TOWARDS POINT CLOUD HARMONIZATION

Neeraj Sirdeshmukh, Lydia Kotoula, Andria Christodoulou,
Manuela Manolova & Laurens Oostwegel

TU Delft

FUGRO

GEOloc

ABSTRACT

The Geomatics Synthesis Project (GSP) is a project that combines scientific research with practical work by applying essential skills acquired in the Master's programme **Geomatics for the Built Environment** at the Delft University of Technology. The topics of GSP 2017 are Point Clouds (PC) and Internet of Things (IoT).

This report is prepared by the GEOloc team as part of the PC project, which focuses on the the accessibility, interoperability, and quality of different point cloud datasets. The main objective of this project is to research the process of harmonizing point cloud datasets of different origins and to incorporate the integrated dataset in a web viewer that allows for visualization and analysis. As a case study, the Three-Country Point (TCP) where Germany, Belgium, and the Netherlands share a border will be evaluated.

The project is split into several stages. The first step is a theoretical research, which aims to give the team members more knowledge and ideas about the main topics in the project. Then, the key requirements from the client are specified and a project plan is developed as a general overview of the main objectives. In the next step, the data are collected from the three countries bordering the TCP and the acquired point cloud datasets are processed. The main challenge in the GEOloc project will be the determination of an ideal common coordinate reference system (CRS) for the datasets and the transformation of the coordinates into the new CRS. In the process, spatial differences between the datasets originating from various sources are analyzed. Consequently, the harmonized datasets are stored using a spatial clustering method and visualized in a web viewer that allows for analysis.

READING GUIDE

This report is organized as follows. The first part contains an executive summary in the form of a scientific paper which reviews the data collection and processing, the CRS transformation, and the development of a web viewer. The second part gives a detailed overview of the project's work flow as a final technical report. The first chapter encompasses the problem definition, specification of the requirements, and risk management. The methodology is captured in the second chapter with a special focus on defining a common CRS and harmonizing the point cloud datasets. Consequently, the results are shown and the data differences are discussed. In the fourth chapter the developed point cloud viewer is presented along with its functionalities. The report ends with a series of conclusions and further recommendations.

Harmonization of point cloud datasets with varying coordinate reference systems*

Neeraj Sirdeshmukh, Lydia Kotoula, Andria Christodoulou, Manuela Manolova and Laurens Oostwegel
Delft University of Technology, 2628 BL Delft, The Netherlands

***Abstract*— This paper has its focus on the accessibility, interoperability and quality of different point cloud datasets. The main objective of the project is to research the process of harmonizing point cloud datasets of different origins and to incorporate the integrated datasets in a web viewer that allows for visualization and analysis. As a case study, the Three-Country Point (TCP) where Germany, Belgium, and the Netherlands share a border is evaluated.**

keywords— Point Cloud Harmonization, Coordinate Reference Systems, Digital Global Grid Systems

I. INTRODUCTION

Point cloud datasets, usually acquired using Light Detection and Ranging (LIDAR) techniques, are becoming increasingly popular nowadays. Different countries have acquired their own national point cloud datasets and many have made them available as open data. Although a significant amount of research has been done on point cloud data management solutions, the research on the subject of harmonizing point cloud datasets of different countries and making them interoperable is very limited.

This research seeks to address the problem of combining different datasets that connect and overlap at a common geographic area into a unified point cloud dataset. As a use case, the Three-Country Point (TCP) where Germany, Belgium, and the Netherlands share a border is evaluated. Consequently, the integrated point cloud data will be incorporated in a web viewer which allows for visualization and/or analysis. Using appropriate methods, the following research question will be answered:

How can differences between point cloud datasets of various origins that connect and overlap at a common geographic area be harmonized to allow for data interoperability between these varying datasets?

The harmonization of point cloud data from different countries implies:

a) The development of a method that allows for integration, easier and quicker access than the existing solutions, processing and visualization of massive point clouds from different sources.

b) The analysis of the point clouds in the overlapping areas of the three countries that aims to explain the possible differences - and if feasible - to harmonize (minimize) them.

