# Quality Inspection and Quality Improvement of Large Spatial Datasets

Hainan Chen and Volker Walter
Institute for Photogrammetry
Universitaet Stuttgart
Geschwister-Scholl-Str. 24D
D-70174 Stuttgart
Germany
Email: firstname.lastname@ifp.uni-stuttgart.de

## Abstract

In this paper we present an approach for quality inspection and quality improvement of spatial data that is based on map matching and map fusion. Two different datasets (GDF/TeleAtlas and OpenStreetMap) are used in this study. At first, the edges in the two datasets are matched manually with a tool developed in VBA and ArcGIS. Then, the form of the matching pairs in both datasets is calculated and segment nodes are searched. Finally, different fusion methods are used, depending on the form of the matching pairs.

**Keywords:** Data Fusion, Matching, Quality Improvement, Quality Inspection

## 1. INTRODUCTION

Spatial data are collected by different institutions for different purposes which lead to multiple representations of the same objects of the world. Multiple representations mean that redundant information is available which can be used for the evaluation and improvement of the quality of the data. In the following we describe an approach for quality improvement based on map matching and map fusion. The approach can be applied for large datasets and can consider not only the geometry of the data but also the attributes and the topological relations.

The paper is structured as following. After a discussion of existing work, the differences of data modeling in GDF and OpenStreetMap are presented. Then, a matching model based on “Buffer growing” is introduced. After this, the automatic recognition of the form of the matching pairs and an automatic node matching are explained in detail. Finally, the data fusion concept is discussed on examples.

## 2. RELATED WORK

In the last decades, different approaches have been developed that merge several spatial datasets into one common dataset in order to improve the data quality. (Lynch & Saalfeld 1985) implemented the first interactive and iterative system for quality improvement that combined two different maps in order to produce a better third map. This process was called conflation.

An automatic conflation approach is described in (Deretsky & Rdony 1993). In this approach, chains of edges are calculated based on an evaluation of their attributes and geometry. The intersections of these chains are treated as relations between the chains. The geometry of matched chains is then transformed into a common dataset using a nonlinear transformation. The attributes are transformed with user defined rules. Then, the maps are divided into small cells which are matched. Specific filters, based on the geometry and attributes, are developed to merge the unmatched objects in each cell of both datasets.

(Cobb et al. 1998) developed a hierarchical rule-based system for conflation, considering the data quality and map scales of the data sources. Feature matching is performed by evaluating the geometrical and semantic similarities. A component based strategy for conflation was proposed in (Yuan & Tao 1999). Components of conflation for specific intentions become interoperable and are able to be developed independently.

The main issue of conflation is to identify the correspondences in different datasets. In (Lupien & Moreland 1987) conflation is divided into two tasks: (1) feature matching and (2) feature alignment. (Walter 1997) developed an algorithm called “Buffer growing” to solve the matching problem. The matchings were subdivided into *1:1*, *1:n* and *n:m* matchings. (Zhang & Meng 2006) extended the matching model of “Buffer growing” with an unsymmetrical buffer.

Rubber-Sheeting (Gillmann 1985) is often applied for feature alignment. (Doytsher et al. 2001) presented an approach for conflation by using linear features instead of point features as counterpart of local rubber-sheeting transformation to keep the shape of transformed objects. (Haunert 2005) interpolated additional points for rubber-sheeting to improve the distribution of control points. (Uitermark 2001) developed an ontology based matching approach.

## 3 TEST DATA

In our study we use two different datasets: TeleAtlas and OpenStreetMap. The TeleAtlas data (TeleAtlas 2005) has been based on the Geographic Data File (GDF) data model (ISO14825 2004) which was developed especially for vehicle navigation systems. OpenStreetMap is a free map project and provides free geographical datasets (OpenStreetMap 2008). A comparison of the coverage of OpenStreetMap and TeleAtlas data is presented in (Fischer 2008).

Due to different data modeling, differences exist in these two datasets. Figure 1 shows the different data modeling in TeleAtlas (left) and in OpenStreetMap (right). It can be seen, that the edges in OpenStreetMap are not subdivided at each intersection. Therefore, the matching between the two datasets is problematic. A preprocessing to overcome this problem is presented in the next chapter.

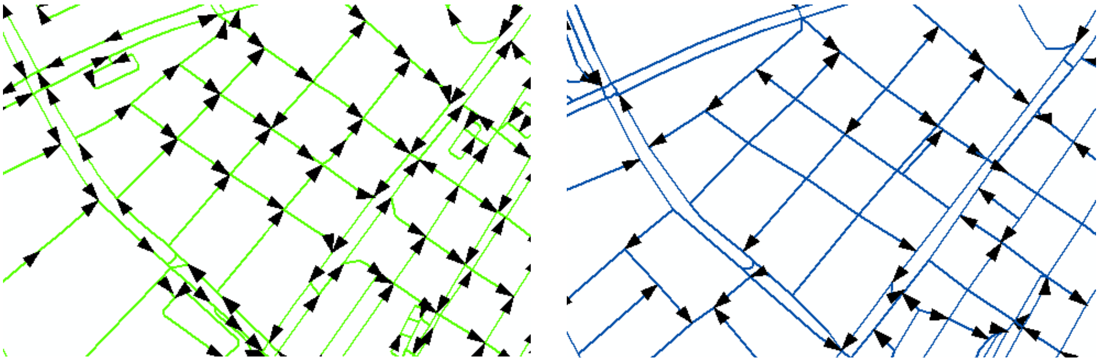


**Figure 1: Data modeling in TeleAtlas (left) and OpenStreetMap (right)**

# 4 DATA FUSION

Figure 2 shows the different steps of data fusion in a flow diagram. The different steps are explained in detail in the following subsections.

