IMPLEMENTATION OF PROGRESSIVE TRANSMISSION ALGORITHMS FOR VECTOR MAP DATA IN WEB-BASED VISUALIZATION

B.S. Yang*, R. S. Purves and R. Weibel

GIS Division, Department of Geography, University of Zurich, Winterthurerstr.190, CH-8057, Zurich, Switzerland
* bisheng@geo.unizh.ch

Commission IV, WG IV/1

KEY WORDS: algorithms, visualization, hierarchy, web based, reconstruction

ABSTRACT:

This paper investigates the benefits of progressive transmission of vector data, through designing and developing a hierarchical data model to compress vector data for progressive transmission over the internet. The aim of the hierarchical data model is to extract a coarse vector data set using vertex removal operations. Furthermore, a set of rules are proposed to control the validity of topology for the 'coarse' vector data version, and a recovery algorithm is developed to reconstruct the 'finer' vector data on the client side. The proposed method reconstructs the original data. The overall procedure is to (a) extract coarser data version on the server side on-line; (b) transmit coarser data version to the client; and (c) progressively recover more detailed data on the client side. An experimental prototype system has been developed to test the performance of the proposed method and illustrate its strengths and weaknesses. The experimental results show that the method can efficiently simplify spatial data, can maintain the shapes characteristics of objects during transmission, and improves transmission time over internet greatly.

1. INTRODUCTION

Initial developments in the delivery of spatial data over the internet focused on raster data, for example through the use of Web Map Servers (OpenGIS, 2002). Raster data volumes could be reduced through the use of progressive transmission (e.g., Rauschenbach and Schumann, 1999) and data compression techniques (e.g., Kern and Carswell, 1994). However, the desire to have access to more functionality, specifically the direct querying of objects, together with the development of SVG (Scalable Vector Graphics; SVG, 2002) as a successful means of viewing and exploring vector representations have refocused attention on the delivery of vector data over the internet. Other than the delivery of TINs (De Floriani and Puppo 1995; Park *et al* 2001), much less work has addressed the issues of the delivery of vector-based data over the internet.

In progressive transmission of raster data, detail in an image is gradually filled in enabling the user to get a 'first look' before all of the data have arrived. In the case of vector data, users may also wish to carry out operations on the data – for instance calculating the approximate area of a polygon – and are prevented from doing this until the full data volume has been delivered. Progressive vector transmission attempts to not only give the user a 'first look' at a data set, but to deliver data of sufficient fidelity that manipulation of some coarser version of the data will produce results consistent with the actual data.

Cecconi and Weibel (2000) drew attention to the need for development of techniques for progressive transmission of vector data, and in particular methods for generalising such data at different levels of detail. Buttenfield (2000), Bertolotto and Egenhofer (1999; 2001) and Han *et al* (2003) all proposed frameworks for progressive vector transmission and discussed concepts, challenges, and implementation issues in developing viable solutions from the perspective of cartographic generalization.

The key challenges in progressive vector data transmission were summarized as (Bertolotto and Egenhofer, 2001):

- preservation of topological consistency;
- the complexity of real time generalization; and
- real time compression and encoding of spatial data.

Implementations of techniques for progressive vector transmission include work by Han and Bertolotto (2003) who developed a prototype system using Oracle Spatial™ based on a set of cartographic generalization principles. Buttenfield (2002) investigated single spatial entity progressive transmission over the internet.

In this paper a methodology for delivering large volumes of vector data through progressive transmission is described and implemented. A data model has been developed from which, through the use of a set of simple rules, an initial vector data representation can be transmitted to a client. The coarse representation is generated through vertex removal, which is optimised to minimise conflicts in topology, together with changes in shape. Finer data is reconstructed on the client side through the progressive transmission of the missing vertices. A client-server architecture to perform these operations is described, before a set of experiments using a variety of spatial data are performed and evaluated. These experiments are used to evaluate the potential of the model and discuss potential problems and solutions.

2. CONCEPT OF PROGRESSIVE TRANSMISSION VECTOR MAP DATA

Vector data consists of points, lines and polygons which can be considered to be vertices, open chains of vertices and closed chains of vertices respectively[*]. Thus progressive transmission of vector data in this work is based on the principle of reducing the number of vertices resulting in either a removal of points or simplification of lines or polygons. Simplified spatial entities are therefore transmitted first to the client, and progressively more vertices are gradually delivered from the server to the client until either the user interrupts the process or the entire dataset has been delivered. Figure 1 illustrates the basic concept of progressive vector transmission for a single polygon.

