Report on the Correction of Erroneous Geometry Data in Land Reuse Projects

Yves Annanias[1] Marc Wahsner[2] Gerik Scheuermann[3] Daniel Wiegreffe[4]

Abstract: Land Reuse processes are large planning and decision-making processes based on a large amount of geographic data. Therefore, it is essential that this data is as accurate as possible. However, errors can occur during the creation of the data and not all of them are directly noticeable. We report here what errors we have encountered while working with this geographic data, what problems they can cause, and how we have fixed them. Since the correction can be very time-consuming with the enormous amount of data, we have focused on an automatic correction. Not all of this data can be corrected this way, for the rest, we briefly indicate a procedure to support and simplify the manual correction.

Keywords: GIS; Data Preparation; Automatic Error Correction

1 Introduction

Opencast lignite mines are temporary, massive interventions in nature since after the coal has been mined, crater landscapes remain. To make these regions usable again and to pursue sustainable concepts, entire stretches of land must be renaturalized. Such concepts are, for example, the agricultural use, the use of these areas for recreation for people as well as nature reserves, and building areas for whole city districts. Whether an area is suitable for recultivation into a certain type of after-use depends on many factors, such as soil characteristics or the location itself. So it may also be that for some areas the subsequent use is not possible due to contaminated sites, while others can be used immediately.

Therefore, a large amount of information has to be considered in detail during planning to decide where and how this land reuse can take place so that it can be done safely and as quickly as possible. The data involved in these land reuse processes consists essentially of two types: geographical data and expert knowledge. The expert knowledge includes information about soil conditions, groundwater levels, forecasts of water level changes of lakes or restrictions on the use of areas (e.g., entering certain areas is prohibited because contaminated soils are present). Geographic data links this knowledge to specific locations on a map. Furthermore, this data also relates individual areas to each other, which allows

[1] Leipzig University, Image and Signal Processing Group annanias@informatik.uni-leipzig.de
[2] Leipzig University, Image and Signal Processing Group mw83waji@studserv.uni-leipzig.de
[3] Leipzig University, Image and Signal Processing Group scheuermann@informatik.uni-leipzig.de
[4] Leipzig University, Image and Signal Processing Group daniel@informatik.uni-leipzig.de

examining the distance between areas, gives information about how far they are from other important points on the map, and sometimes it shows that areas even overlap.

Tools that link these two types of data are called geographic information systems (GIS). A GIS can provide many ways to measure, interpret, visualize, generalize, and interpolate collected samples [GYC07]. Specifically, spatial interpolation techniques are used to complete those spatial locations where measurements are not available or missing. GIS has penetrated a variety of fields supporting experts in decision making and are successfully used to create continuous surfaces in, e.g., meteorology, climatology, and water management (see Kienberger and Steinbruch [KS05] with focus on cyclones, floods and droughts, Abbas et al. [Ab09] for the combination of disaster management with a GIS for a district disaster management system for floods, or Khan [Kh13] for flood analysis and prediction using GIS). In addition, Chen et al. [Ch18] use machine learning to create landslide susceptibility maps to reduce this hazard if possible or to integrate this information into land-use planning. In general, the aforementioned works show that especially in disaster risk management GIS can be useful, this concerns, e.g., the different phases of risk management activities, such as planning, mitigation, preparedness and response (see Bala et al. [BT17]).

Since these planning and decision-making processes depend heavily on geographic data, these data must be as accurate as possible. Otherwise, erroneous data would lead to incorrect decisions that unnecessarily delay the land reuse process, which can be costly and result in legal consequences or, especially in the case of disaster management, endanger human lives. Unfortunately, errors in the geographic data are not always directly detectable. For example, a GIS may visually represent an area supposedly correctly, but underlying algorithms generate incorrect values when calculating the area's content and other parameters. Furthermore, calculations of intersections may be wrong or are not possible at all, because the algorithms already abort with error messages before due to various reasons. As a result, important information would be missing from the decision-making process. Finding and correcting these errors is very time-consuming.

To facilitate error detection and correction, we provide a brief overview of the specific errors that can occur, what causes them, the problems they cause, and how to fix them. A large part of the errors are due to syntactic inaccuracies and can therefore be corrected automatically, which saves a lot of time. Some GIS tools offer support to detect syntactical errors [ESRa], some are able to fix them directly, like incorrect ring orientations. Even self-intersections are partially adjusted automatically by them [ESRb]. However, changes in the shape of a geographic area might occur here depending on the algorithm used. Cui et al. [Cu20], instead, specify an algorithm that can be used to split self-intersecting polygons into partial polygons without self-intersection. Due to insufficient modeling possibilities at the time of creation of the data, it is possible that semantic errors were explicitly used to create a certain visual result. An automatic correction can then lead to different results. Therefore, it is reasonable to check and correct this geographic data manually by an expert. For this, we provide a structured procedure, with the help of which an expert is guided during the manual correction.

