A WAY TO STORE, QUERY, UPDATE AN INSAR SPATIOTEMPORAL DATASET

Ze Li

Delft University of Technology, Civil Engineering and Geosciences Faculty

KEY WORDS: InSAR database, points, feature category, query, update, Python

ABSTRACT:

In SAR interferometry, the data acquired by satellite are large. The points in the dataset are usually expressed in spatial-temporal dimension in which many epochs may be included. Extra information including contextual values and data from other sources sometimes are important to be taken into account when analyzing the deformation based on the time series. However, this kind of extra information is not supplied in the dataset. Here we show a way to augment the InSAR dataset by updating the new acquisitions, adding contextual values and combining data with other sources. We create an enriched dataset which provides one more knowledge on the points in the dataset, such that more accurate analysis and decision can be made. Also, a way to query a certain bunch of points with the same feature category is descripted, which allows one to find the points of interest based on their features. Our result provides a way to utilize multiple data sources and to combine them into an InSAR dataset in order to make the dataset be more efficiently used. The way depicted in this paper might be a hint of how to do a more accurate and comprehensive analysis based on a dataset. Furthermore, an automatic or semiautomatic way to modify the dataset with some tools are helpful.

1. INTRODUCTION

Satellite based information products obtained by SAR(Synthetic Aperture Radar) interferometry time series contain large volumes of data. The two main dimensions are space and time. In space, every measurement point has a coordinate (longitude, latitude and height). In time, each point may exhibit motion: it may move in the line-of-sight of the radar instrument, i.e. displacements. The spatial and temporal attributes of measurements of persistent scatters are usually combined together to make the dataset more enriched. In reality, the size of measurements and spatial-temporal attributes can reach 1 million points and 200 epochs respectively. That would be a huge work load to query the information of a point if no effective query method is used. The large volume of available observations presents us with a number of challenges, including the characterization and treatment of data management and practical considerations of how to minimize the computational cost for updating and modifying such large data sets (Lohman , Simons, 2005).

In the current state of the art, we are able to obtain much more information about our measurement points. For example, for one particular measurement point, it might be a soil-type point other than a point on a building. This is important information when one studies the deformation of a point while the function of that point should be taken into account, such that necessary actions for protecting human vital properties are taken. This kind of information, however, is not given in most datasets. In this study, feature categories are appended to the InSAR dataset. One of the most vital functions of the deformation values in time series is, by studying the variations of the height of certain points in an area, we are able to detect some motions of a particular point which may lead to collapse of a building. However, sometimes the decision made only based on the measurement is not quite reliable as the point might be acquired in a severely weather or it experienced some changes resulted from human activities. Thus, this kind of extra information needs to be added to the original dataset.

Thus, the original database of measurement points should be augmented by other information that we deem relevant, which is not done for most datasets. One may not make a perfect decision or analysis based on the incomplete” dataset. In this paper, we investigate a way to query, update, and modify the data structure to achieve the goal mentioned above. Also, an idea is put forward to generate a more established data structure with geometric relationships(Sammon, 1969), giving one a hint of how to utilize other data sources together with an InSAR dataset.

In section 2, methodologies used in this paper, including the data sources and the tools are provided. The results are shown in section 3 with the outputs after applying the methods in section 2. One application on Limburg of the Netherlands is also given in section 3. Conclusion and discussion are made in section 4.

2. METHODOLOGY

Each point is located on a geo-object. Thus, the points which on the same geo-object can be collected together. One can connect to the points of interest easily by this way and add whatever contextual information they want to them. To do so, the data table should be scalable and modifiable. What's more, feature categories are supposed to be added as an attribute of the corresponding point at that location. Thus, these points on a certain geo-object can be selected very fast and directly.

In practice, there might be an event that happens before or after the acquisition date, such as an earthquake, unexpected subsidence or deformation, volcanic activities, or any changes by human activity. Moreover, the atmospheric and environmental conditions can also make a big difference on the quality of the data, or more directly, on the states of the geo-objects. These influencing factors on the dataset need to be taken into account when adding attributes to a point on a geo-object, as the values of the data might be affected by them. To add these additional values to the dataset, a structure storing the InSAR data with a good logic and organization is mentioned in this paper.

In this section, data sources used as an instance are described first. The tool and programming language are introduced in the second subsection. Then in the following subsections, the methods of querying, updating and modifying a dataset are provided.

