A Framework for Dynamic Updates of Map Data in Mobile Devices

Hae-Kyong Kang[◇], and Ki-Joune Li[†]

[◇]Department of GIS, [†] Department of Computer Science and Engineering
Pusan National University, Pusan 609-735, South Korea
{hkkang,lik}@pnu.edu

Abstract. Stored map services in mobile devices are being commercialized for mobile and wireless environment such as cellular phones. In order to ensure the quality and accuracy of map, the update on the source map must be reflected automatically to mobile devices. Due to the expensive communication cost and lack of hardware capacity of mobile devices, we should find a compromising solution between the transmission of an entire map and the transmission of only update logs to mobile devices. The transmission of the completely processed whole map results in an expensive communication cost, while simple transfer of update logs requires a large amount of processing to maintain geometric and topological consistency between multiple LODs(Level of Details). In this paper, we first propose a framework of update mechanism in mobile environments where the map data is stored on each mobile device. This framework provides an efficient strategy for processing an update and its propagation to multiple LODs without an expensive cost of communication and large amount of processing at mobile devices. Secondly, several methods are introduced to maintain topological consistency between LODs depending on the type of generalization operators. Finally, we propose an extended SVG(Simple Vector Graphics) to be used as a transfer format of update message to mobile devices.

1 Introduction

With the recent progress of mobile devices, it becomes possible to store a certain volume of map data on a mobile device such as cellular phones or PDAs. For example, several commercial services are being provided for LBS and telematics applications with road map data stored on cellular phones. In order to maintain the accuracy and quality of map stored on mobile devices, the map data must be up to date. If updates take place on the source map databases, they must be transferred and reflected on every mobile devices, where the source map databases are managed by a server. We should carefully consider several points to efficiently handle the update of map data on the server and mobile devices.

The map data distributed into a server and a large number of mobile devices has several important characteristics differing from the conventional distributed databases for several reasons. First, the updates for map services take place only

at the server side, while they can occur at several sites in conventional distributed database systems. This difference makes the update management of map services simple. The second difference comes from the communication cost. For example, the cost of communication between the server and cellular phones is to be paid by subscribers, and becomes a crucial factor. Therefore, we should reduce the size of communication between the map data server and mobile devices. The third difference is the limited hardware capacity and battery of mobile devices. It means that a large amount of processing for updates in a mobile device may be critical.

A tradeoff relationship is found between the communication cost and processing overhead in mobile devices. If we try to reduce the communication cost, it often results in an increase of computing in a mobile device, and vice versa. Suppose that an update on an object occurs on the source map. Then it must be propagated to correspondent objects on levels of details(LOD) derived from the source map and to other objects, which are not even derived from the udpated source objects, to maintain the topological consistency between several LODs. On one hand, a mobile device is however not capable of processing updates and the propagation due to its lack of hardware capacity. On the other hand, it is too expensive to transfer the entire map to mobile devices, after the server has completely processed the updates and propagations.

In this paper, we make a contribution to find a compromising solution between these contradicting requirements. First, we will propose the framework for updating mechanism of a system under development, called MobiMap, which is a system consisting of a map server and mobile devices to provide mobile map services. Second, we propose methods to process the propagations of an update to several LODs for maintaining the consistency between LODs. Third, we propose an extension of SVG(Scaleable Vector Graphics) describing udpate objects so that mobile devices can receive the updated objects with the extended SVG by reducing the amount of communication for transferring the updates to mobile devices.

This paper is organized as follows. In section 2, we will present the motivation of our research and the related work. The framework for processing updates and their propagations in our MobiMap system will be presented in section 3. In the subsequent section, several methods are to be proposed for maintaining topological consistency between LODs according to generalization operators. In section 5, we will propose an extension of SVG, which is to be used as a message format of update reports from the server to mobile devices. Finally, we will conclude this paper in section 6.

2 Motivations and Related Work

The map stored on a mobile device consists of several LODs. Thus, an update of the source map implies not only the modification of an object but also the propagation of updates to the corresponding objects of different LODs. Further-

more, its neighboring objects may need to modify to maintain the topological consistency between LODs.