2 Data

The geographical data follows the Simple Feature Access specification [Ope], and is called a feature, here. In general, a feature could be a point (e.g., a well), a line (e.g., a road), a polygon (e.g., an area or an area with a hole), or a multi-polygon (e.g., two areas without a connecting line). To store and process these geometry types the specification includes a definition for representation of the geometry as a well-known-text (WKT) format. The different types of geometry can thus be represented as follows:

Simple Point
$Point(0, 0)$

Line
$LineString(0\ 0, 100\ 0, 100\ 100)$

Simple Polygon
$Polygon((0\ 0, 0\ 100, 100\ 100, 100\ 0, 0\ 0))$

Polygon with a Hole
$Polygon((0\ 0, 0\ 100, 100\ 100, 100\ 0, 0\ 0), (20\ 20, 20\ 80, 80\ 80, 80\ 20, 20\ 20))$

Two Polygons
$MultiPolygon(((0\ 0, 0\ 100, 100\ 100, 0\ 0)), ((20\ 0, 120\ 100, 120\ 0, 20\ 0)))$

The data set provided to us includes 36,623 features and was obtained from the Lausitzer und Mitteldeutsche Bergbauverwaltungsgesellschaft mbH (LMBV)[5]. The LMBV is a company which is responsible for the management and recultivation of abandoned opencast lignite mines, and one of its tasks is to provide the public with reliable information about the areas it manages. The provided data set was created continuously over several decades using various input methods and software systems, and large portions of the data were created manually. The large majority of features are areas described by polygons and multi-polygons (33,955 features, 92.7%). The remaining features are wells and other features described by point data (2,668 features, 7.3%). There are a few polygons with an area smaller than 10 m^2. However, there are also a few particularly large polygons with areas up to $5,000\ km^2$.

The polygons and multi-polygons describing areas were generated in different ways. One way was to use GIS stations that displayed a map and offered the possibility to create points on it in order to connect them afterwards via lines. The problem with this is that sometimes points were created by mistake, resulting in additional unnecessary lines. Another problem lies in the fact that especially older stations were only able to create simple polygons. For more complex polygons, workarounds had to be used, such as unnecessary connecting lines or self-intersecting shapes, since these visually showed the desired result but did not adhere to the correct format. In a second way, a person walked along the border of the area. Using

[5] https://de.wikipedia.org/wiki/LMBV

a GPS tracker, points on that border were stored at regular time intervals. This also can result in problems that are not always visually recognizable (e.g. redundant points caused by standing longer at the same point). Redundant points and lines, as well as self-intersecting polygons, cause problems, e.g., when intersecting polygons with each other. This error handling is discussed in the following section.

3 Error Detection and Handling

Defects within the geometry of features are usually difficult or sometimes impossible to perceive visually and can impede the processing of those features. With the aforementioned format, it is possible to solve these errors by simple string substitutions. An example that occurs in the data is related to the creation of the polygon on a map. Here, two points were created on a map and connected to another one. By mistake, another point was then created on this line and connected to the others, creating unnecessary and redundant lines. The result of this is shown in Fig. 3 (top), visually there is only one line recognizable and the error is therefore difficult to detect later. The problem lies in the so created self-intersection, whereupon some algorithms cause problems because they deliver incorrect values when the size of an area is calculated or even stop working when the overlap with another polygon should be calculated.

In this section, we indicate how these and other problems within the geometry of features could be solved. At first, a definition of validity is required. Here, the validity of the geometry is determined using the *is_valid* function of the *shapely python package* [Gi], which in turn uses an *OpenGIS* specification [Ope] to define validity. For example, a polygon is defined by one exterior border and zero or more interior boundaries (see $Polygon$ and $MultiPolygon$ definitions in section 2). A valid polygon has to fulfil certain conditions, the most important ones here in terms of errors appearing in the data set are the following (for more conditions, see the specification [Ope]):

a “Polygons are topologically closed.”

b “No two rings in the boundary cross and the rings in the boundary of a polygon may intersect at a Point but only as a tangent [...]”

d “A polygon may not have cut lines, spikes or punctures [...]”