2.1 Data Source

The input Multi-temporal InSAR dataset is a csv file which contains temporal and spatial information of million InSAR points. The longitudinal, latitudinal and vertical values of the points are given in the table as well as their corresponding vertical deformation ordered in time series. The database is an object database(ODB), as the relationships are determined by the relationship properties or reference attributes that include object ID(s) of the related objects(Elmasri , Navathe, 2017). Thus, the properties or the attributes are references of ID(s) to the related objects.

Compared to relational database, in which relationships among tuples (records) are specified by attributes with matching values, ODB is used efficiently by specifying the ID of related objects, so that their information can be displayed easily. The attributes of the relational database can be considered as value references and are specified via foreign keys, which are values of primary key attributes repeated in tuples of the referencing relation (Elmasri , Navathe, 2017). This is also a good point for our database if we connect the database with other tables that contain the states, the contextual values and the weather information when the points are collected and so forth.

In this project, the points which are located in Groningen are selected as examples. Because of the gas fields here and its seaports, deformation might happen in both a smooth way or a sudden way. Therefore, it is important to have a good view on how the InSAR points vary at this place.

Another important dataset used in this project is the basic registration large-scale topography from Public service On the Map (PDOK), which is a Dutch platform for accessing reliable geo-data from the government for both public and private sectors. The topography can be downloaded by choosing the desired location inside the Netherlands.

2.2 The DBMS and Language Used in This Study

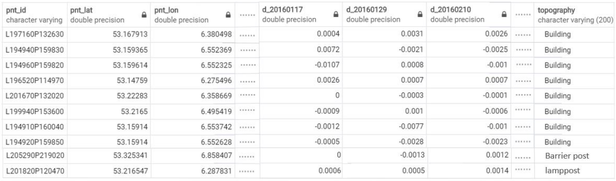
The Database Management System(DBMS) is used to manage InSAR data. This domain-specific tool is good at handling data in the relational database management system, especially for structured data. In this study, PostgreSQL is used. PostgreSQL is a powerful, open source object-relational database management system (Wikipedia, 2020). It is reliable and robust with good performance on query, storage and modification on a dataset.

A data table can be created by PostgreSQL with the same features as the input excel table and it is stored in local directory as a ".file" file. However, one can easily open and use it via SQL shell or in PgAdmin4, which is an open source management tool for PostgreSQL software. Almost every command in SQL shell can be executed in PgAdmin4, but with a better visualization. One thing to be noted is that the names of columns of the data table should be the same as that in the excel table in order to make sure the right information is assigned to the right columns when one creates a table. This procedure can be done in around 3 minutes for more than 2 million points with 175 time series deformation, namely the data matrix including spatial information and point ID can reach $[2million \times 179]$ or so.

One important goal of Python is to add the feature category information from platform PDOK to the data table and update its values. Before that, a coordinate reference system transformation needs to be done for the topographical layers, i.e. transforming CRS(Coordinate Reference System) from "EPSG:28992, Amersfoort/RD New" used in Netherlands to "EPSG:4236-WGS 84" to ensure they are in the same CRS.

2.3 Query on a Dataset

Given the name of a record point, the information of that point can be accessed by querying it in PgAdmin4, or even access PostgreSQL in python. Both of the ways allow one to query certain data among millions of points easily. In practice, one might not study all the points in the image, but instead the points in an area of interest, or even the points which represent a certain type of geo-object, like a building, lamppost, the road and so forth. It makes more sense to use the points standing for point-like objects when one wants to register two InSAR images. Because the error is much likely smaller when using point-like objects than using other objects like buildings, houses, as the point-like object has a smaller width. Currently most datasets do not include these information themselves, one has to find out it from other data sources. Therefore, it is not efficient to query a certain type of points. In this study, a way via Python to query the data/points that satisfy a statement is given. This method allows to query certain data combining the data from PDOK among millions of points. For example, given a statement ”find all the points on buildings and lampposts”, the points located on buildings are selected as the picture below,