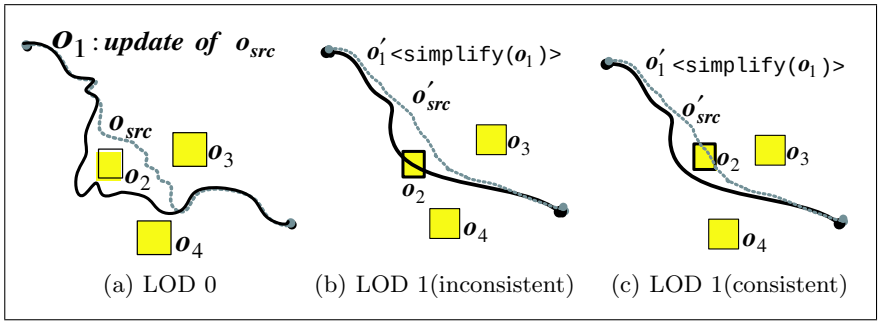


Fig. 1. Topological consistency for line simplification

Figure 1 explains the processing of an update on the source map. In the application area like navigation, the landmarks on the roads are extremely important and the topological relationships between roads and landmarks are crucial to drivers. An updated object o_1 is not intersect with o_2 on the source map (LOD_0) as depicted by figure 1(a), while they are intersect on LOD_1 as figure 1(b), which is a generalized map from LOD_0. Because this topological inconsistency is critical, it must be corrected as shown by figure 1(c) to ensure the topological consistency between roads and landmarks. Consequently, a simplification of an object on the source map results in not only an simplification of an object on LOD_1 but also several modification of its neighboring objects. In order to process an update and its propagation to other LODs, two approaches may be possible as follows,

- **Transfer of map data**: The processing for updates and its propagations to LODs are carried out at the server side and the map data modified is transferred to mobile devices by this approach. However, it results in an expensive communication cost for transferring a large amount of data. The price for transferring 1M bytes data to a cellular phone is approximately US$50, which is not negligible.
- **Transfer of update logs**: According to this approach, we transfer only update log data to mobile devices. It is evident that the communication cost is greatly reduced by this approach. However each mobile device should process the update and its propagations to several LODs, which are a large amount of overhead to a small device like cellular phone. And this processing requires a lot of energy as well.

However, these two approaches are not feasible since each approach focuses on only one of two requirements contradicting, which are the reductions of communication cost versus the processing overhead of mobile devices. Therefore both

of the approaches fail to satisfy the two requirements. In this paper, we focus on a compromise between the two opposite requirements.

The basic issue of this paper is related with how to process an update and its propagation to multiple LODs to ensure geometric and topological consistency. A number of work has been done to maintain the consistency between multi-scale or multiple LOD spatial databases. But few of them deal with the topological consistency except [5, 6, 15]. Several methods have been proposed for maintaining topological consistency depending on the operation type of generalization. In [6], a method to assess the topological integrity has been proposed for aggregation operator. A similar work has been carried out for collapse operator in [13]. J. Sharma has proposed a method to ensure topological integrity for line simplification operator in [5]. These work provide a theoretical background of our study and development of our MobiMAP manager.

3 Update Framework of MobiMap Manager

In this section, we present the framework of update mechanism designed for MobiMAP Manager, which is under development by our research team. The main focus in designing the framework is on how to reduce the communication cost as well as processing overhead of mobile devices. The framework is shown by figure 2, where an update and its propagation to multiple LODs are processed as follows.

- **step 1 : update on $\mathbf{LOD}_0$**
  An update is processed in the source database (LOD_0).

- **step 2 : geometric propagation on $\mathbf{LOD}_n$**
  The server stores all LODs. The corresponding objects in different LODs are updated by using generalization operators such as line simplification, collapse, and aggregation operations[9].

- **step 3 : handling consistency**
  This step consists of the two sub tasks, which are assessment of topological consistency and correction of inconsistency.
  a. assessment of topological consistency: For each LOD derived from LOD_0, neighboring objects are checked to assess topological consistency with LOD_0. The methods for assessment of topological consistency [5, 6, 13] will be explained in the next section.

  b. topological propagation on $\mathbf{LOD}_n$: If any topological inconsistency is found at a LOD, the neighboring objects must be corrected to ensure the consistency with LOD_0. The correction methods are discussed in [10, 14].