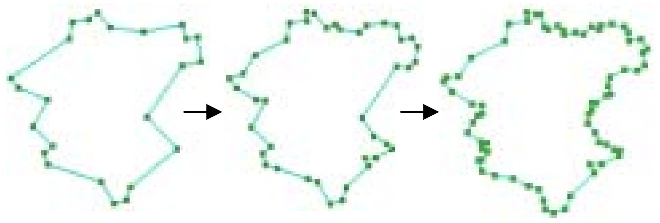


Figure 1 The concept of progressive transmission

The challenge in progressive vector transmission is to reduce the amount of data sent to a client initially, but still allow the user to perform useful tasks. Furthermore, the algorithms used must be sufficiently efficient that they can be used in real-time and therefore not simply move the overhead in delivery of vector data from transmission to calculation of reduced vertex sets. In this paper, we adopt a set of generalization methods which select suitable vertices for removal (Weibel and Dutton, 1999) without any displacement. A dataset can therefore be represented as a full set of vertices and their associated chains and progressively coarser sets of vertices which still maintain key characteristics of the data.

In the next section of the paper a data structure that facilitates vertex removal for progressive vector transmission and a set of rules for carrying it out, whilst maintaining important properties of the data is introduced.

3. HIERARCHY DATA MODEL FOR VECTOR MAP DATA

When displaying a map with a user-selected set of layers, it is impossible to know for what purpose layers have been selected. Therefore, a simple and logical assumption is to consider all layers as having equal weight, and to attempt to preserve topological relationships between and within all layers. To this end we adopt a simple 'spaghetti' data model, where points, lines and polygons are represented by vertices and open and closed chains of vertices respectively. Thus a 'full resolution' map can be represented as:

$$Map_{fullresolution} = \bigcup_{i=0}^{m} point_i \oplus \bigcup_{j=0}^{n} line_j \oplus \bigcup_{k=0}^{o} polygon_k$$

In order to produce a reduced vertex version of the map for use in progressive transmission we must identify a set of rules which allow us to remove vertices whilst:

a) maintaining the shape of objects; and
b) preventing topological inconsistencies with respect to the source data.

[*] In this paper we do not use the terminology of vertices and nodes (c.f. Burrough and McDonnell, 1998)

To meet these constraints vertex removal is carried out in stages, and requires consideration of the following elements:

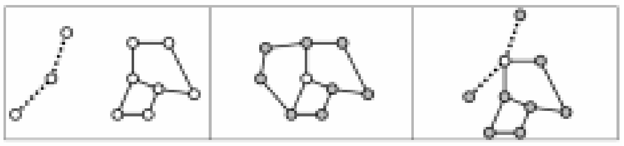
- identification of vertex type;
- operations for vertex removal; and
- ranking of vertices for removal.

Identification of vertex type

Vertices in a map can be considered to have three possible types according to the number of objects which share a vertex and the number of source layers for that vertex:

a) A vertex which belongs only to one or two objects from a single layer.
b) A vertex shared by multiple (more than two) objects from the same layer.
c) A vertex which is shared by more than one object from multiple layers.

These vertex types are illustrated in Figure 2.



(a) (b) (c)

Figure 2 Vertices of types (a), (b) and (c). In each figure the **open vertex** represents the vertex of the appropriate type. The dashed and solid lines are line and polygon data, of different layers, respectively.

Vertices of type (c) are defined as being immovable vertices and cannot be removed from a map.

Operations for vertex removal

Two classes of vertex removal operation are defined – those which can be applied to vertices of type (a) – which can be considered as *simple vertex removal*, and those which are applied to objects of type (b), which can be considered as *complex vertex removal*.

Simple vertex removal

Taking polygon entities as an example, supposing the vertex V_i will be removed from the vertex set of a polygon, the operation will cause one vertex and two segments to disappear, and a new segment to be generated. If the new segment does not intersect with other spatial entities, the operation is *safe*.