pnt_id character varying	pnt_lat double precision	pnt_lon double precision	······	d_20160117 double precision	d_20160129 double precision	d_20160210 double precision	······	topography character varying (200)
L197160P132630	53.167913	6.380498	······	0.0004	0.0031	0.0026	······	Building
L194940P159830	53.159365	6.552369	······	0.0072	-0.0021	-0.0025	······	Building
L194960P159820	53.159614	6.552325	······	-0.0107	0.0008	-0.001	······	Building
L196520P114970	53.14759	6.275496	······	0.0026	0.0007	0.0007	······	Building
L201670P132020	53.22283	6.358669	······	0	-0.0003	-0.0001	······	Building
L199940P153600	53.2165	6.495419	······	-0.0009	0.001	-0.0006	······	Building
L194910P160040	53.15914	6.553742	······	-0.0012	-0.0077	-0.001	······	Building
L194920P159850	53.15914	6.552628	······	-0.0005	-0.0028	-0.0023	······	Building
L205290P219020	53.325341	6.858407	······	0	-0.0013	0.0012	······	Barrier post
L201820P120470	53.216547	6.287831	······	0.0006	0.0005	0.0014	······	lamppost

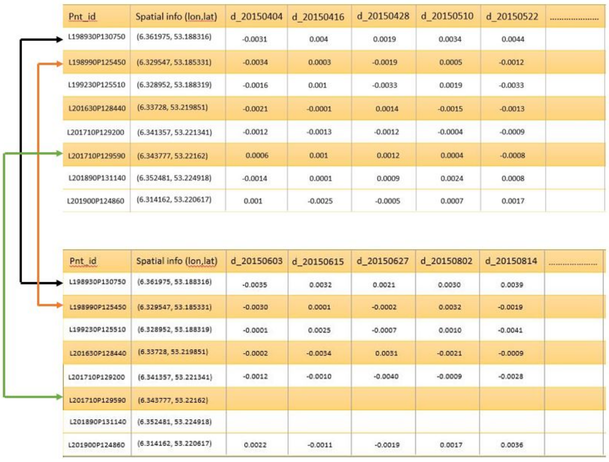
Figure 1. An example of retrieving building and lamppost points. In the figure, the point with ID ”L205290P219020” stands for a barrier post, which is selected because it represents a point-like object as a ”post” as well. Thus, one can select this point as a reference point for registration.

2.4 Update of a Dataset

When new acquisition comes, the dataset is required to be updated. As the satellite(like sentinel-1) has new flights, two types of updates for the data points need to be done. One type is that when new points of the objects on the ground are collected, the data operator should update them into the original data table. This type of update is a vertical update, because it increases the number of rows of a table. The other type is that some existing points get new acquisitions on time series. This type is a horizontal update, because the number of columns of the dataset is increased.

Both types of updates are able to be done in PostgreSQL. The first type of update can be achieved by inserting the new acquisition to the existing table. This method is the same as inserting existing data points into a data table. The second type of update can be done by the ”join” method, namely, join the new time series to existing time series (figure 2 (a)). In the figure, two data tables with same points names but different time series are given. The join method merges not only the time series, but every useful column. Also, this method has no problem on combining rows with empty or non-empty time series. This is practical because not all the points in the existing table have new acquisitions. To connect to PostgreSQL in Python, basic information for PSQL like username, password, host, port and database name are needed.

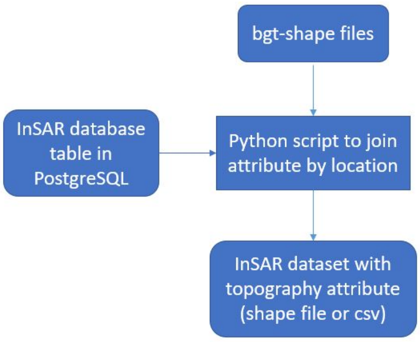
Besides updating new acquisitions, other data layers such as weather data, environment status and feature category values can also be added to the original dataset by the join function. This function is significant as in many cases: the points are collected by a satellite under different weather conditions, like rain, snow and so on. Thus, the returned dataset is enriched, which might outbalance the dataset that without doing the update.