*This work was supported by TU Delft and Fugro

Little is written about point cloud harmonization, regarding different coordinate reference systems. However, more is written about the subjects separately. A good overview of different reference systems and coordinate transformations is made by Marel (2016). Ultimately, the system that is used is a Discrete Global Grid System (DGGS), which will be explained furthermore later on. On of the first researches is made by Dutton (1996). Amiri et al. (2015) compared multiple ways to design a DGGS. There are some point cloud viewers on-line, among which the AHN2-viewer. However, this dataset is in the RD-system, since this is the coordinate reference system the data is acquired in.

II. METHODOLOGY

A. Pre-processing

The data have been collected from the three countries (Netherlands, Germany and Belgium) that our project is focused. While trying to collect point cloud data many differences and problems were faced. For example, Netherlands provides its data in LAZ format, while the German dataset is provided in XYZ File format and the Belgian as a GeoTIFF raster image. All the countries provide their point cloud data as Digital Surface Model(DSM) and Digital Terrain Model (DTM). For the scope of this project the DTM is chosen for the reason that DSM does not include unnecessary objects like vegetation and buildings so, DTM is more suitable for the point cloud harmonization. In the case study area each county uses different coordinate reference systems (CRS) and also the accuracy of the data is different as the way that have been collected is different. The accuracy and the CRS systems from each country is shown in detail in Figure 1 below.

	Netherlands	Germany	Belgium
Data Accuracy			
Position Accuracy (m)	<0.5	0.5	1
Height Accuracy (m	0.1	0.2	0.12
Point Density (pt/m2)	6-10	1-4	1
Format	LAZ	XYZ	GeoTIFF
Coordinate Reference Systems			
Compound CRS	EPSG:7415	EPSG:5555	EPSG: 6190
Horizontal CRS	EPSG:28992 (Amersfoort/RD New) Datum: Amersfoort	EPSG:25832 (ETRS89/UTM Zone 32) Datum: ETRS89	EPSG:31370 (Belgian Lambert 72) Datum: Reseau National Belge 1972
Vertical	EPSG:5709 NAP Height ***Datum:*** Normaal Amsterdams Peil (NAP)	EPSG:5783 DHHN92 Height ***Datum:*** Deutsches Haupthöhennetz 1992	EPSG:5710 Ostend Height ***Datum:*** Ostend

Fig. 1: Data Accuracy and Coordinate Reference Systems.

The presence of different coordinate systems and datums means that many transformations must be done in order to have our data in a common CRS. There are several geodetic

softwares that are able to do the transformations between all the kind of datums (Vanicek et al., 2012).

In order to be able to work with the point cloud data of the three countries a procedure of pre-processing filter is required. Our study area and the boundaries that our project is focused is shown at the Figure 2 were our data were clipped according to this area. The main tools that have been chosen in order to filter our data are the LAStools that could give a wide range of tools that can handle LAS/LAZ files. According the filtering on Netherlands datasets our study area is covering from the two tiles from AHN2 that they can be acquired free from the Dutch PDOK geoportal (PDOK, 2012).

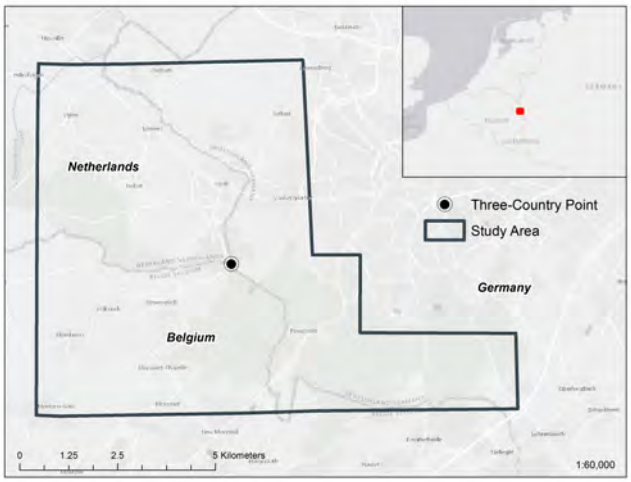


Fig. 2: The boundary of the study area of the research, containing the TCP.