Thus, this operator will extract a simplified polygon $\{\overline{V_0V_1},....\overline{V_{i-1}V_{i+1}}.....,\overline{V_nV_0}\}$ from the full polygon $\{\overline{V_0V_1},..,\overline{V_{i-1}V_i},\overline{V_iV_{i+1}},..,\overline{V_nV_0}\}$, and the operation can be represented as

$$P = P_i \oplus V_i, V_i = (x, y, posid, Polygonid) \qquad (1)$$

where *posid* is the order in the vertices set, and *Polygonid* is the ID of the polygon spatial object.

The procedure of the simple vertex removal from line entities is similar to that from the polygon objects. The operation of extracting a simplified line $\{\overline{V_0V_1},.\overline{V_{i-1}V_{i+1}}.,\overline{V_{n-1}V_n}\}$ from a full line $\{\overline{V_0V_1},..,\overline{V_{i-1}V_i},\overline{V_iV_{i+1}},..,\overline{V_{n-1}V_n}\}$ can be represented as

$$C = C_i \oplus V_i, V_i = (x, y, posid, Lineid) \qquad (2)$$

where *posid* is the order in the vertices set, and *Lineid* is the ID of the line spatial object.

Complex vertex removal

If simple vertex removal is used on a polygon vertex of category (b) then there is uncertainty as to how to rebuild the closed chains making up the polygon, and a hole may result (Figure 3). Therefore, a two step operation, first removing the vertex and then ensuring that topological consistency is maintained is carried out.

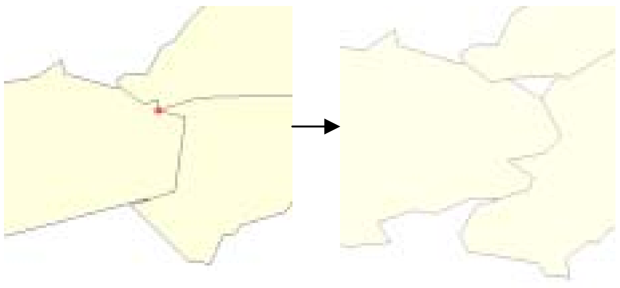


Figure 3 Invalid topology in complex vertex removal

Suppose vertex V_i selected for removal is a common vertex of *n* polygons $\{P_1, P_2, ..., P_n\}$ - therefore *n* vertices $\{V_1, V_2, ..., V_n\}$ are associated with vertex (V_i). Each pair consists of one chain. Thus, to avoid gap generation, the vertex remove operation is defined as follows:

For a set of polygons $\{P_1, P_2, ..., P_n\}$, the operation of common vertex (V_i) remove is defined as

$$P_1 = P'_1 \oplus V_i, , P_{n-1} = P'_{n-1} \oplus V_i, P_n = P'_n - \bigcup_{m=2}^{n-1} V_m \oplus V_i \quad (3)$$

On polygons $\{P_1, P_2, ..., P_{n-1}\}$ the operation carried out is identical to *simple vertex removal*. However, for polygon P_n the following operation, illustrated in Figure 4 is applied:

$$P_n = P'_n - \bigcup_{m=2}^{n-1} V_m \oplus V_i$$

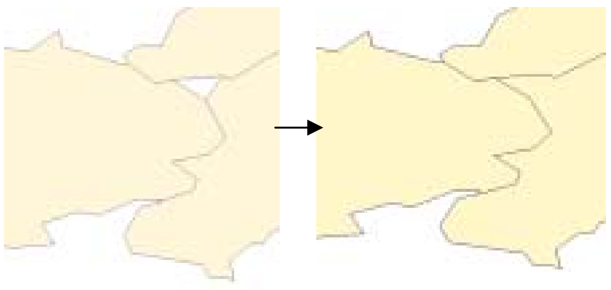


Figure 4 Sealing a hole generated by complex vertex removal

Ranking of vertices for removal

The operations described above provide a means to identify immovable vertices, and operations to remove vertices of different classes. Further criteria are required to rank vertices for removal and to prevent invalid topologies from being generated through self intersection. We adapt the line simplification algorithm of Visvalingam and Whyatt (1993) to carry out this task. This algorithm is based on the principle of simplifying lines by removing the vertices which form the minimum area triangles within the line. By minimising the area removed from polygons or lines the likelihood of maintaining shape fidelity with the original object is increased (Visvalingam and Herbert, 1999). The triangles generated for this operation are also used for a simple topology check – a vertex can only be removed of the triangle formed by it and its neighbouring vertices do not contain any other vertices.