While some errors are only tagged and have to be dealt with manually, the majority allow for automated processing.

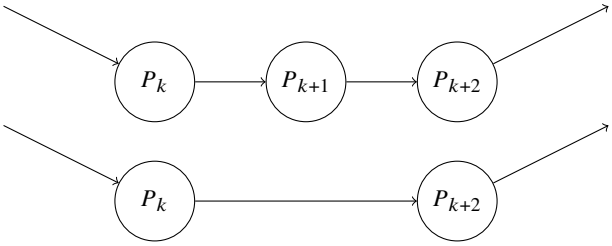

Fig. 1: Polygon with redundant point P_{k+1} (top) and fixed polygon (bottom)

3.1 Preprocessing

Prior to fixing any defects, the following redundancies are removed to allow for easier processing:

1. Duplicate Points: Whenever two adjacent points within the geometry of a feature are identical, one of those points is removed. Therefore, $Polygon(\ldots, 0\,0, 0\,0, \ldots)$ will be simplified to $Polygon(\ldots, 0\,0, \ldots)$.

2. Redundant Points: Whenever a triplet of adjacent points $(P_k, \underline{P_{k+1}, P_{k+2}})$ exists within the geometry of a feature, where P_{k+1} lies on the line segment $\overline{P_k P_{k+2}}$, P_{k+1} is removed. This constellation can be seen in Fig. 1. Therefore, $Polygon(\ldots, 0\,0, 0\,1, 0\,2, \ldots)$ will be simplified to $Polygon(\ldots, 0\,0, 0\,2, \ldots)$. However, it is assumed here that point P_{k+1} lies exactly on the line segment. Numerically, this may not always be exactly the case. For this an epsilon environment can be used as a threshold and it is assumed that the point belongs to the line segment if it lies within this threshold. However, such simplification can lead to problems, see section 4.
   A redundancy that involves the first and last point of the polygon is currently not being fixed, as their can only exist one such defect within a polygon, which does neither affect the automated processing, nor does it prevent any other redundancies from being fixed. Hence, the $Polygon(0\,2, 0\,3, \ldots, 0\,1, 0\,2)$ is not simplified to $Polygon(0\,3, \ldots, 0\,1, 0\,3)$. Two adjacent points are guaranteed to be non-identical because of the prior removal of duplicate points. Without this restriction, it would not be sufficient to merely check triplets of points to detect redundancies. In case of $P_k = P_{k+2}$, P_{k+1} can not be on the line segment $\overline{P_k P_{k+2}}$. However, this case is considered to be a defect and a more general case will be automatically fixed in the next subsection.

3.2 Automated Processing

Although shapely is able to identify many invalidity reasons, the only invalidity that occurs and allows for automated processing is *self-intersection*. Here the following cases of *self-intersection* are automatically processed:

1. Within polygons and multi-polygons: Whenever a triplet of adjacent points (P_k, P_{k+1}, P_{k+2}) exists within a polygon, where the normalized vector $\hat{v}(P_k, P_{k+1}) = -\hat{v}(P_{k+1}, P_{k+2})$, P_{k+1} is removed. This operation can influence the shape of a feature, as can be seen in Fig. 2, but it will not change its area. This defect may also occur at the end of a polygon of size n, with $\hat{v}(P_n, P_1) = -\hat{v}(P_1, P_2)$, here P_1 has to be removed. It should be noted here, that such special lines can be meant as accesses to areas, e.g., paths and roads to the area. However, this should be created as extra line segments or as an extra polygon with a given width. Following our experts, however, this can actually be seen as an error in the present data set and corrected automatically. The removal of P_{k+1} may lead to P_{k+2} being made redundant, due to the following configuration: The points $P_k, P_{k+1}, P_{k+2}, P_{k+3}$ lie on a line in the order $P_k, P_{k+2}, P_{k+1}, P_{k+3}$, as is shown in Fig. 3. In this case P_{k+2} is also removed.
2. Within multi-polygons: If two polygons A, B of a multi-polygon have a polygon as an intersection, the first Polygon is reduced to $A = A - B$. This step is shown in Fig. 4.

3.3 Manual Processing

All invalid features that could not be processed automatically are tagged accordingly and can be fixed through user interaction. The hourglass serves as an example of such an invalid feature (see Fig. 5 left). In principle, it consists of two triangles that touch at one point. In the example shown, the direction in which the points are indicated also changes for each triangle. This has the consequence that the calculated area is incorrectly indicated as 0. In addition, the present self-intersection leads to further errors when intersecting the feature with other features.