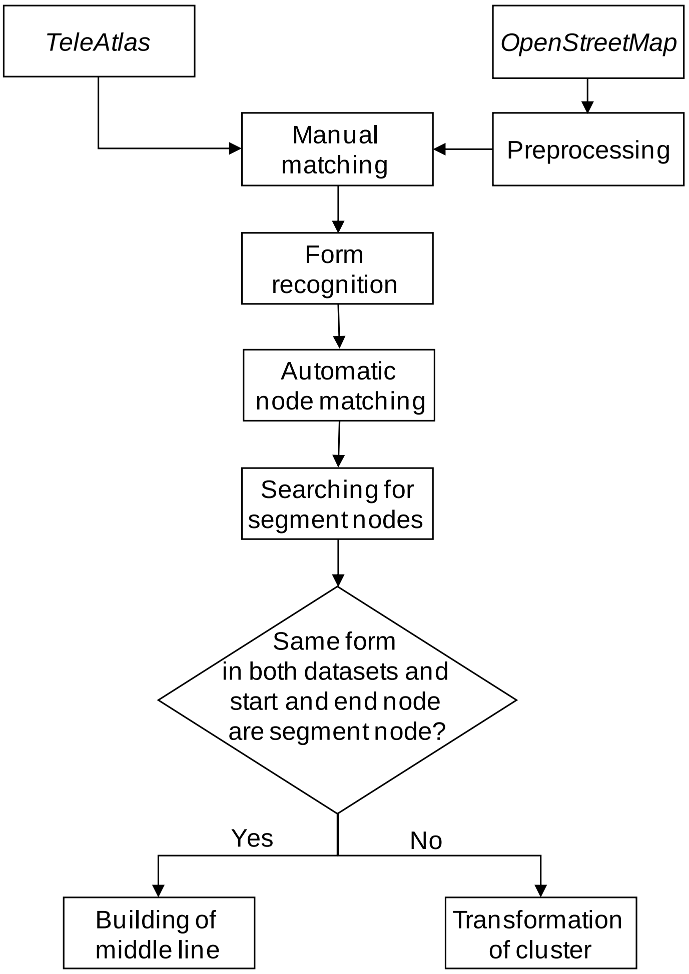

TeleAtlas
OpenStreetMap
Manual matching
Preprocessing
Form recognition
Automatic node matching
Searching for segment nodes
Same form in both datasets and start and end node are segment node?
Yes
No
Building of middle line
Transformation of cluster

**Figure 2: Approach for fusion of matched objects**

## 4.1 Preprocessing

The preprocessing is subdivided into three steps. In the first step, the start and end nodes of all edges in the OpenStreetMap dataset are searched. Then, all intersection nodes are calculated. Finally, the edges are subdivided into subedges according to the intersection, start and end nodes.

Figure 3 shows the original edges of an OpenStreetMap dataset. The green nodes in Figure 4 represent the start and end nodes of the edges. The intersection nodes are shown in Figure 5 in red color. Figure 6 shows the final result of the preprocessing.