- **step 4 : transfer of updates to each mobile device**
  After having found the updates on each LOD, the server transfers the update request to each mobile device in extended SVG, which will be explained in

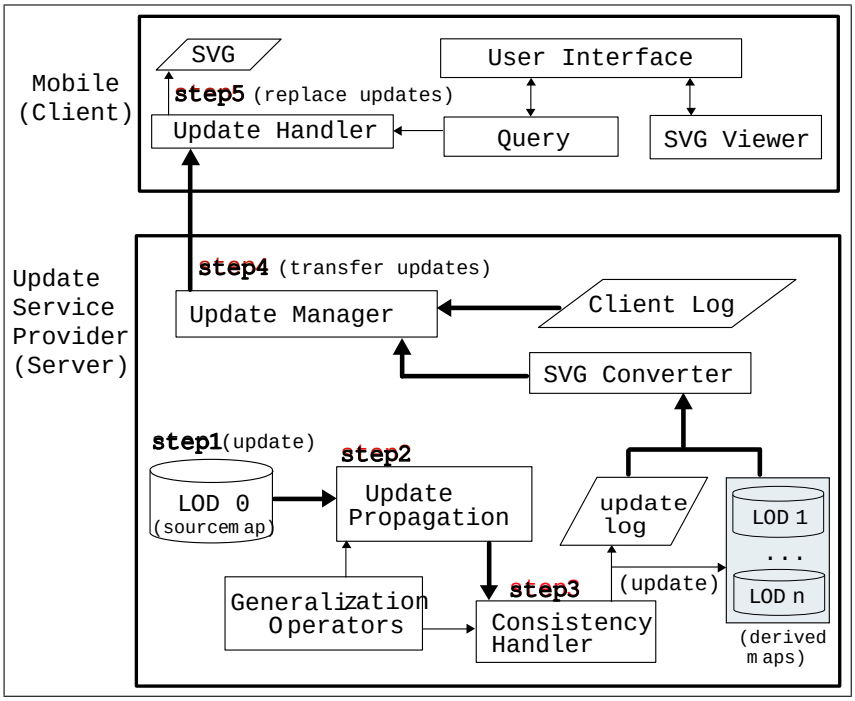


Fig. 2. Framework of update mechanism

section 5.

- **step 5 : updates on mobile devices**
  The updates received from the server with an extended SVG format are to be reflected on a mobile device. The updates in an extended SVG format replace the old objects in original SVG files by *Update handler*.

Because hardward capacity of mobile devices are limited, we should consider the way to reduce the processing overhead on mobile devices. Therefore, our MobiMAP Manager generate updated LOD_n using a mobile device on a server-side, instead of generating it on a mobile device. We will discuss more details about this step 3(assessment of topological consistency) in the next section 4.

When MobiMAP Manager derives an updated LOD_n, it stores an udpate log. By reference the update log, updated objects of LOD_n are converted to an extended SVG format. It reduces the cummunication cost because the extended SVG format includes updated objects, instead of entire objects. In the section 5, the extended SVG schema and step 4 will be explained.

4 Checking Topological Consistency Between LODs

If an update takes place on the source map LOD_0, it must be propagated to other LODs, which are generalized from LOD_0. As we have seen in the previous sections, the propagation includes not only geometric generalizations, but also assessment of topological consistency, which differs depending on the type of geometric generalizations. In this section, we will present the assessment methods for three geometric generalizations - line simplification, collapse, and aggregation. These assessment methods are implemented in the `consistency handler` in figure 2.

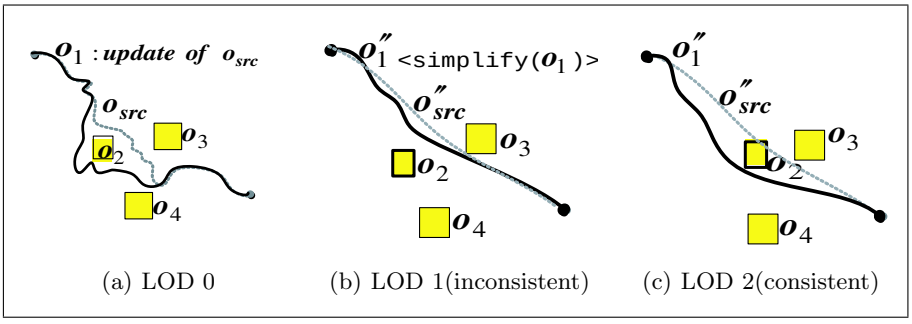
4.1 Topological Consistency for Line Simplification

By line simplification, we can remove vertices of less importance from a line with a large number of vertices. The most popular algorithm of line simplification is proposed by Douglas and Peuker[2]. There is an one-to-one mapping between the original line and the simplified line, the difference between LOD_0 and other LODs is only found in geometric properties. It means that the topology in LOD_0 should be preserved in other LODs. If a different topology is found in two LODs, they are considered as topologically inconsistent.