An automatic correction would first correct the order of the points. Then it is possible to divide the intersection point into two points that are as close to each other as possible. The error handling described before is no problem here, even if a threshold is used when removing redundant points, since only points on a line segment are affected, which are thus directly connected by edges. The points I_1 and I_2 lie close together, but not on a line segment by connected points. This solution would solve the two problems mentioned above, which is shown in Fig. 5 (right). Nevertheless, this changes the shape of the polygon and an alternative is to create two single polygons from the two triangles, which then are combined into one multi-polygon.

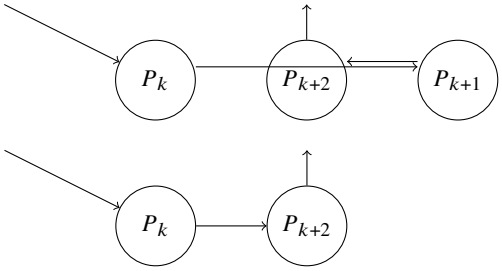

Fig. 2: Polygon with faulty point P_{k+1} (top) and fixed polygon (bottom)

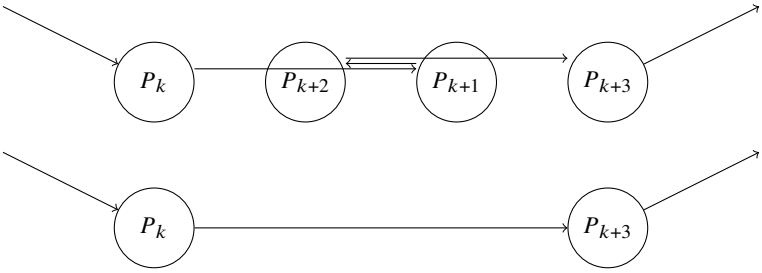

Fig. 3: Polygon with faulty point P_{k+1}, where the removal of P_{k+1} leads to P_{k+2} being made redundant (top) and fixed polygon (bottom)

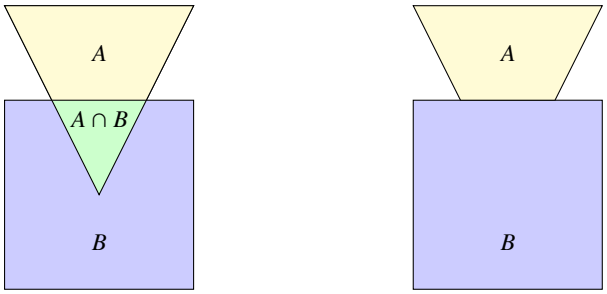

Fig. 4: Faulty multi-polygon (left) and fixed multi-polygon (right)

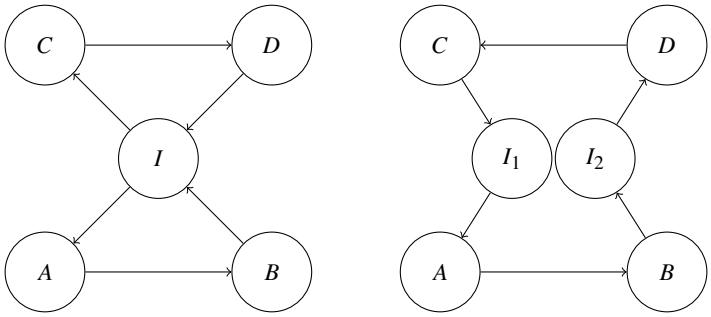

Fig. 5: **Left:** Hourglass polygon with self-intersection in point I and opposite point ordering in both triangles. The area of this polygon is calculated as 0. **Right:** The polygon with corrected point ordering and resolved self-intersection, where point I has been divided into the two points I_1 and I_2, which are very close to each other. Therefore, the area is correctly calculated as ≈ 2 (if the area of each triangle is 1).

Not all of these problems are as simple as this one. Especially with polygons that have holes, changes could quickly lead to an altered representation and subsequently, the question arises as to what the polygon was originally intended to represent. In consultation with our experts, it is better to handle these cases manually. To simplify the process of manual correction, we use the following procedure:

1. Check feature for defects.
   a) If there are no defects or they can be corrected automatically, correct them and calculate the required values.
   b) If there are defects that need manual correction, tag the feature and go to the next step.
2. Mark the invalidity reason (a point which marks the start of the defect as determined by the *shapely python-package*).
3. Display invalid feature on a map and highlight marked position.
4. Allow the feature to be modified by an expert by repositioning, inserting, and deleting points.
5. After the feature was modified, go to 1. and check for other defects.