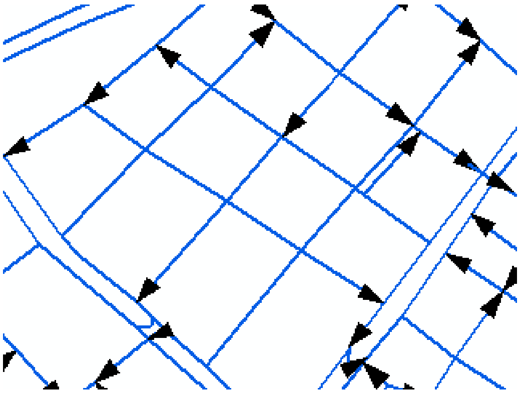


**Figure 3: Original edges in OpenStreetMap**

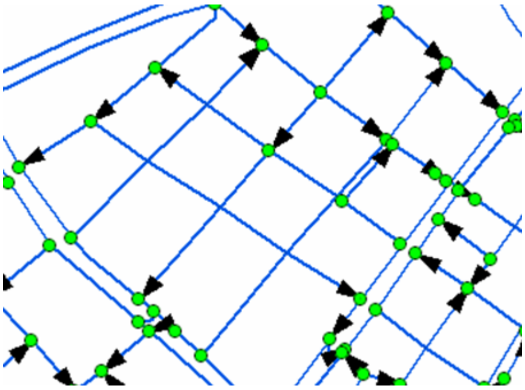


**Figure 4: Start and end nodes of edges (green color)**

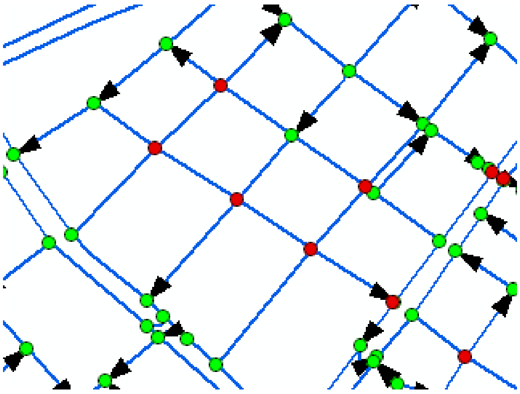


**Figure 5: Intersection nodes of edges (red color)**

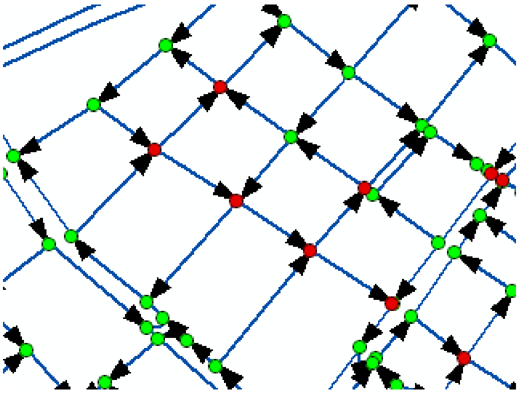


**Figure 6: Edges after preprocessing**

## 4.2 Manual Matching

To consider the topological differences between the two datasets, we extended the "Buffer growing" matching model presented in (Walter 1997) in order that not only matchings between edges but also between edges and nodes are possible. The node $n_1$ in Figure 7 (left) is matched to edge $e_1$ (Relation P:1). In Figure 7 (right) the node $n_1$ is matched to four edges $e_1$, $e_2$, $e_3$, $e_4$ (Relation P:n).

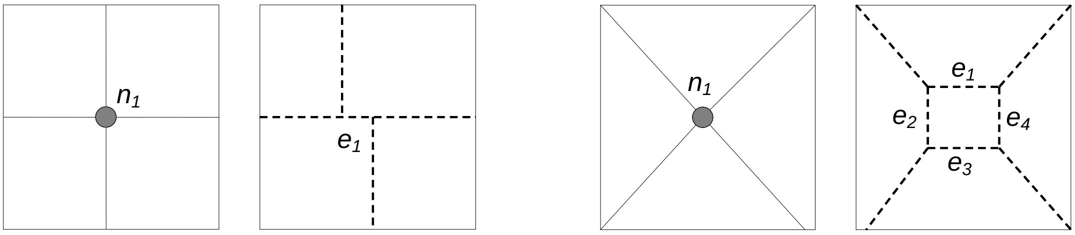


**Figure 7: Matching between one node and one edge P:1 (left) and matching between one node and four edges P:n (right)**

The matching in our study is performed manually with a software tool developed with VBA and ArcGIS. Table 1 summarizes the results of the manual matching and indicates that there are many differences between the two datasets.

**Table 1: Result of manual matching**

| Relation | Test Area I | | | Test Area II | | |
|---|---|---|---|---|---|---|
| | Matching | Tele Atlas Edges | OSM Edges | Matching | Tele Atlas Edges | OSM Edges |
| 1:1 | 408 | 408 | 408 | 52 | 52 | 52 |
| N:1 | 136 | 332 | 136 | 63 | 271 | 63 |
| 1:n | 144 | 144 | 338 | 9 | 12 | 21 |
| N:m | 140 | 401 | 438 | 39 | 153 | 116 |
| 1:P | 21 | 21 | - | 5 | 5 | 0 |
| N:P | 11 | 23 | - | 0 | - | 0 |
| P:1 | 13 | - | 13 | 3 | - | 3 |
| P:n | 1 | - | 4 | 3 | - | 3 |
| 1:* | - | 384 | - | - | 1498 | - |
| *:1 | - | - | 927 | - | - | 81 |
| Total | 874 | 1713 | 2264 | 174 | 1991 | 339 |

### 4.3 Form Recognition

The form of the edges of a matching pair in each dataset can be classified into eight basic classes according to the topology (see Figure 8). The class “Mix” is a combination of two or more basic classes.