Our study area includes the North-Rhine Westphalia (NRW) German province and only the city Aachen is on the borders with Belgium and Netherlands. The Lidar Data are available for free due to some recent legislation. The German data require more pre-processing procedure and the files have been provided as it is mentioned before in ASCII XYZ format NRW (2017). There are many files for the same region because of the classification, for example there are different files for the different classes (Buildings, Bridge, Water). These files must be merged into one common file for each tile (Isenburg, 2017). For our study area only 150 were useful and inside the area of the Aachen city. Beyond the different types of classification that is mentioned there is also classification about the return types of the points and some synthetic points. The synthetic points are created by the land survey to provide an accurate surface description in areas with low density of the point cloud. Because our project is working with the DSM only the last return points are taking into account (Isenburg, 2017).

In contrast to Netherlands and German data, the Belgium region of Wallonia provide their data in GeoTiff. the data were not available for free and have to be requested after a time consuming procedure. A procedure is followed in order to convert the data from GeoTiff to LAZ format. First the cells with no elevation data have been assigned as No Data and then with Geospatial Data Abstraction GDAL library transform the raster files to vector format. Then using lastools a conversion from the XYZ files to Las files is done.

B. Processing

The goal of this research is to find a global solution in combining datasets from different origins and with different CRSs. Using a standard map projection to achieve this, will give high distortions, which are different at the equator and the poles. On top of that, it will display the surface of the Earth as flat, while it in fact is curved. There are mainly two ways to describe a position that is related to Earth, apart from these projections: the Cartesian Coordinate System or the Geodetic Coordinate System, also referred to as Curvilinear Coordinate System. The first one is mainly used by satellites to measure locations on the Earth (Featherstone and Vanicek, 1998). It has some disadvantages in terms of storing a point that is on the Earth's surface, which takes at least 7 digits for each coordinate if stored in meter-resolution. The second system is more widely used, it represents the surface of the earth through an ellipsoid using ϕ as the geocentric latitude, λ as the longitude, and h as the height above the ellipsoid (Marel, 2016). It is difficult to represent ϕ and λ coordinates in a digital viewer, since these viewers are mostly based on a Cartesian system. Therefore the ellipsoid has to be approximated by planar surfaces.

A Discrete Global Grid System (DGGS) is a spatial reference that uses a hierarchical tessellation of faces to discretized the earth (Amiri et al., 2015). Therefore, a Geodesic DGGS is considered as a sequence of discrete grids that are consisted of grids of finer resolution (Sahr et al., 2003). It can be constructed by defining 4 parameters. Firstly, a 3D shape with planar faces needs to be defined to approximate the earth. This polyhedron (vertices, edges and faces) needs to have a fixed orientation in relation to the earth's surface (PYXIS, 2006). This polyhedron needs to be subdivided into smaller planar polygons, that make up a (multi-resolution) discrete grid. Lastly, a method to transform the planar faces to coordinates on the earth and a way to assign points to each cell is needed.

In this case study, the icosahedron is chosen to be the best polyhedron. It has less distortions than the other polyhedrons, because the size of the faces of a icosahedron is lower. The only exception is a dodecahedron, that is subdivided into 5 isosceles triangles, but there is no possible refinement to get a greater resolution in this dodecahedron (Amiri et al., 2015). The orientation that serves the use case best is the projection Sahr et al. (2003) describes, which has all but one vertex in sea. It is chosen above the orientation of Fuller, which has all its vertices in sea, because it has a symmetry axis on the equator. As a further subdivision, hexagons are most suitable as cells. These quantize the plane with the smallest error, provide the greatest angular resolution and have uniform adjacency: a hexagon is on the same distance with all its 6 neighbours and shares an edge with each.