Thus for a polygon with *n* vertices, *n*-1 triangles will be generated, and for a line with *n* vertices, *n*-2 triangles will be generated. Moreover, the start and end vertex of the line are defined as immovable vertices. After identification of immovable vertices, vertices are ranked by triangle area size order for possible removal if, and only if, the triangle formed does not contain any further vertices. The vertex with the minimum triangle area will be removed first. Finally, according to the vertex type a simple or complex vertex removal operation is carried out.

4. CLIENT-SERVER ARCHITECTURE FOR PROGRESSIVE TRANSMISSION VECTOR MAP DATA

In order to implement web-based progressive transmission client and server side components are required. On the server side the reduced vertex entities must be generated and transmitted, whilst on the client side these data must be appropriately displayed and reconstructed as more vertices are delivered.

Figure 5 illustrates the prototype architecture for progressive transmission of vector map data. The whole architecture encompasses three interrelated components: a client side component, an application server component, and a data simplification component. The data simplification component is responsible for extracting coarser data from a database or files in real time and for representing the data in the appropriate data model before the data is sent to the application server component. The server side of the component was integrated into the OpenSource Web Feature Server (WFS) provided by Deegree (Deegree, 2003), allowing the generation of vector maps in response to a standard OpenGIS WFS call (OpenGIS, 2002). The data simplification in the prototype architecture is processed in real time in response to queries from the client, alleviating the need to store a variety of intermediate products and allowing all maps to be generated from a single source data set as required.

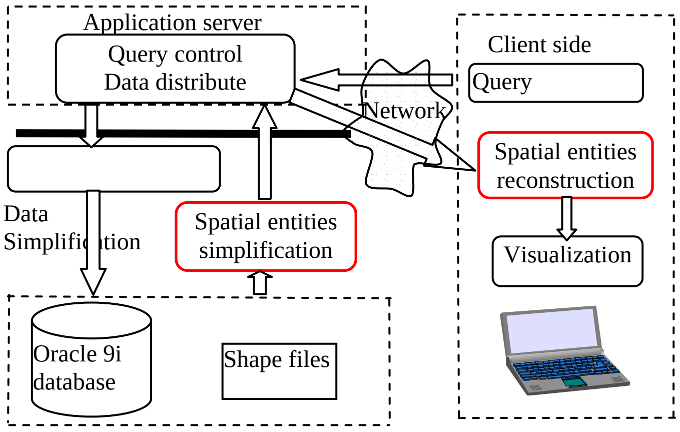


Figure 5 Prototype system architecture

The client side component deals with visualizing the query result, allowing querying of the data and reconstructing the source vector map data.

Figure 5 illustrates the overall architecture of the prototype as implemented, whilst Figure 6 illustrates the process of progressive transmission from the server to the client.

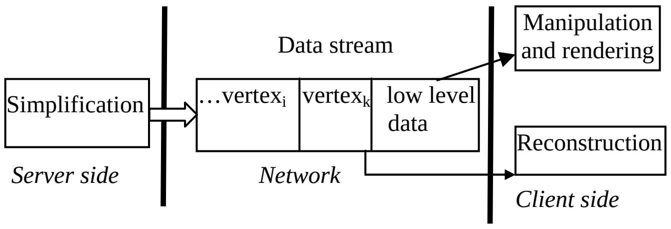
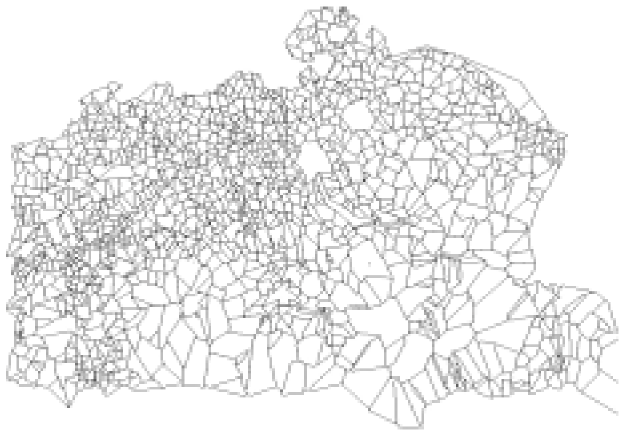