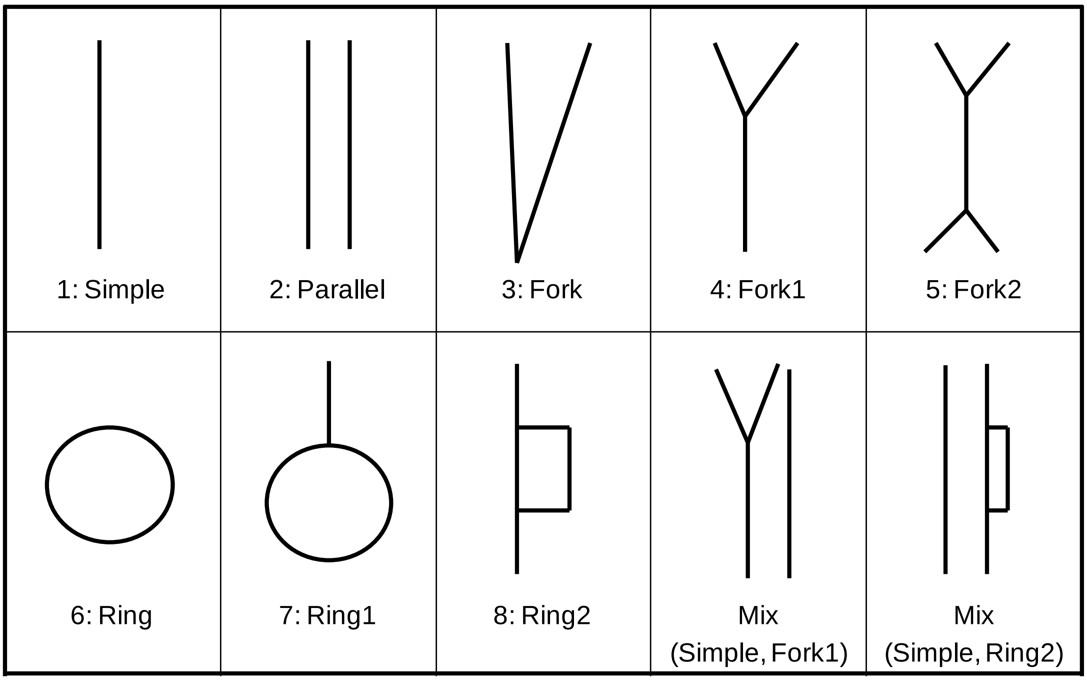


**Figure 8: Different form classes**

To identify the form class, a mini-network algorithm is implemented. For each dataset all edges of each matching pair are converted into a mini-network. The node degree of each node in each mini-network is calculated. According to this degree, the nodes are classified as following:

- *start* or *end node*: degree = 1
- *intermediate point*: degree = 2
- *intermediate node*: degree > 2

Depending on the node types, the mini-network is separated into several parts:

- *begin*: part from *start node* to *end node*
- *middle*: part from one *intermediate node* to another *intermediate node*
- *end*: part from *intermediate node* to *end node*
- *whole*: part from *start node* to *end node*

The mini-network of dataset *A* (solid line) in Figure 9 consists of five parts and includes one “Ring2” (one *begin*, two *middle* and one *end* parts) and one “Simple” (one *whole* part). Therefore, the form class is “Mix”. The form class of dataset *B* (dashed line) is “Simple”, because the corresponding mini-network includes only one *whole* part.

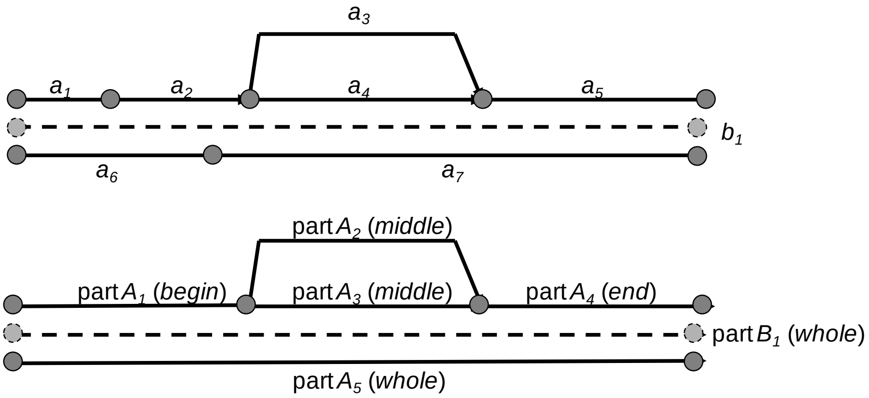


**Figure 9: Parts of mini-network**

## 4.4 Automatic Node Matching

After the recognition of the form, the *start* and *end nodes* of the mini-networks can be matched automatically (Figure 10).

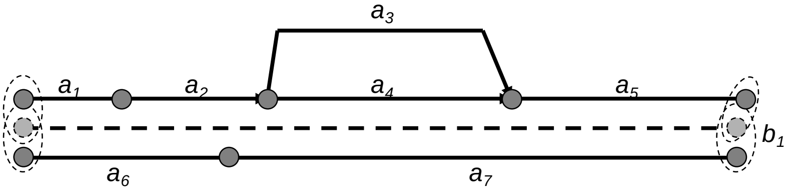


**Figure 10: Automatic node matching**

## 4.5 Searching for Segment Nodes

Because of different topologies, not every matching pair can be merged simply by calculating the middle line. Therefore, all matching pairs with different topologies are allocated into clusters and fused by transformation. The size of the clusters should be as small as possible in order to minimize the complexity of transformation. The clusters and matching pairs are regarded as segments in the following data fusion process.

The nodes, which connect the segments (clusters and/or matching pairs) are called *segment nodes*. In the first step, all node matching pairs with 1:1 relations are inserted into the list of *segment nodes*. For these *segment nodes*, the geometry of the middle point of the corresponding node matching pair is inserted into the final dataset. In the second step, the nodes which are manually matched to edges (P:1 and P:n matchings) are inserted into the list of *segment nodes*.

Figure 11 shows the searching for segment nodes at an example. The dashed lines represent the node matchings. Node $a_1$ of dataset *A* is matched manually to an edge ($b_1$, $b_2$) in dataset *B* (P:1 matching). After the automatic node matching, node $a_1$ is matched to nodes $b_1$ and $b_2$. If the simple form is preferred, the middle point of node $a_1$ (weight 0.5), node $b_1$ (weight 0.25) and node $b_2$ (weight 0.25) is inserted into the final dataset. In case that the complex form is preferred, the translation vector from the middle point of the nodes $b_1$ and $b_2$ to the middle point of $a_1$ (weight 0.5), $b_1$ (weight 0.25) and $b_2$ (weight 0.25) is calculated. Then, the nodes $b_1$ and $b_2$ are transformed with this translation vector and inserted into the final dataset.

