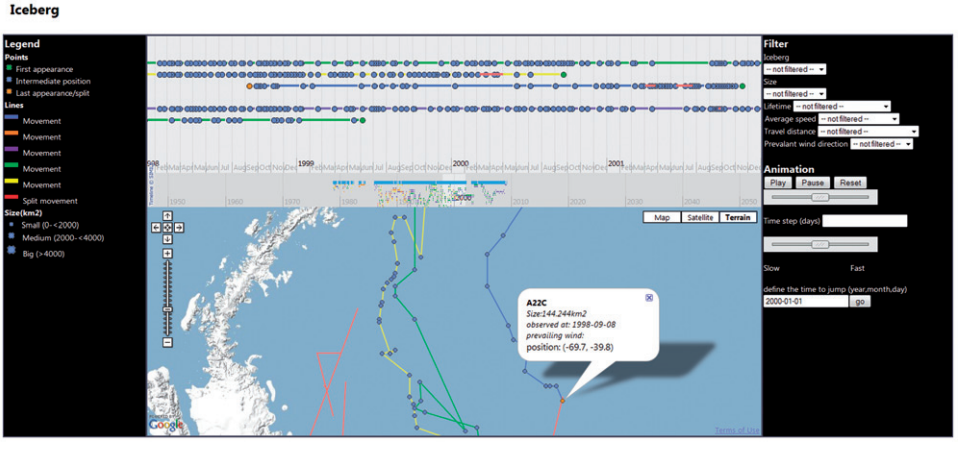

Figure 6. Client side interface for visualisation of iceberg trajectories on detailed level. Visible temporal range is from February 1998 to December 2001.

Timeline (http://code.google.com/p/simile-widgets/) to provide spatial and temporal views on the data, and processes KML files for visualisation (Figures 5 and 6).

Google Maps offers not only a map, image or hybrid layer as base for the KML overlays, but it also has some limitations. The projection cannot be changed (particularly a disadvantage for phenomena that occur around Antarctica: the continent is spread all over the bottom of the spatial view). Also, there is no timeline to display spatio-temporal data, and no animation to explore spatio-temporal patterns. The last limitations could be overcome (see below). Timeline, originally developed by the SIMILE project, is a tool to map time-stamped data. It helps users to interact with, and browse through temporal data. Different timelines can be generated, including views with different bands. The bands are synchronised so that

panning in one band scrolls the other. A disadvantage is that only items that are in the visible range of the Timeline are shown in the spatial view.

The visualisation of the Antarctic iceberg case is focused on the display of important iceberg events, like first appearances, iceberg splitting (or calvings) and last appearance (or disappearances) and on trajectories, in space and time. To provide overview in the spatial view, it was decided to emphasise at low spatial zoom levels (<5) the first and last appearances of each iceberg by points, and to link these two points by a straight line in the spatial view (Figure 5). After all, these events are related to the same iceberg, and a line gives an indication of the main direction of movement. Details (intermediate iceberg positions and portions of the trajectories) are visible from spatial zoom levels of 5 and higher (Figure 6). This has been realised by using different KLM files for overview and details. Control of their display via zoom levels is not supported by Timemap JavaScript, but it could be realised by using the GEventListener function provided by the Google Maps API.

In the Timeline, overview and details were realised in our prototype by configuring two bands: the bottom one displays decades (to provide overview), while the top one provides details by a division in months. Events were originally positioned randomly on the Timeline, hence JavaScript code was added to improve the display. All the events happening to the same iceberg were connected by one line. Each line indicates the lifetime of an iceberg on the Timeline.

In the spatial view, first, last and intermediate positions of icebergs can be distinguished because they are represented in different colours. The sizes of icebergs (classified) are also represented at each of the displayed positions. Trajectories can be distinguished on colour: a random selection out of five colours is made, except if a split occurs (e.g. where iceberg A20 splits into A20A and A20B), then that part of the trajectory is represented in red. Similar colours appear for events and trajectories in the Timeline.

The prototype enables zooming and panning, and brushing in the linked views. Timemap JavaScript also supports filtering functions that have been used to create several options:

- iceberg name (ID)
- iceberg size (small, medium or big)
- lifetime (short, medium or long)
- average speed (slow, normal or fast)
- travel distance (short, medium or long)
- prevalent wind (eight main directions).

All filtering makes use of pre-defined classes in the KML files. The option to filter on prevalent wind only appears at detailed level, since it does not make sense at the overview level. By clicking on an object in the Timeline or in Google Maps, details on demand can be obtained from a pop-up window that appears.