Figure 6. The progressive transmission data from server to client

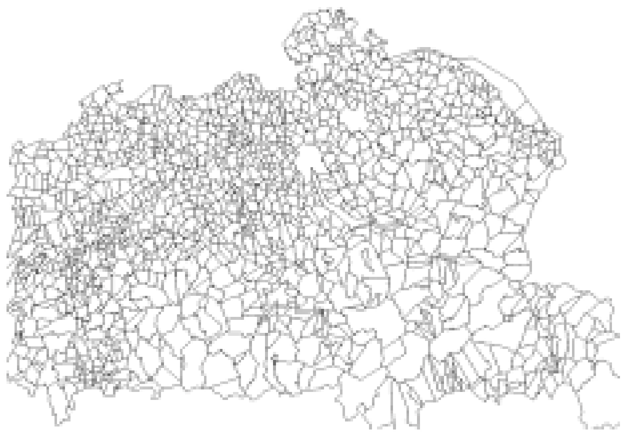
On the client side, data manipulation and rendering and reconstruction threads and querying threads run in parallel once the initial transmission of low level data is complete. Thus, as data is downloaded the user can perform exploratory (e.g., panning and zooming) and querying (e.g., attribute of polygon, area of polygon) operations.

5. EXPERIMENTAL ANALYSIS AND PERFORMANCE TESTING

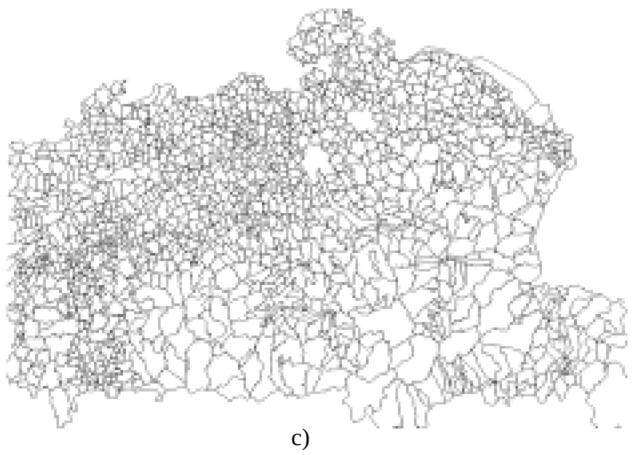
A prototype was developed in Java to test the performance of the progressive transmission algorithm and to check the validity of the algorithm. Figure 7 illustrates three different phases of the progressive transmission visualization procedure on the client side using data illustrating political (municipal) borders in Switzerland (a total of 3062 polygons).



a)



b)



c)

Figure 7 Progressive transmission of vector data

Figure 7 clearly illustrates both the progressive increase in detail at different levels of detail and the fidelity of the intermediate levels of vector data. Figure 8 shows the number of polygons with different areas for the three stages of transmission.

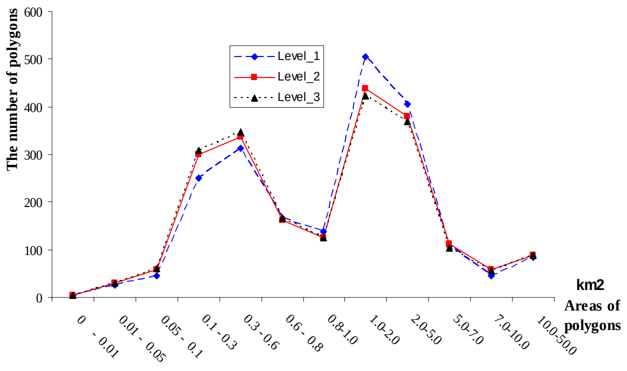


Figure 8 The number of polygons with different areas at three stages of transmission

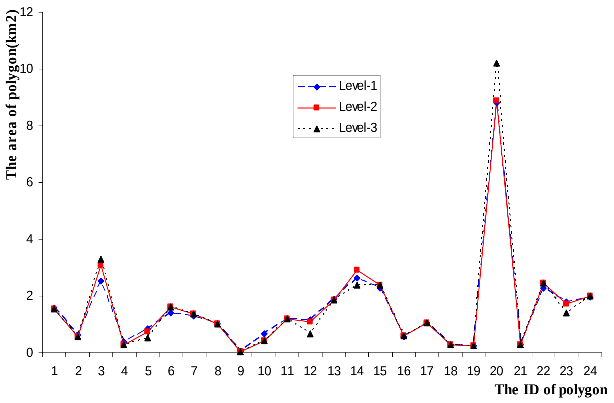


Figure 9 Area of 24 polygons at three stages of transmission

It is clear from Figure 8 that the numbers of the polygons in the three stages of transmission (Level_1, Level_2 and Level_3), which are located in the same area range do not vary greatly. For example, in the three stages of transmission, the numbers of polygons which have an area between 1.0-2.0 km^2 are 506, 438, and 423 respectively. Therefore, the procedure of the transmission is able to maintain the shapes of the polygons.