Pnt_id	Spatial info (lon,lat)	d_20150404	d_20150416	d_20150428	d_20150510	d_20150522	…………………
L198930P130750	(6.361975, 53.188316)	-0.0031	0.004	0.0019	0.0034	0.0044	
L198990P125450	(6.329547, 53.185331)	-0.0034	0.0003	-0.0019	0.0005	-0.0012	
L199230P125510	(6.328952, 53.188319)	-0.0016	0.001	-0.0033	0.0019	-0.0033	
L201630P128440	(6.33728, 53.219851)	-0.0021	-0.0001	0.0014	-0.0015	-0.0013	
L201710P129200	(6.341357, 53.221341)	-0.0012	-0.0013	-0.0012	-0.0004	-0.0009	
L201710P129590	(6.343777, 53.22162)	0.0006	0.001	0.0012	0.0004	-0.0008	
L201890P131140	(6.352481, 53.224918)	-0.0014	0.0001	0.0009	0.0024	0.0008	
L201900P124860	(6.314162, 53.220617)	0.001	-0.0025	-0.0005	0.0007	0.0017	

Pnt_id	Spatial info (lon,lat)	d_20150603	d_20150615	d_20150627	d_20150802	d_20150814	…………………
L198930P130750	(6.361975, 53.188316)	-0.0035	0.0032	0.0021	0.0030	0.0039	
L198990P125450	(6.329547, 53.185331)	-0.0030	0.0001	-0.0002	0.0032	-0.0019	
L199230P125510	(6.328952, 53.188319)	-0.0001	0.0025	-0.0007	0.0010	-0.0041	
L201630P128440	(6.33728, 53.219851)	-0.0002	-0.0034	0.0031	-0.0021	-0.0009	
L201710P129200	(6.341357, 53.221341)	-0.0012	-0.0010	-0.0040	-0.0009	-0.0028	
L201710P129590	(6.343777, 53.22162)						
L201890P131140	(6.352481, 53.224918)						
L201900P124860	(6.314162, 53.220617)	0.0022	-0.0011	-0.0019	0.0017	0.0036	

(a)



(b)

Figure 2. (a) Update in time series of data in PostgreSQL. By joining two tables, one big table containing the information from both tables is generated. (b) Update dataset by joining feature category attributes.

2.5 Modification of a Dataset

There are many new events happening everyday, thus, new contextual information is likely to be added to the table. As mentioned in the introduction, if people dig a crater on the road for some reasons, and an InSAR point is just located in the scope of this crater, the height(or deformation) of this point is changed. Thus, it is good if there is some contextual value added to this point, so that the right analysis can be made other than seeing this height as an outlier.

In practice, modifying a dataset like this is very important, because an estimate is made often based on the deformation of the related points. However, if the deformation is mis-analysed, a wrong decision is made. For example, one may consider a building as dangerous as the deformation changes suddenly from a normal value to a high value, but this can actually result from people reconstructing the wall of the building.

Whenever a table is updated or not, it can always easily be stored as a table by a database management system like PostgreSQL. The table can be a csv file or a "file" type file. Also, the table can be shared by other users, either by receiving the output csv files or by visiting the host's dataset via PostgreSQL. With the help of Python, one is allowed to do almost every manipulation mentioned above quickly. Thus, they can be done in a convenient way. In the next section, the results obtained utilizing the methods above are displaced.

3. RESULTS

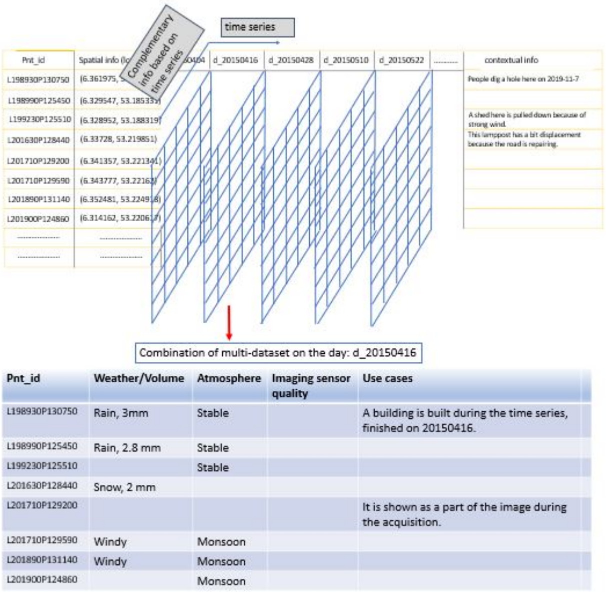
In this section, the results are showed in 3 subsections. We start with how data are structured, and then followed by the results of update and modification of a dataset. In the third subsection, an application on Limburg of the Netherlands is given.