With Timeline, an animated display in the spatial view can be simulated by manually panning the Timeline, but functions like play, pause and control of the display speed are not provided. We have implemented animation controls in the prototype using JavaScript Timing events. Two sliders are provided: one to control the display speed (borrowed from the JavaScript slider created by Erik Arvidsson (http://webfx.eae.net/dhtml/slider/slider.html/) and another one to control the time interval (in World Time) of scenes to be displayed in the animation. Additionally,

the function SetCenterVisibleDate (provided by TimeMap) was implemented to move the Timeline, since normally only objects that are visible in the Timeline are displayed in the spatial view. Finally, we implemented a function to jump to a user-defined time, which can be useful in long-time series. The prototype can be explored at http://geoserver.itc.nl:8181/icebergs_timemap/.

4. Discussion and future research

The main focus of this article was to develop a web service for remote sensing and trajectory data integration and to show how the integration mechanism would fit in with other components of a geovisual analytics system: DBMS, Analytics and Visualisation layers resulting in a distributed architecture. Such architecture allows substitution of one of the components for a more advanced alternative if necessary.

Technically, the weakest component of the proposed system is the Analytics layer. In the prototype, the Analytics layer is realised using pl/pgsql scripting language of PostgreSQL DBMS. Implementing it on the DBMS means, there is little support for analytical or geostatistical functionality in the proposed system (e.g. for clustering, classification methods). Also, in the prototype, the results of the application of the REMO algorithm are not directly visualised yet, since that needs further development. Instead, filtering based on integrated wind direction, essentially giving the same result, has been included (Figure 7). Note, however, that only a subset of iceberg trajectory records for quadrant A in 1995 was used, so filtering gives a relatively small output. Wind speed and SST (although integrated for the subset) are also not yet included, but the methods to do so would be the same.

The current implementation of the visualisation component has some other limitations. The changing displays (use of another KML file) upon zooming is rather confusing; a flexible transition would be needed to improve this. Furthermore, if there are many events, they may overlap. Also, filtering on more than one parameter is currently not possible, and progressive filtering is not enabled.

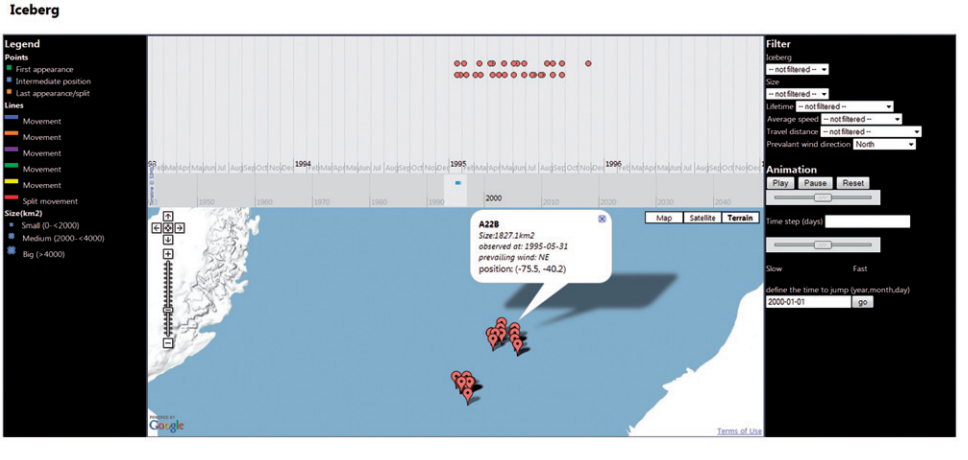

Figure 7. Filtering iceberg trajectories for the year 1995 based on prevailing northern wind direction.

Finally, there are some technical problems in our modifications of the TimeMap JavaScript, some bugs exist (e.g. IE constantly issues an error, sometimes events disappear from the SIMILE Timeline) and display is relatively slow, particularly when many items have to be displayed (at high zoom levels, with little or no filtering). Optimisation of the prototype is possible by making the code – and therefore the time needed to display the data – more efficient.

A first experiment has been carried out with TimeMap, but no proper usability testing has been done since the system would first need further improvements, and our aim in this article was merely to demonstrate possible implementations of web services that handle trajectory and grid data integration and include analytical functionalities.