Figure 9 illustrates the area a sample of 24 polygons at different transmission stages. It is clear that there is not great variation in the areas of these polygons.

Response time is a key criterion to evaluate server performance. The response time is defined here as the time span from the user submitting a query to the client side until a result is visualised by the client. It can be represented as

$$T(response) = T_0(queryfromdatabase) + T_1(transmission) + T_2(rendering)$$

T_0(*queryfromdatabse*) is the time span from submitting SQL statements to getting the result; T_1(*transmission*) is the data transmission time from the server to the client; and T_2 (*rendering*) is the time cost of rendering the results on the client side. *T*(*response*) is the time cost without simplification operations. Suppose the simplification operation is performed on the server side, the response time $T^{'}(response)$ will be represented as

$$T^{'}(response) = T_0(queryfromdatabase) + T_s(simplification) + T_1^{'}(transmission) + T_2^{'}(rendering)$$

T_s (*simplification*) is the running time of the algorithm to simplify the data set to a certain level. The algorithm has been used to simplify several data sets to different levels. Moreover, the running time of the algorithm and the response time are compared to evaluate performance, and the response time with/without simplification operations is also compared. The experiments were undertaken on an intranet with a Tomcat server running under Linux. The client side was a Java applet-based interface.

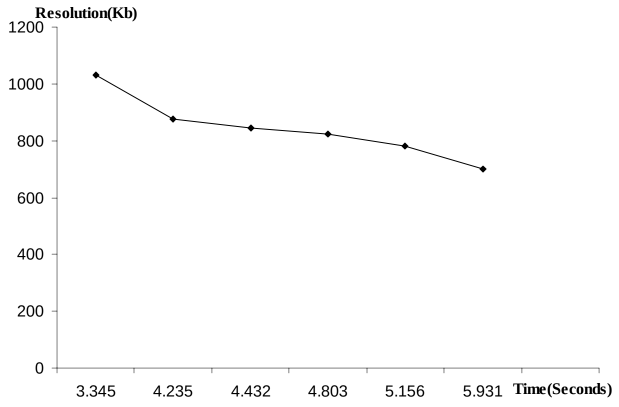


Figure 10. Running time of simplification at different levels

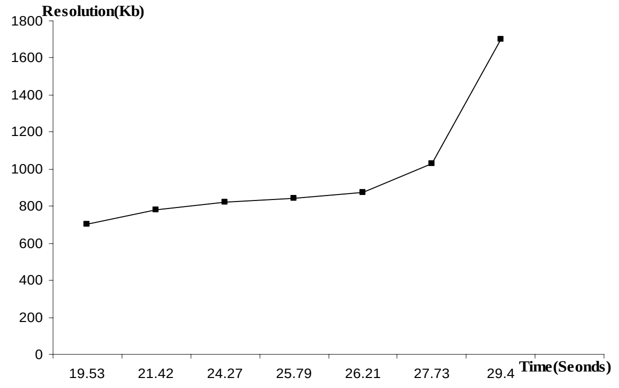


Figure 11. The response time at different simplification levels

Figures 10 and Figure 11 show the running time of the algorithm and the response time of different levels of data. A key advantage of the progressive transmission algorithm is that it is able to reduce the response time of the server greatly. Figure 10 shows that it takes the algorithm about 3.3 seconds to simplify the data set from 1,800Kb to 1,100Kb, and about 6.1 seconds to simplify the data set to 710Kb. It demonstrates the algorithm performs well in simplifying data sets which is important especially when the data set is complex and network bandwidth is narrow. Figure 11 illustrates the response time at different data levels. The response times of level_1 and level_2 are about 19.5 seconds and 21.42 seconds respectively. Compared with the response time of the original, unsimplified data – 29.4 seconds, the algorithm improves the response time. The data volume to be transmitted is reduced after simplification and, as less data volume needs to be transmitted, the response time is improved.