In figure 1, we already showed an examples of topological consistency for line simplification. In addition, figure 3 shows an another example. In figure 3, o_{src} in LOD_0 is an original line object and o_1 is the updated object from o_{src}. o''_{src} in LOD_2 are the simplified objects from o_{src} in LOD_0. o''_1 in LOD_2 are the simplified objects from o_1 in LOD_0.

We observe that o_2 is at the south-west of o''_1 in LOD_1 as shown in figure 3(b), while o_2 is at the north-east of o_1 in LOD_0 as figure 3(a). Thus, a topological difference is found in LOD_1. In order to maintain the topological consistency, we must correct o'' or move o_2 like figure 3(c) so that the topologies in LOD_0 and LOD_1 be identical.

We observe that o_2 is placed on the left side of o'_1 in LOD_1 as shown in figure 3(b), while o_2 is on the right side of o'_1 in LOD_0. A topological mismatch is found



(a) LOD 0 (b) LOD 1(inconsistent) (c) LOD 2(consistent)

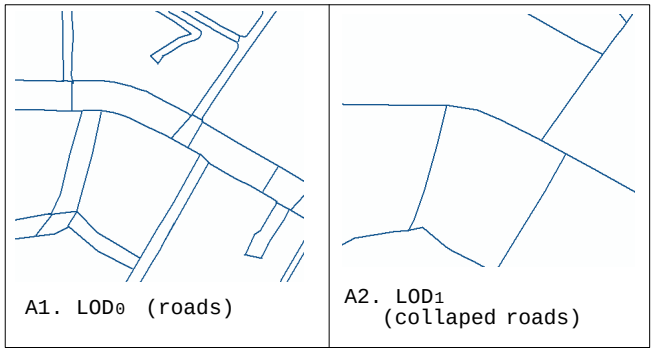
Fig. 3. Directional Consistency for Line-Simplification

and it must be corrected by moving o_2. In order to maintain the topological consistency, we must correct o' or move o_2 like figure 3(c) so that the topologies in LOD_0 and LOD_1 be identical.

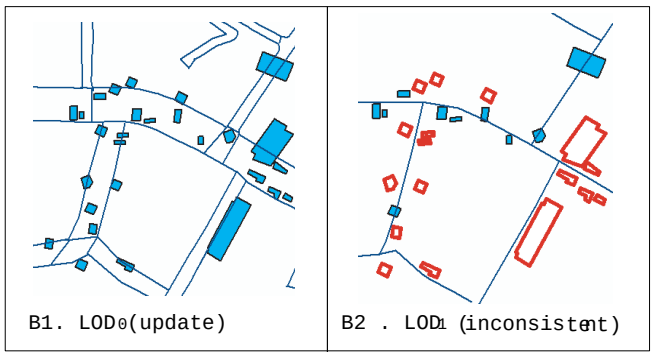
A similar case is found in figure 1. In both cases, we observe that an update of a line object results in at least two modifications. The detail algorithm to assess the topological inconsistency is presented in [5] and the correction methods are proposed by [10, 14]

4.2 Topological Consistency for Collapse

Collapse operator reduces the dimension of a spatial object, for example, from polygon to point. But in some cases, it is impossible to maintain the same topology. For example, if a polygon a containing another polygon b is collapsed to a point a', a' does no longer contain b. Consequently the topological consistency does not imply merely identical topology. For this reason, the assessment of



(a) original and derived road maps



(b) detection of inconsistent relations(white polygons) after update(insert) of new facilities into LOD_0

Fig. 4. Topological Consistency for Collapse

topological consistency for a `collapse` operation is a little more complicated than `line-simplification`.