3.1 Data Structure

The information is stored in a database with a certain format which is called a "structure", and the stored information is known as structured data. The data structure can be combined with other data sources, like the weather and atmosphere data that can be obtained from the weather bureau. The imaging sensor quality can be obtained from e.g. ESA. Also, the use cases can be added by people who know what happens at that point. These different sources can be formed as a 3D format table–for each point, any data can be added in another dimension on any date. Thus, whatever information/values corresponding to this point are able to be extracted from this structure. The data structure is showed in figure 3:

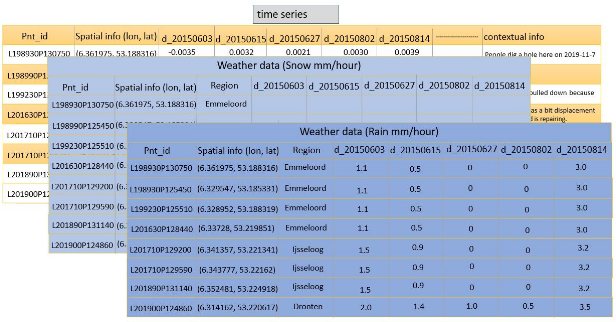
Figure 3 shows for example the data from other sources. A point is regarded as for example a rainy point by judging if this point is located inside the rainy region provided by the weather bureau. The same principle can be applied to snowy points which are specified for the snowy case. In this dimension, many data sources are allowed to be connected, and each column stands for a certain data source.

Another benefit of this data structure is that it is also able to apply the data which is recorded in the direction of the time series (figure 4). Compared with the structure in figure 3, this structure allows the accessorial data to be added to time series as well, thus to effectively study the trend at one certain point. As shown in the figure, the "Pnt_id" and "Spatial info (lon, lat)" attributes are appended to the weather data by evaluating if the points are in the polygon region. As long as the "Pnt_id" attibute exists in both datasets, the values in both datasets can be extracted and joined together.



Pnt_id	Weather/Volume	Atmosphere	Imaging sensor quality	Use cases
L198930P130750	Rain, 3mm	Stable		A building is built during the time series, finished on 20150416.
L198990P125450	Rain, 2.8 mm	Stable		
L199230P125510		Stable		
L201630P128440	Snow, 2 mm			
L201710P129200				It is shown as a part of the image during the acquisition.
L201710P129590	Windy	Monsoon		
L201890P131140	Windy	Monsoon		
L201900P124860		Monsoon		

Figure 3. 3D format data structure. Time series dimension is from InSAR data, complementary information is added based on time series in a perpendicular dimension. Different time series are also connected.



Weather data (Rain mm/hour)

Pnt_id	Spatial info (lon, lat)	Region	d_20150603	d_20150615	d_20150627	d_20150802	d_20150814
L198930P130750	(6.361975, 53.188316)	Emmeloord	1.1	0.5	0	0	3.0
L198930P125450	(6.329547, 53.185331)	Emmeloord	1.1	0.5	0	0	3.0
L199230P125510	(6.328952, 53.188319)	Emmeloord	1.1	0.5	0	0	3.0
L201630P128440	(6.33728, 53.219851)	Emmeloord	1.1	0.5	0	0	3.0
L201710P129200	(6.341357, 53.221341)	Ijsseloog	1.5	0.9	0	0	3.2
L201710P129590	(6.343777, 53.22162)	Ijsseloog	1.5	0.9	0	0	3.2
L201890P131140	(6.352481, 53.224918)	Ijsseloog	1.5	0.9	0	0	3.2
L201900P124860	(6.314162, 53.220617)	Dronten	2.0	1.4	1.0	0.5	3.5