6. CONCLUSIONS

With the wide popularity of web-based applications and services, spatial data delivery over the internet is becoming the ‘bottleneck’ to rapid data transmission. Progressive transmission of vector map data is a promising solution to overcome this ‘bottleneck’. The precondition of progressive transmission extraction of a coarse level data set from the original data set. Extracting multiple-level representations of spatial entities on-line is an efficient solution to implement progressive vector data transmission over the internet.

In this paper a methodology for delivering large volumes of vector data via progressive transmission is presented and implemented. The method represents the original data as coarser level data plus a series of removed vertices through vertex removal. Moreover rules for vertex removal, that maintain the shape fidelity of vector objects and retain correct topology, are proposed in this paper. Secondly, the corresponding simplification algorithm and recovery algorithm, respectively, have been developed in Java, and a client-server system architecture has been designed to test the performance of the progressive transmission algorithm. Thirdly, the experimental results demonstrate that the methodology can simplify original data sets to a low level of detail efficiently and maintain the shapes of geometry objects as well. Because of ability to simplify data the methodology decreases the transmission time considerably. The experimental results demonstrate that the methodology can provide a viable and efficient solution for the progressive transmission of vector map data.

Further work is required to investigate the applicability of the method to large datasets with complex geometries and many layers.

ACKNOWLEDGEMENTS

This research is supported by the EU-IST Project No. IST-2001-35047 (SPIRIT).

REFERENCES

Burrough, P., A. and McDonnell, R., 1998. *Principles of Geographical Information Systems*. Oxford University Press.

Bertolotto, M. and Egenhofer, M.J., 1999. Progressive Vector Transmission. In: *Proceedings ACMGIS'99,* Kansas City, MO, pp.152-157.

Bertolotto, M. and Egenhofer, M.J., 2001. Progressive Transmission of Vector Map Data over the World Wide Web. *Geoinformatica*, 5(4), pp. 345-373.

Buttenfield, B.P., 2002. Transmitting Vector Geospatial Data across the Internet, In: *Proceedings GIScience 2002*, Lecture Notes in Computer Science, Vol. 2748 (Egenhofer and Mark eds), pp.51-64.

Cecconi, A. and Weibel, R., 2000. Map Generalization for On-demand Web Mapping, In: *GIScience 2000*, Savannah, Georgia, pp.302-304.

De Floriani, L and Puppo, E., 1995. Hierarchical Triangulation for Multiresolution Surface Description. *ACM Transactions on Graphics*, 14(4), pp.363-411.

Deegree, 2003. http://www.deegree.net

Han, Q. and Betolotto,M.,2003. A Prototype for Progressive Vector Transmission with an Oracle Spatial Environment. In: *Proceedings of GIS Research UK 11th Annual Conference*, City of University, London, 9-11 April, 2003,pp.189-194.

Han, H., Tao, V., and Wu, H., 2003. Progressive Vector Data Transmission. In: *Proceedings of the 6th AGILE*, Lyon, France, pp.103-113.

Kern, P. and Carswell, J.D., 1994. An Investigation into the Use of JPEG Image Compression for Digital Photogrammetry: Does the Compression of Images Affect Measurement Accuracy. In: *Proceedings EGIS94*, Paris, France.

OpenGIS, 2002. http://www.opengis.org

Park, D., Cho, H. and Kim, Y., 2001. A TIN Compression Method using Delaunay Triangulation. *International Journal of Geographical Information Science*, 15(3), pp.255-270.

Rauschenbach, U. and Schumann,H., 1999. Demand-driven Image Transmission with Levels of Details and Regions of Interest. *Computer and Graphics*, 23(6), pp.857-866.

SVG, 2002. http://www.w3.org/TR/SVG/

Visvalingam, M. and Whyatt, J.D., 1993. Line Generalisation by Repeated Elimination of Points. *Cartographic Journal*, 30(1), pp. 46-51.

Visvalingam, M. and Herbert, S., 1999. A Computer Science Perspective on the Bendsimplification on Algorithm. *Cartography and Geographic Information Science*, 26(4), pp.253-270.

Weibel, R. and Dutton, G., 1999. Generalizing Spatial Data and Dealing with Multiple Representations. In: *Geographical Information Systems: Principles, Techniques, Management and Applications* (Longley, P., Goodchild,M.F., Maguire, D.J., and Rhind, D.W., eds), Second edition, Cambridge, Geoinformation International, pp.125-155.