Figure 4 shows how to assess topological consistency between multiple LODs. Initially, we have map data of roads on LOD_0 and LOD_1, where roads on LOD_0 are collapsed to lines as shown by figure 4(a). Then, several road facilities are inserted into the map of LOD_0. All of facilities intersect with road polygons on LOD_0 as figure 4(b)-B1, while there are non-intersect facilities with road polygon on LOD_1 as shown by figure 4(b)-B2.

In this figure, the topologies between roads and facilities marked as a black polygon are consistent with the topologies on LOD_0, even though the topologies on LOD_0 and LOD_1 are not identical each other. However, it is evident that the topologies between roads and facilities with white polygons are inconsistent with the topologies on LOD_0. Consequently these objects should be displaced so that they intersect with road polygons.

In [13, 16], a formal model is proposed to define the consistent topological correspondence between databases of multiple scales. The propagation method is also proposed to correct topologically inconsistent objects by [10, 14].

4.3 Topological Consistency for Aggregation

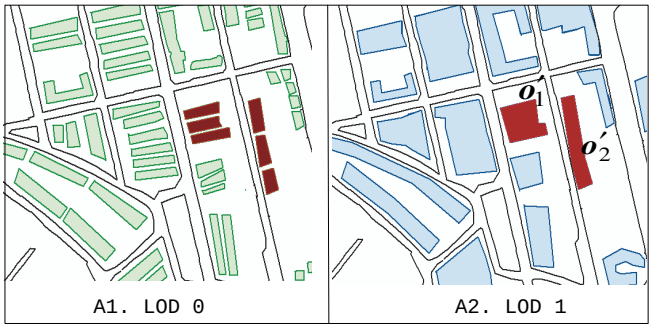
Aggregation operator merges a set of objects satisfying predefined condition into a new object. For this reason, the topology of the original objects must be carefully considered for assessing the topological consistency between LODs. If an update occurs to an object on LOD_0, which is aggregated to an object in LOD_1, the aggregated object in LOD_1 must be re-aggregated taking the update into account. Moreover, the topology on LOD_0 must be respected during the re-aggregation process as shown by figure 5.

In figure 5(a) shows the initial states of LOD_0 and LOD_1, where the polygons in figure 5(a)-A2 are created by aggregation of objects on LOD_0. Then new three objects o_1, o_2, and o_3 are inserted into LOD_0 and the shapes of o_4 and road are changed as shown by figure 5(b)-B1.

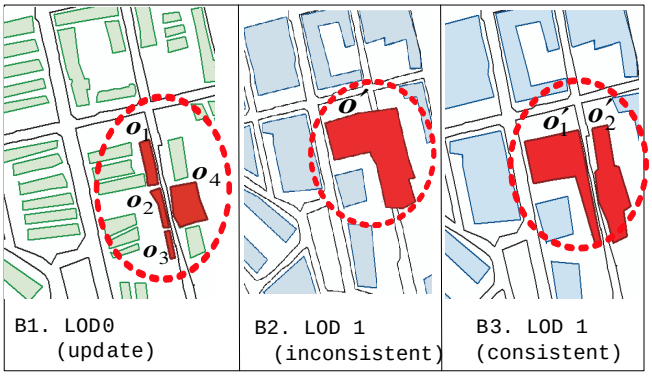
In the process of update propagation, a new aggregated object o' on LOD_1 is computed from updated objects on LOD_0 as depicted by figure 5(b)-B2. But since the topology between the new aggregated object and road on LOD_1 differs from LOD_0, the topology of LOD_1 is considered as inconsistent. In order to make it consistent, the aggregation must be re-computed as shown by figure 5(b)-B3. The derivation method of topology for aggregation is proposed [6], and it can be used to recompute the aggregation with updated or newly inserted objects.