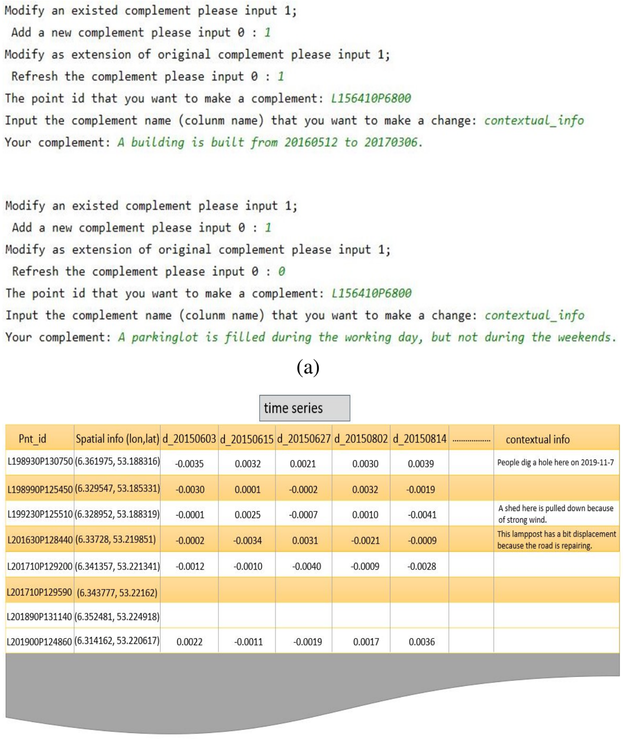
Figure 4. Data recorded in time series direction can be joined to the InSAR dataset by common point ID. Based on this principle, multiple datasets from different sources are allowed to be combined. Note that they are virtual values.

3.2 Update and Modify a Dataset

By the method mentioned in figure 2 (a), a larger data table with longer time series can be obtained when new acquisition arrive. When one analyses the deformation in this longer time series, in practice, human activities affecting the deformation of a small region where the points located are good to be realized. In order to make a correct analysis on the record of the deformation at those points, there are two options. The first one is to make a statement that in the area the height of points are changed before the new acquisition. Therefore, one knows the values of points in this area are not in a common state. The other one is to update the values after the new acquisition. This manipulation allows the user to modify any attribute of the data from the time series. Moreover, contextual information can be added to the dataset by appending the attribute columns.

All the steps mentioned above can be done in the way mentioned in this study using Python and PostgreSQL.

The designed Python script allows one to modify or to add any numeric and textual values to the data, and then be stored. Therefore, it is convenient for a user to maintain the dataset to keep the values as accurate as possible. Figure 5 demonstrates a simple example of updating a contextual value. Since it is probable that one would like to update all the points in a region or all the points on one certain geo-object, the values of the points of interest are able to be updated easily once the region or the type of geo-object is given.



```
Modify an existed complement please input 1;
 Add a new complement please input 0 : 1
Modify as extension of original complement please input 1;
 Refresh the complement please input 0 : 1
The point id that you want to make a complement: L156410P6800
Input the complement name (colunm name) that you want to make a change: contextual_info
Your complement: A building is built from 20160512 to 20170306.


Modify an existed complement please input 1;
 Add a new complement please input 0 : 1
Modify as extension of original complement please input 1;
 Refresh the complement please input 0 : 0
The point id that you want to make a complement: L156410P6800
Input the complement name (colunm name) that you want to make a change: contextual_info
Your complement: A parkinglot is filled during the working day, but not during the weekends.
```

(a)

time series

Pnt_id	Spatial info (lon,lat)	d_20150603	d_20150615	d_20150627	d_20150802	d_20150814	....	contextual info
L198930P130750	(6.361975, 53.188316)	-0.0035	0.0032	0.0021	0.0030	0.0039		People dig a hole here on 2019-11-7
L198990P125450	(6.329547, 53.185331)	-0.0030	0.0001	-0.0002	0.0032	-0.0019		
L199230P125510	(6.328952, 53.188319)	-0.0001	0.0025	-0.0007	0.0010	-0.0041		A shed here is pulled down because of strong wind.
L201630P128440	(6.33728, 53.219851)	-0.0002	-0.0034	0.0031	-0.0021	-0.0009		This lamppost has a bit displacement because the road is repairing.
L201710P129200	(6.341357, 53.221341)	-0.0012	-0.0010	-0.0040	-0.0009	-0.0028		
L201710P129590	(6.343777, 53.22162)							
L201890P131140	(6.352481, 53.224918)							
L201900P124860	(6.314162, 53.220617)	0.0022	-0.0011	-0.0019	0.0017	0.0036		

(b)

Figure 5. Two examples of adding or modifying contextual value for a point via a Python script.(a) shows the way to update information in the Python interface. To do this, one is required to know either the ID of that point or the region where the point located so that all the points in that region can be updated. (b) displays the output of adding contextual values.

The method described in this study allows one to maintain a dataset in basically anyway desired. The points located on the roads for example can be selected and modified once the statement is given by the user. Even the computation and the plotting of the time series can be done this way. It makes sense in practice that one finds an abnormal value of a point during a period, but it turns out to be because of the heavy rain on soil type ground, which makes the height of that point decrease since the soil flowed away.