The disadvantage is that a hexagon is not congruent in subdivision, and therefore it is troulbesome to use multi-resolution, although with further processing this is possible. OGC described that the inverse projection of the hexagons must have the property that it preserves area. The Snyder Equal Area projection serves the DGGS best as inverse map projection, on top of the Equidistant Dymaxion projection of Fuller. The result can be seen in Figure 3.

This ideal DGGS is constructed using DGGrid, a software that has the capabilities to create the icosahedral grid with the planar hexagonal topology and inversely project those (hexagons) on the WGS84 datum. Since the case study is a relatively small point cloud and since it is not in the scope of the project to have a multi-resolution grid, only one resolution was created. It happens to be resolution 14, because this contains around 10.000 to 50.000 points per hexagon. The output is a file with all hexagon centroids and their position in WGS84 latitude/longitude coordinates and a file with the boundary of the hexagons.

C. CRS Transformation

The DGGS has as input WGS84 data. Since WGS84 is a standard datum that is applicable for the entire Earth, this is the preferred system in which the point clouds can be harmonized. The ETRS89 system is more applicable at the specific study area, but the aim of this project is to find a global solution. The need for a centimeter positional accuracy is low, since the data itself is only of maximum 50 centimeter (positional) accuracy. Therefore, as it defines a standard CRS that is applicable to the entire Earth, WGS84 is the preferred system.

A number of softwares is considered to use for the conversion of data to this common system. Out of PCTRANS, FME and PROJ.4, the latter is chosen as the best software for this operation, considering the global usage and the accuracy. It can work directly with LAS/LAZ formats and is implemented as a Python library (pyproj). It supports most of the needed datum transformations without downloading separate grid files and most of the GIS software packages implemented it in their software.

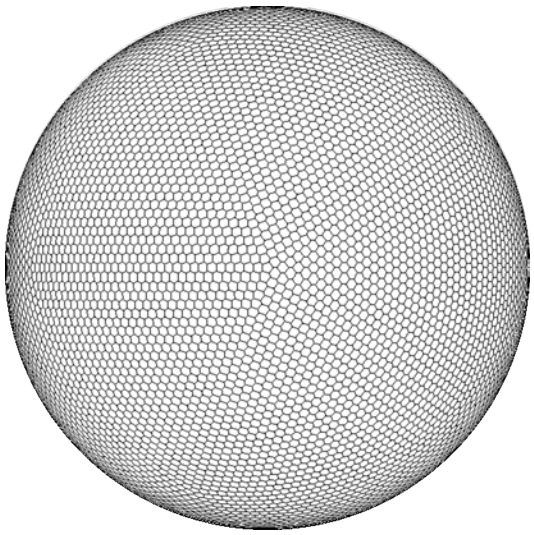


Fig. 3: DGGS that shows one resolution with hexagonal grid cells.

When all points are in WGS84 latitude/longitude coordinates, the point clouds can be clipped with the hexagons. The output is a LAS-file with the country and the hexagon ID in the name. As described in section II-B, visualizing data in WGS84 is rather difficult to do. Therefore the hexagons are used to construct a local coordinate system for each file. The ϕ and λ of the centroid, together with the X,Y,Z ECEF coordinates of the same point are used to construct a transformation to get to this local CRS (see Eq. 4a, Eq. 4b). It will store the point in Cartesian coordinates, with the hexagon centroid as origin and the plane of the hexagon as ground plane. To visualize the point cloud data, the ground plane can be multiplied by the hexagon rotation and the shift of the centroid can be added again.

III. Data Analysis

After having converted the horizontal CRSs of the distinct datasets into geographic coordinates (latitude and longitude)) and their vertical CRSs to ellipsoidal height on the WGS84 ellipsoid, some areas appeared in which two or all three of the datasets overlap. These areas are analyzed for any discrepancies in the X, Y, Z coordinate values.