5 Extending SVG to Transfer Updates to Mobile Devices

As we described in the section 1, we should reduce the size of map data for saving communication cost. Therefore, we transfer the updated parts of map to mobile devices, instead of replacing entire map. In this section, we will propose an update schema based on SVG(Scalable Vector Graphics)[17] to describe the



(a) original and derived maps



(b) update of the original map and its propagation results

Fig. 5. Topological Consistency for Aggregation

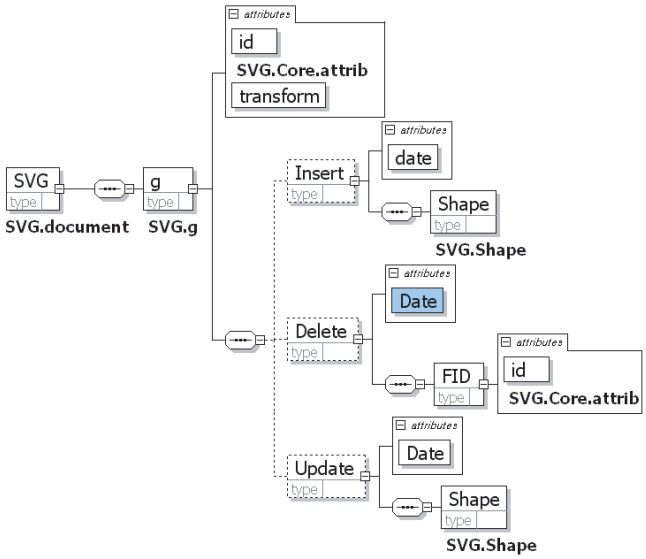
updated parts. The udpate schema is generated by `update manager` on the server in figure 2.

The update schema consists of several tags to describe an update type, an identifier of an updated object and a new geometry. The diagram for the update schema is illustrated by figure 6.

In figure 6, the `SVG` element indicates the beginning of a SVG document, and consists of `g` elements. Each `g` element represents transform conditions for coordinate systems, or describes updated data with an attribute `id`. Each `id` corresponds a map layer, for instance, *RoadFacilityPolygon*.

When the attribute `id` is specified in the `g` element, upate types have to be indicated. Depending on the types of update, different attributes are included in the update schema. For example, updated geometries is needed for `insert` and `update` types of update, but it is not needed for `delete`. Each udpate type has attribute `Date` describing the time stamp of the update transfer.

Udpated geometry is specified by core attributes of a SVG document such as `path`. An example of a SVG extension in figure 6 shows insert of new obejcts into




```
<?xml version="1.0"?>
<svg xmlns:xlink="http://www.w3.org/1999/xlink" x="0" y="0" width="100%" heigh
 <style type="text/css">
  .style-RoadFacilityPolygon-0 {fill:RGB(199,107,240);fill-opacity:1;stroke:RGB(
  .style-Agg-Building-0 {fill:RGB(160,191,156);fill-opacity:1;stroke:RGB(0,0,0);s
  .style-EDUSRC-0 {fill:RGB(222,85,110);fill-opacity:1;stroke:RGB(0,0,0);stroke-
 </style>
 <g transform="matrix(1 0 0 -1 0 857111.936 )">
  <g id="RoadFacilityPolygon">
    <insert date="2005 07 25 11:00"> <!-- Example : insert geometry-->
      <path id="RoadFacilityPolygon_f90" class="style-RoadFacilityPolygon-
      <path id="RoadFacilityPolygon_f91" class="style-RoadFacilityPolygon-
      <path id="RoadFacilityPolygon_f92" class="style-RoadFacilityPolygon-
    </insert>
  </g>
  <g id="Agg_Building">
    <update date="2005 07 25 11:30"><!-- Example : update geometry-->
      <path id="Agg_Building_f2" class="style-Agg-Building-0" d="M 198741.3
      <path id="Agg_Building_f3" class="style-Agg-Building-0" d="M 199594.2
    </update>
  </g>
  <g id="EDUSRC">
    <delete date="2005 07 25 11:22"> <!-- Example : delete objects -->
      <fid id="EDUSRC_f110" />
      <fid id="EDUSRC_f111"/>
      ...
```

Fig. 6. Update schema and example

RoadFacilityPolygon, modification of two objects on *Agg_Building*, and delete of several objects on *EDUSRC*.

This SVG document is to be interpreted by an update handler(2) of MobiMap in mobile device rather than directly displayed by a SVG viewer, since it contains only the updated part of the entire map. The update handler of a mobile device must replace the old data by the new ones in the udpate document to build new map data. But this processing does not result in an overhead of mobile device, since it includes only searching by id and simple replacement operations.