4 Large Polygons

Another problem we noticed, concerns polygons with a lot of points. As an example, there was a railroad track in the data set that was recorded using a GPS tracker. The tracker drove along the railroad track from start to end and back, recording data after each specific time interval. The result is a polygon that extends over a distance of approximately 400 km and contains over 66,000 points. Although this polygon is syntactically correct after error correction, it can lead to high memory and time consumption in other operations. In addition, the high number of points may degrade the response time of the GIS tool.

A possible solution would be, on the one hand, the simplification of the geometry. In this way, points can be removed that do not lie on a straight line, but where the angle between the lines and the neighbouring points is not too large. However, this changes the surface and is therefore not always an acceptable solution. In addition, such simplification can again lead to the previously mentioned problems, such as self-intersection. Therefore, rechecking for these defects after the adjustment is necessary. On the other hand, the geometry can be divided into several smaller parts. In this way, the entire polygon does not have to be considered when displaying it, but all parts must still be taken into account in the area calculation.

5 Conclusion

As a result, we were able to correct a significant amount of data from our data set, which consists of 36,623 features. Of these features, 30,278 (82.7%) were completely error-free. 5,802 features (15.8%) had errors that could be corrected automatically, although not all of these errors were problematic. For example, simple redundancies of points on a straight line do not cause problems. Only 543 features (1.5%) had to be adjusted manually, and the process of correction was supported by tagging the defects. The error handling ensured that all the data could be used and lead to correct results. Thus, the intersection of areas was also possible without any problems. This leads to further and reliable results of the data when used in land reuse processes. As a result, data created several decades ago can still be used.

ACKNOWLEDGMENT

This research was partially funded by the Development Bank of Saxony (SAB) under project number 100335729.

Bibliography

[Ab09] Abbas, S.H.; Srivastava, R.K.; Tiwari, R.; Ramudu, P.: GIS-based disaster management: A case study for Allahabad Sadar sub-district (India). Management of Environmental Quality: An International Journal, 20:33–51, 01 2009.

[BT17] Bala, Papiya; Tom, Santhi: GIS and Remote Sensing In Disaster Management. Imperial journal of interdisciplinary research, 3, 2017.

[Ch18] Chen, Wei; Peng, Jianbing; Hong, Haoyuan; Shahabi, Himan; Pradhan, Biswajeet; Liu, Junzhi; Zhu, A-Xing; Pei, Xiangjun; Duan, Zhao: Landslide susceptibility modelling using GIS-based machine learning techniques for Chongren County, Jiangxi Province, China. Science of The Total Environment, 626, 06 2018.

[Cu20] Cui, Yong; Liu, Qian; Chen, Guo; Zhang, Hujun: A general method for decomposing self-intersecting polygon to normal based on self-intersection points. Theoretical Computer Science, 842:118–129, 2020.

[ESRa] ESRI Inc. ArcGIS Pro: Invalid Geometry. URL: `https://pro.arcgis.com/en/pro-app/latest/help/data/validating-data/invalid-geometry.htm` [accessed 2021-07-08].

[ESRb] ESRI Inc. ArcGIS Pro: Repair Geometry (Data Management). URL: `https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/repair-geometry.htm` [accessed 2021-07-08].

[Gi] Gillies, Sean: Shapely: Manipulation and analysis of geometric objects in the Cartesian plane. URL: `https://pypi.org/project/Shapely/` [accessed 2021-04-08].

[GYC07] Goodchild, Michael F.; Yuan, May; Cova, Thomas J.: Towards a general theory of geographic representation in GIS. International Journal of Geographical Information Science, 21(3):239–260, 2007.

[Kh13] Khan, Naveed: Flood Prediction and Disaster Risk Analysis using GIS based Wireless Sensor Networks, A Review. Journel of Basic and Applied Scientific Research, 09 2013.

[KS05] Kienberger, Stefan; Steinbruch, Franziska: P-GIS and disaster risk management: Assessing vulnerability with P-GIS methods–Experiences from Búzi, Mozambique. 01 2005.

[Ope] Open Geospatial Consortium. OpenGIS Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture. URL: `https://www.opengeospatial.org/standards/sfa` [accessed 2021-04-08].