3.3 Application on the Region of Limburg, the Netherlands

One application of the method in this study is to combine the risky regions of surface subsidence with the InSAR points and to find the feature categories of these InSAR points located in South-Limburg in the Netherlands. The surface subsidence in this area is considered to be a result of the mining systems(see figure 6) which have been undertaken for more than 500 years at depths ranging from 0 to more than 800 m (Bekendam , Pottgens, 1995). In South-Limburg only reserves that have a safe distance to the old mining works can be considered for future exploitation, in order to avoid uncontrolled water flooding and surface subsidence. The majority of remaining coal deposits are located north of the former mines Maurits and Emma/Hendrik(Civil Engineering & Geosciences, n.d.).

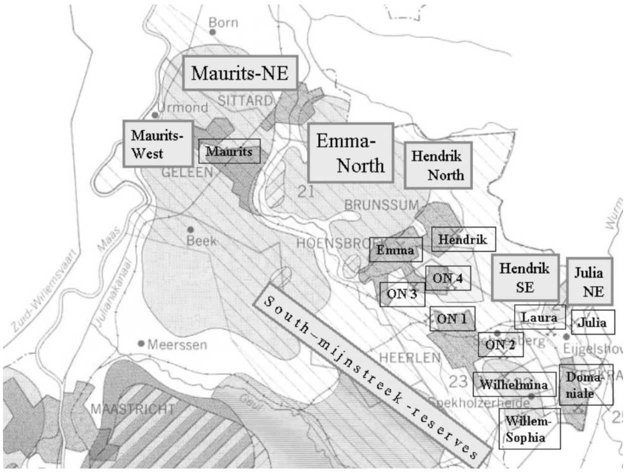


Figure 6. Location of the remaining South-Limburg reserves (filled blocks). The former mines are indicated in the open blocks.(The picture is from Delft University of Technology website).

Using Basic Registration Large-scale Topography (BGT) layers from PDOK (The platform for high-quality geodata) and an InSAR dataset as input, a spatial-temporal data table with feature category attribute can be produced by joining the feature values to the data table. Once we have the feature type of the points, we can have some knowledge on the plot of the time series reflecting the deformation. If there is something wrong on a type of feature, then one can analyse correctly the trend of the surface at this point. Then combining the method to update and to modify the data, the plot might be corrected, e.g. if there is an outlier or not, or the changes at that point due to human activities or the effect from the weather. Table 1 shows how many points are recorded with a type of feature. The result in this study shows that house-type features have most InSAR points which are indicated by different risk levels.

Table 1. Distribution of part of InSAR points from Sentinel 1 track-139 in period of 20150119-20181229 in South-Limburg. The columns are the feature types, the corresponding values indicate the number of points belonging to that type of feature.

risk level	house	bridge	fence/wall	traffic area
1	1	2	0	0
2	5	1	5	7
3	136	66	104	75

The points which experienced large deformations can be retrieved by a querying method, and then be updated if there is a requirement. By collecting the points sharing a similar pattern of deformation and combining them with corresponding feature category it is possible to give one a hint to get to know if there is some relation between surface subsidence and feature types.

To study what kind of places have a chance to experience the surface subsidence using InSAR data, it is important to know how the points from InSAR distribute geometrically in mining system zones. Figure 7 displays how the InSAR points (Sentinel-1 track-139 from ascending direction) with different risk levels are distributed in the Street Map of South-Limburg. It more or less agrees with the distribution of mining systems. The information of the mining zones can be assigned to InSAR points. By doing this, one can link the InSAR points with risk level areas and mining zones. The histogram in figure 8 depicts how many InSAR points are captured in the polygons that stands for different risk levels. All 3 risky-level regions have empty polygons, meaning that no InSAR points for studying the surface deformation are obtained in these small areas. Thus, one can somehow omit those regions as they do not have persistent scatters that may represent important objects such as houses and buildings. From this result, one can query the polygons where there are persistent scattering points located in the polygons to analyse how these points vary in time series. More attention should be paid to the polygons with more persistent scatters as more chance can be seen to appear surface subsidence in these areas, and more properties where people live in .