6 Conclusion

With the recent progress of mobile devices, stored map services begin to be commercialized for cellular phone and PDA. In order to ensure the quality and accuracy of map, the updates on the source map must be automatically reflected to mobile devices. However, the expensive communication cost and lack of hardware capacity of mobile devices are important constraints in dealing with the updates. It implies that we should find a compromising solution between the transmission of the entire map and the transmission of only update logs to mobile devices. The transmission of completely processed map results in an expensive communication cost, while simple transfer of update logs requires a large amount of processing to maintain geometric and topological consistency between multiple LODs. Therefore, we proposed a way to transfer udpated objects with SVG documents, neither the entire map nor update logs.

This paper has three contributions. First, we propose a framework of update mechanism in mobile environments where the map data is stored on each mobile device. This framework provides an efficient strategy for processing an update and its propagation to multiple LODs without an expensive cost of communication and large amount of processing at mobile devices. Second, several methods are introduced to maintain topological consistency between LODs depending on the type of generalization operators. The third contribution is the extended SVG to be used as a transfer format of update message to mobile devices.

References

1. M. J. Egenhofer and H. Herring, *Categorizing Binary Topological Relations Between Regions, Lines, and Points in Geographic Databases*, Technical Report, Department of Surveying Engineering, University of Maine, 1990.
2. R. B. McMaster, and K. S. Shea, Generalization in Digital Cartography. Association of American Geographers, 72-28, 1992.
3. M. Egenhofer, *Evaluating Inconsistencies Among multiple Representations*, Proc. 6th international Symposium on Spatial Data Handling, 902-920, 1994.
4. T. Kilpelainen, T. Saarjakoski, *Incremental Generalization for Multiple Representations of Geographical Objects*, GIS and Generalization : Methodology and Practice, Muller, J. P. Lagrange, R. Weibel(editors), Taylor & Francis, 1995.

5. J. Sharma, *Integrated Spatial Reasoning in Geographic Information Systems: Combining Topology and Direction"*, Ph.D Thesis, The Graduate School of Spatial Information Science and Engineering, University of Maine, 1996.
6. N. Tryfona and M. J. Egenhofer, *Consistency among Parts and Aggregates: A Computational Model*, Transactions in GIS, 1(3):189-206, 1997.
7. J. A. C. Paiva, *Topological Consistency in Geographic Databases With Multiple Representations*, Ph. D. Thesis, University of Maine, 1998, http://library.umaine.edu/theses/pdf/paiva.pdf.
8. A. Belussi, M. Negri, and G. Pelagatti, *An integrity constraints driven system for updating spatial databases*, ACM-GIS, 121-128, 2000.
9. Kang H., Do S., and Li K., 2001: Model-Oriented Generalization Rules. Proc.(in CD) ESRI User Conference, San Diego, USA, July.
10. A. Ruas, *Generalization of Updated Data(in the context of multiple representation)*, ISPRS/ICA, 2002.
11. Z. Guo, S. Zhou, Z. Xu, and A. Zhou, *G2ST: A Novel Method to Transform GML to SVG*, ACM-GIS, 161-168, 2003.
12. S. Balley, C. Parent, and S. Spaccapietra, *Modelling geographic data with multiple representation*, International Journal of Geographical Information Science, 18(4):327-352, 2004
13. H. Kang, T. Kim, and K. Li, *Topological Consistency for Collapse Operation in Multi-scale Databases*, Proc. 1st Workshop on Conceptual Modeling for Geographic Information Systems in Conjunction with ER2004, Lecture Notes in Computer Science 3289, Springer-Verlag, 91-102, 2004.
14. C. Duchene, *The CartACom model: a generalisation model for taking relational constraints into account*, Proc. 6th ICA Workshop on progress in automated map generalisation, Leicester, 2004, http://aci.ign.fr/Leicester/paper/duchene-v2-ICAWorkshop.pdf
15. H. Kang, J. Moon and K. Li, *Data Update Across Multi-Scale Databases*, Proc. of the 12th International Conference on Geoinformatics, 2004
16. H. Kang, and K. Li, *Assessing Topological Consistency for Collapse Operation in Generalization of Spatial Databases*, In Reviewing by the 2st Workshop on Conceptual Modeling for Geographic Information Systems in Conjunction with ER2005.
17. W3G, Scalable Vector Graphics(SVG) 1.1 Specification, W3C Recommendation, 2003.