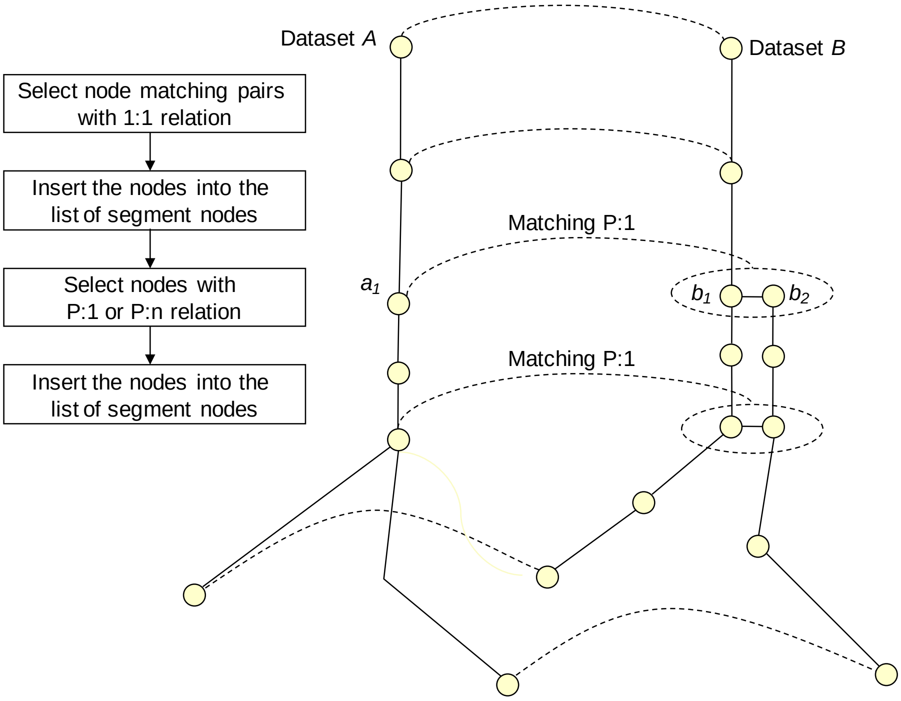

Dataset A
Dataset B
Select node matching pairs with 1:1 relation
Insert the nodes into the list of segment nodes
Select nodes with P:1 or P:n relation
Insert the nodes into the list of segment nodes
Matching P:1
a1
b1
b2
Matching P:1

**Figure 11: Searching for segment nodes**

## 4.6 Building of Middle Line

If the forms of a matching pair in both datasets are “Simple” and the start and end node of them are *segment nodes*, the data fusion is performed by calculating the middle line of this matching pair. The orange lines in Figure 12 represent the matching pairs which are merged by building of middle lines.

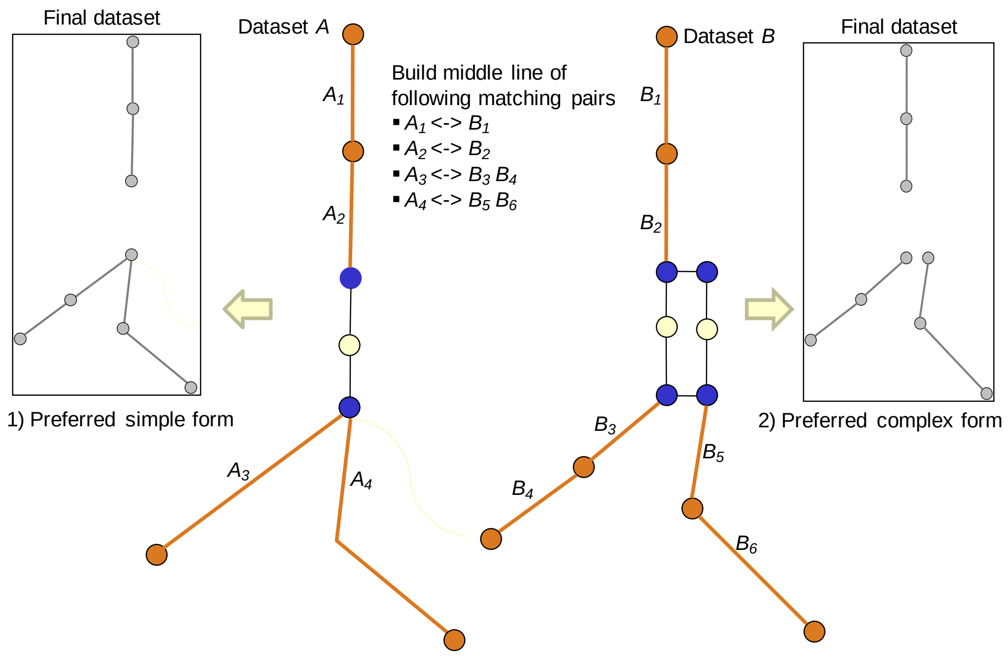


**Figure 12: Data fusion by calculating the middle lines**

In order to calculate the middle line of two lines we use the perpendicular distances. An example is presented in Figure 13. First, all perpendicular distances are calculated from line 1 to line 2 and vice verse. Then, the middle points of the perpendicular lines are calculated and the middle line is built by connecting the middle points.