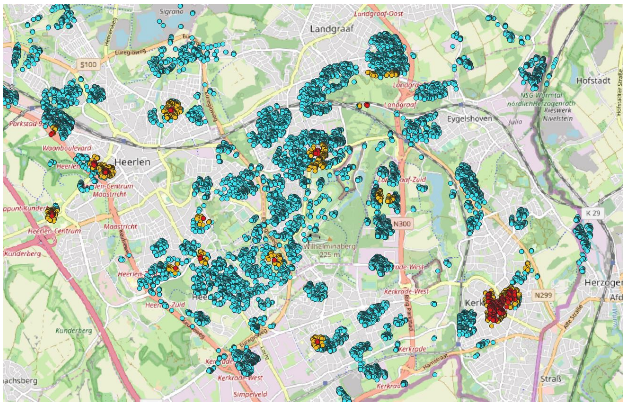


Figure 7. The distribution of part of InSAR points with different risk levels in South-Limburg. Red: points with a risk level 1 for sinkholes. Yellow: points with a risk level 2 for sinkholes. Blue: points with a risk level 3 for sinkholes. Risk level decreases as the level number increases.

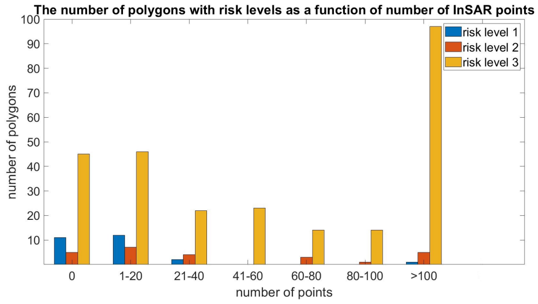


Figure 8. The distribution of InSAR points in different polygons that are classified as 3 risk levels.

4. CONCLUSION AND DISCUSSION

The described way to manipulate the InSAR dataset shows a good ability that allows people to query, update and modify a dataset, especially the function of joining attributes by location and combining multiple data sources. It enriches the InSAR dataset and allows people to correct any apparently unreasonable data values. Also, compared with the method of joining attributes that needs to be done manually in QGIS, this way makes it semi-automatic to obtain the feature types of the points using Python, thus, it is convenient for a user without doing it manually. Different sources of data are usually useful when analysing the variations of the deformation. The structure mentioned in this study is suitable to integrate these sources together, so that any illogical data can be singled out to achieve a reasonable result. In practice, many factors affect the data record both artificially and naturally. When these bad records are recognized, they should be re-recorded as right values. The method in this study make this job easier, either expanding the dataset or revamping the existing one.

The method described in this study is semi-automatic, meaning that adding the csv file of the dataset to a database management system and transforming the topography files to shape files manually. The method is applicable to other regimes as long as the dataset can be opened in PostgreSQL, and this is realizable for most datasets. Not only querying, joining attributes and updating a dataset, but many other functions like calculation and expression of data values in an intelligent way can be executed. However, there are limitations in this method. One is that not every point in the dataset has a type of feature as seen in the South-Limburg application, i.e., due to the imperfection of topography layer files such that some points may be located at the areas where the feature is not collected. Another limitation is that the run-time of Python script in this method increases as the size of the InSAR dataset grows. For a very large sentinel-1 dataset, joining feature to it will run for a long time if there are too many polygons in the topography files. The limitations can thus be the direction of improvement.

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to Marc Bruna, who helped a lot and gave good directions at the initial state of the work. And thanks Freek van Leijen for providing the InSAR data.

REFERENCES

Bekendam, RF, Pottgens, JJ, 1995. Ground movements over the coal mines of southern Limburg, The Netherlands, and their relation to rising mine waters. *IAHS Publications-Series of Proceedings and Reports-Intern Assoc Hydrological Sciences*, 234, 3–12.

Civil Engineering & Geosciences, Delft University of Technology, n.d. Remaining reserves: South-limburg.

Elmasri, Ramez, Navathe, Sham, 2017. *Fundamentals of database systems*. Pearson.

Lohman, Rowena B, Simons, Mark, 2005. Some thoughts on the use of InSAR data to constrain models of surface deformation: Noise structure and data downsampling. *Geochemistry, Geophysics, Geosystems*, 6.

Sammon, John W, 1969. A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 100, 401–409.

Wikipedia, 2020. Postgresql — Wikipedia, the free encyclopedia. [Online; accessed 14-January-2020].