Future research will include evaluation of other RS and trajectory data integration mechanisms, e.g. using OGC Web Coverage Services. We will also continue on improving the performance of the system, since it is currently inadequate for a large trajectory dataset (Section 3.3) by, for instance, grouping iceberg records with the same dates in one server transaction. On the analytical part, the main focus will be on implementing more mechanisms to discover patterns in the integrated data set and on extending the Analytics layer to contain full analytical functionality (e.g. coupling PostgreSQL/PostGIS with R software using PL/R script or with Python using PL/Python script). Another main aspect for future research would be directed towards a better and extended visualisation component, e.g. by adding relevant attribute views, implementing progressive filtering and flexible transitions between visualisations. Ultimately, the aim is to propose methods and tools that are applicable to a broad range of real-world phenomena.

Acknowledgements

The authors wish to thank Timothée Becker for his work on the TimeMapper WMS.

References

Andrienko, G., *et al*., 2007. Geovisual analytics for spatial decision support: setting the research agenda. *International Journal of Geographical Information Science*, 21 (8), 839–857.

Andrienko, G., *et al*., 2008. Geovisualization of dynamics, movement and change: key issues and developing approaches in visualisation research. *Information Visualization*, 7 (3/4), 173–180.

Becker, T., Köbben, B., and Blok, C., 2009. TimeMapper: visualizing moving object data using WMS time dimension and SVG SMIL animations. *In: Proceedings of SVG Open 2009 – SVG coming of age, 2–4 October 2009, Mountain View, CA, USA*. Available from: http://svgopen.org/2009/[Accessed 27 October 2010].

Benn, D.I., Warren, C.R., and Mottram, R.H., 2007. Calving processes and the dynamics of calving glaciers. *Earth-Science Reviews*, 82 (3–4), 143–179.

Bigg, G.R., *et al*., 1997. Modelling the dynamics and thermodynamics of icebergs. *Cold Regions Science and Technology*, 26 (2), 113–135.

Giannotti, F. and Pedreschi, D., 2008. Mobility, data mining and privacy: a vision of convergence. *In*: F. Giannotti and D. Pedreschi, eds. *Mobility, data mining and privacy,* Geographic Knowledge Discovery. Berlin: Springer, 1–11.

Keim, D.A., *et al*., 2006. Challenges in visual data analysis. *In: 10th international conference on information visualisation*. Los Alamitos, USA: IEEE Computer Society Press, 9–16.

Kramis, M., *et al*., 2009. An XML-based infrastructure to enhance collaborative geographic visual analytics. *Cartography and Geographic Information Science*, 36 (6), 281–293.

Laube, P. and Imfeld, S., 2002. Analyzing relative motion within groups of trackable moving point objects. *In*: M.J. Egenhofer and D.M. Mark, eds. *Lecture notes in computer science*. Berlin, Heidelberg: Springer, 132–144.

Laube, P., van Kreveld, M., and Imfeld, S., 2005. Finding remo – detecting relative motion. *In*: P.F. Fisher, ed. *Developments in spatial data handling*. Berlin, Heidelberg: Springer, 201–215.

Nanni, M., *et al*., 2005. Deductive and inductive reasoning on spatio-temporal data, *In:* D. Seipel, *et al.,* eds. *Applications of declarative programming and knowledge management*, Vol. 3392/2005 of Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 98–115.

National Ice Center, 2009. 'Antarctic icebergs'. Available from: http://www.natice.noaa.gov/products/iceberg/ [Accessed 27 October 2010].

Pelekis, *et al*., 2007. Similarity search in trajectory databases, *In: Proceedings of the 14th international symposium on temporal representation and reasoning*, IEEE Computer Society, 129–140.

Rinzivillo, S., *et al*., 2008. Visually driven analysis of movement data by progressive clustering. *Information Visualization*, 7, 225–239.

Robe, R.Q. and Maier, D.C., 1980. Long-term drift of icebergs in Baffin Bay and the Labrador Sea. *Cold Regions Science and Technology*, 1, 183–193.

Schmittner, A., Yoshimori, M., and Weaver, A.J., 2002. Instability of glacial climate in a model of the oceanatmosphere–cryosphere system. *Science Express*, 295 (5559), 1489–1493.

Shneiderman, B., 1996. The eyes have it: A task by data type taxonomy for information visualiations. *In: Proceedings of the IEEE symposium on visual languages*. Washington, DC: IEEE Computer Society Press, 336–343.

Tiakas, E., *et al*., 2009. Searching for similar trajectories in spatial networks. *Journal of Systems and Software*, 82 (5), 772–788.