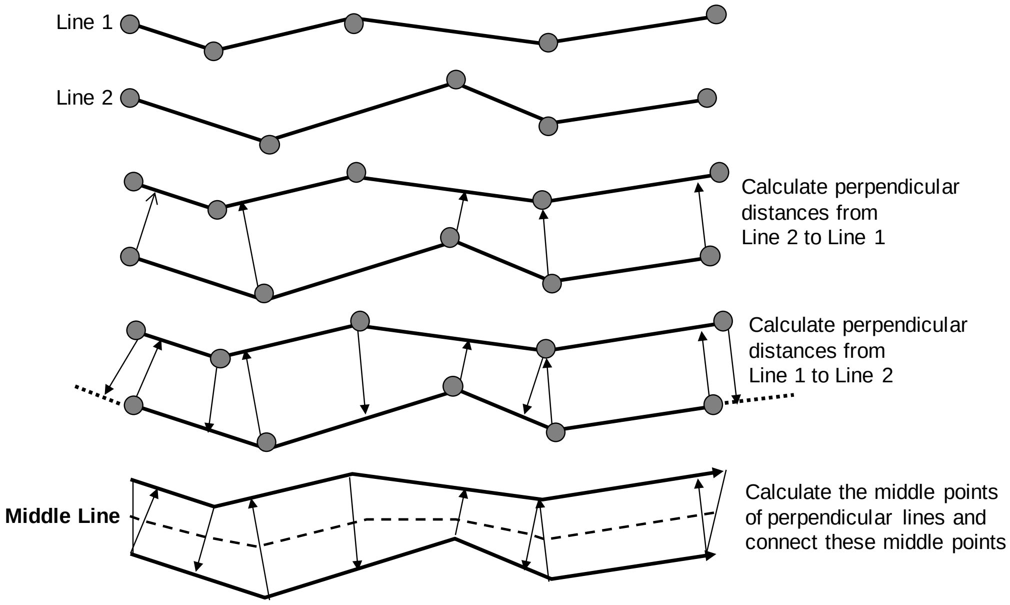

Line 1
Line 2
Calculate perpendicular distances from Line 2 to Line 1
Calculate perpendicular distances from Line 1 to Line 2
Middle Line
Calculate the middle points of perpendicular lines and connect these middle points

**Figure 13: Calculating of middle line**

For matching pairs with connectivity difference, the middle line in the final dataset is extended in order to keep the connectivity. In Figure 14 line 1 ($a_1$, $a_2$) does not connect with any edge at node $a_2$ but line 2 ($b_1$, $b_2$) connects with other edges at node $b_2$. Line 2 is subdivided into two parts (($b_1$, $b_3$), ($b_3$, $b_2$)) according to the perpendicular distance from $a_2$ to line 2. The middle line of ($b_1$, $b_3$) and ($a_1$, $a_2$) is calculated. Then, the part (*$b_3$, $b_2$*) is transformed into the final dataset to maintain the connectivity.

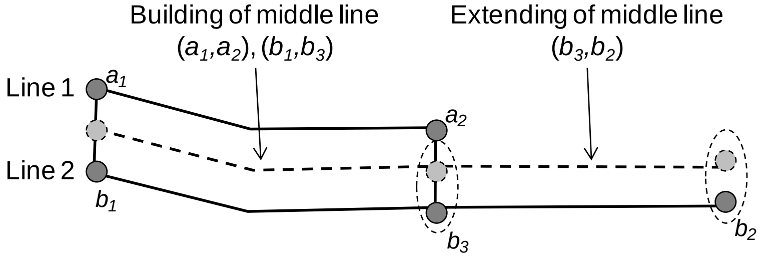


**Figure 14: Extending of middle line**

## 4.7 Transformation of Cluster

The edges of the remaining matching pairs are grouped into clusters according to their connectivity. The algorithm of the transformation of the clusters is described in detail in the following flowchart. After the clustering, the parameters of a Helmert-Transfomation are calculated based on the *segment nodes* in the cluster. Then, the weights of the clusters are computed. Depending on the weights, the edges of the cluster in dataset *A* or dataset *B* are transformed into the final dataset.

```
Build a list for all edges of remaining matching pairs (edge list)
Until the edge list is empty
    Build a list with one edge (list of cluster)
    Until no edge is found which is connected with edges in list of cluster
        Is start node of edge a segment node?
            No:  Search all edges in edge list which contain this start
                 node. Insert the found edges into list of cluster and
                 delete these edges from edge list
            Yes: Do nothing
        Is end node of edge a segment node?
            No:  Search all edges in edge list which contain this end
                 node. Insert the found edges into list of cluster and
                 delete these edges from edge list
            Yes: Do nothing
    Calculate the parameter for Helmert-Transformation according to the segment nodes in list of
    cluster
    Calculate the weights of clusters in dataset A and dataset B
    Transformation the cluster of dataset A or dataset B depending on weights
```

Figure 15 shows an example of a cluster. Depending on the preferred complexity of the form, the cluster in dataset *A* or dataset *B* are transformed into the final dataset.

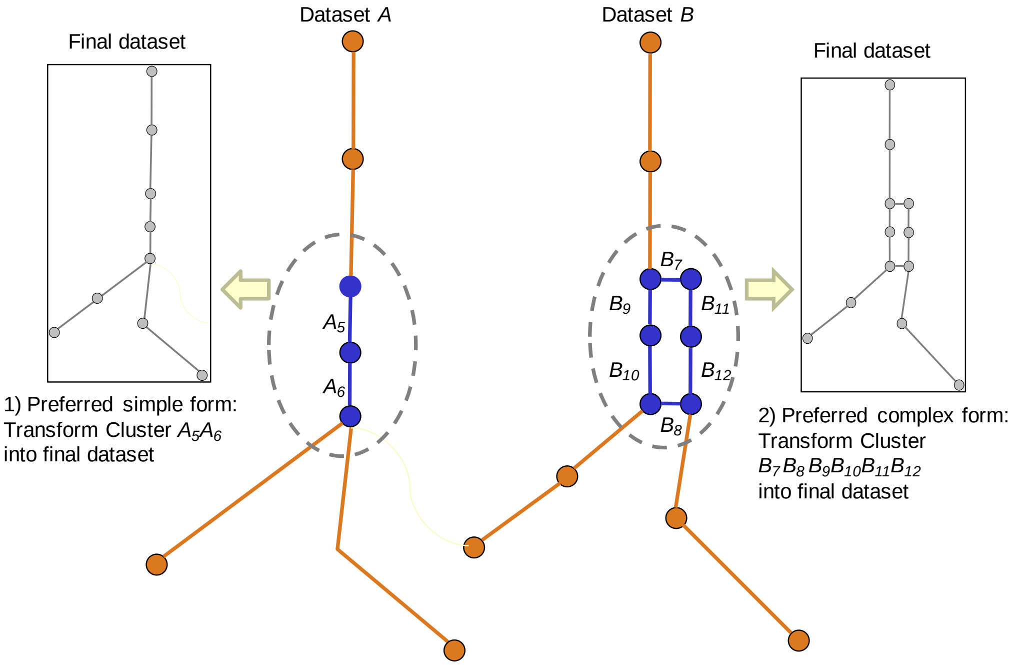
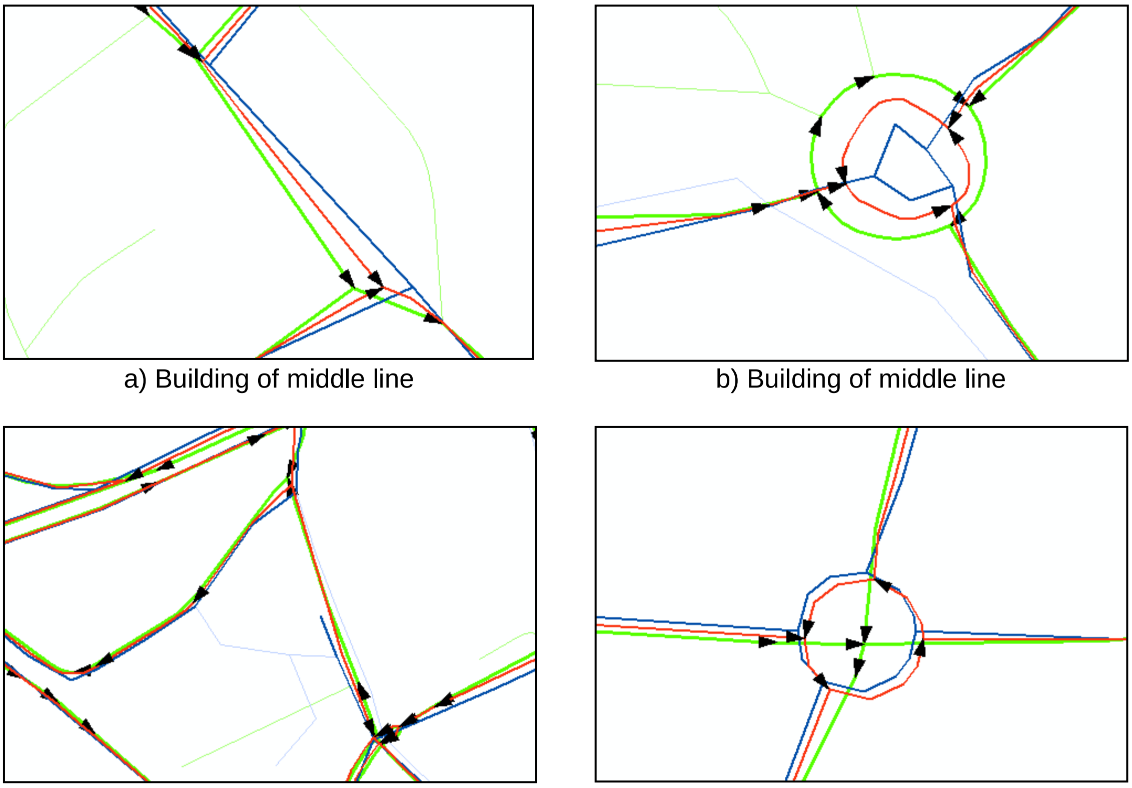


**Figure 15: Cluster example**

## 4.8 Results

Figure 16 shows several data fusion examples. The TeleAtlas edges are represented in green color and the OpenStreetMap edges in blue color. The edges after data fusion are represented in red color. In all examples, the complex form is preferred.

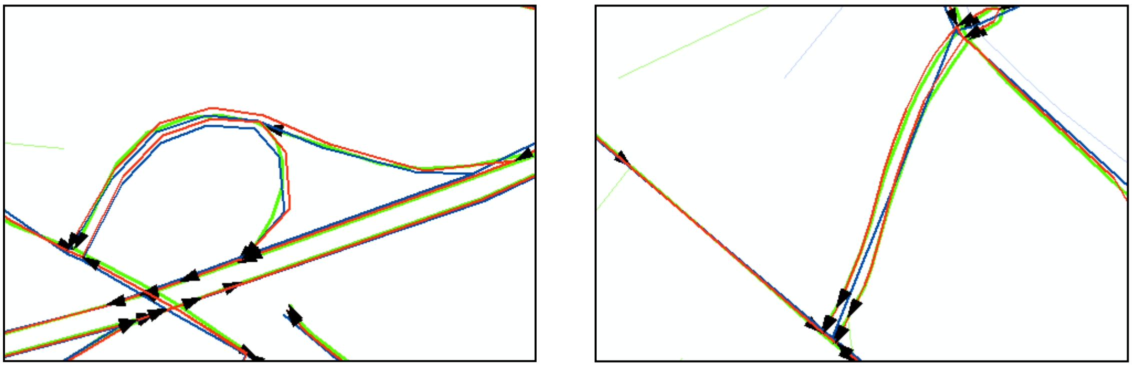


a) Building of middle line

b) Building of middle line

c)Building of middle line

d)Transformation of cluster



e)Transformation of cluster f)Transformation of cluster

**Figure 16: Transformation of Cluster**

Figure 15 a) presents a simple example for fusion by building of middle line. The connectivity of edges maintains after map fusion. In Figure 15 b) the roundabout is divided into three matching pairs and fused by building of middle line. In Figure 15 c) the middle line is extended to keep the connectivity.

In Figure 15 d) the roundabout in OpenStreetMap is represented in TeleAtlas as a node. Therefore, the roundabout in OpenStreetMap is transformed into the final dataset. In Figure 15 e) the cluster in OpenStreetMap (Form: Parallel) is more complex as the cluster in TeleAtlas (Form: Fork1). Figure 15 f) shows a form matching of “Parallel” in TeleAtlas and “Simple” in OpenStreetMap.

## 5. SUMMARY

In this paper we introduced an approach for data quality improvement based on map matching and fusion. In the first part we presented our matching model and our approach for form recognition and automatic node matching. In the second part of the paper we described a map fusion approach for matched objects depending on the form of the matching pairs.

In the future research we will focus on a further investigation of the quality measures and we want to extend the map fusion approach for the fusion of attributes. Conflicts and inconsistencies may appear in fused datasets. We think that a rule-based approach can overcome such problems. Furthermore, the results of map fusion have also to be evaluated using quality measures.

## REFERENCES

Cobb, M. A., M. J. Chung, H. Foley, F. E. Petry, K. B. Shaw & H. V. Miller (1998): A Rule-based Approach for the Conflation of Attributed Vector Data. Geoinformatica, 2/1, 7-35.

Deretsky, Z. & U. Rdony (1993): Automatic Conflation of Digital Maps. In: Proceedings of IEEE - IEE Vehicle Navigation & Information Systems Conference, Ottawa, A27-A29.

Doytsher, Y., S. Filin & E. Ezra (2001): Transformation of Datasets in a Linear-based Map Conflation Framework. American Congress on Surveying and Mapping, 61/3, 159-169.

Fischer, F. (2008): Collaborative Mapping - How Wikinomics is Manifest in the Geo-information Economy. Geoinformatics, 11/2, 28–31.

Gillmann, D. (1985): Triangulations for Rubber-Sheeting. In: Proceedings of 7th International Symposium on Computer Assisted Cartography (AutoCarto 7), 191-199.

Haunert, J.-H. (2005): Link based Conflation of Geographic Datasets. In: Proceedings of 8th ICA WORKSHOP on Generalisation and Multiple Representation, La Coruna, Spanien, published on CDROM.

ISO14825 (2004): GDF-Geographic Data Files-Version 4. Berlin, Beuth.

Lupien, A. E. & W. H. Moreland (1987): A General Approach to Map Conflation. In: Proceedings of 8th International Symposium on Computer Assisted Cartography (AutoCarto 8), Maryland, 630-639.

Lynch, M. & A. Saalfeld (1985): Conflation: Automated Map Compilation, a Video Game Approach. In: Proceedings of Auto-Carto VII, Washington, D.C., 343-352.

OpenStreetMap (2008): OpenStreetMap Homepage. http://www.openstreetmap.org/. Access: October 21, 2008

TeleAtlas (2005): Tele Atlas MultiNet™ Shapefile 4.3.1 Format Specifications.

Uitermark, H. (2001): Ontology-based Geographic Data Set Integration. Dissertation, Deventer, Netherlands.

Walter, V. (1997): Zuordnung von raumbezogenen Daten - am Beispiel ATKIS und GDF. Dissertation, München, Deutsche Geodätische Kommission (DGK),

Yuan, S. & C. Tao (1999): Development of Conflation Components. In: Proceedings of Geoinformatics'99 Conference, Ann Arbor, Michigan, USA, 363-372.

Zhang, M. & L. Meng (2006): Implementation of a Generic Road-matching Approach for the Integration of Postal Data. In: Proceedings of 1st ICA Workshop on Geospatial Analysis and Modeling, Vienna, Austria, 141-154.