In order to get a first impression regarding the discrepancies of the data in the 3rd (Z) dimension, the point cloud datasets are converted to raster Digital Elevation Models (DEM's) and then the WGS84 heights are classified in 8 classes, almost all of which are equal interval (see Figure 5). If there were no (or minor) differences in the heights of the overlapping areas between the datasets and if the point clouds from the different countries were perfectly aligned, then there should not be such abrupt changes at the point cloud boundaries. However, this is not true in this figure, e.g. there is a sudden change in elevation value from the range 161-200 meters to the range 201-240 meters at the Dutch-German border.

Now, to compare and contrast the datasets with one another, their overlapping areas need to be delineated. A simple *Minus* operation can be performed to subtract one DEM from another DEM on a cell-by-cell basis. This operation is performed for each of the countries: Netherlands-Germany, Netherlands-Belgium, Germany-Belgium.

Figure 7a shows the German DEM subtracted from the Dutch DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. This area

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \left(\begin{bmatrix} X_{point} \\ Y_{point} \\ Z_{point} \end{bmatrix} - \begin{bmatrix} X_{centroid} \\ Y_{centroid} \\ Z_{centroid} \end{bmatrix} \right)$$

(a) shift

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} * \begin{bmatrix} -\sin\lambda & -\sin\varphi\cos\lambda & \cos\varphi\cos\lambda \\ \cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi\sin\lambda \\ 0 & \cos\varphi & \sin\varphi \end{bmatrix}^{-1}$$

(b) rotation

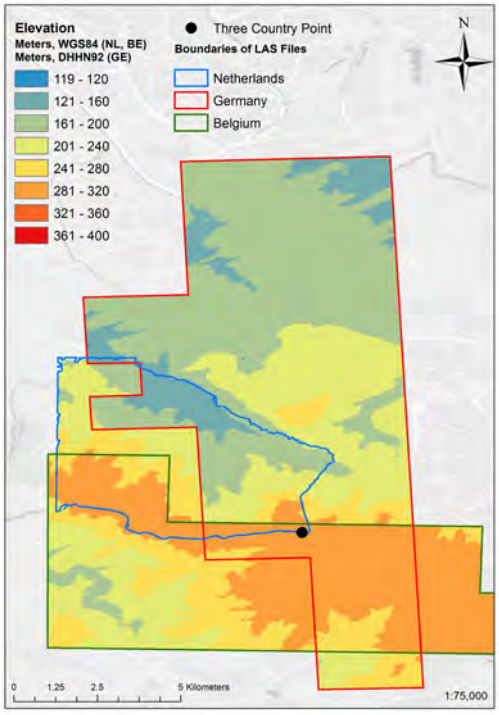
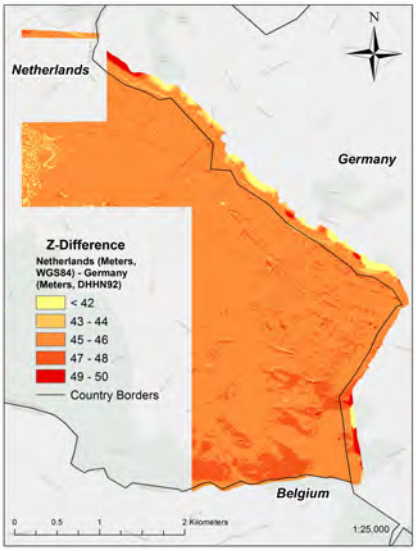
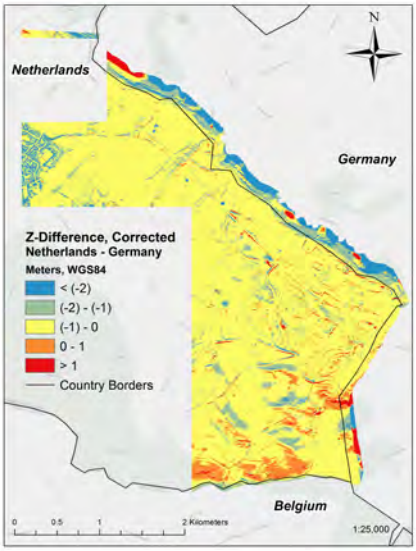


Fig. 5: Input map.



(a)



(b)

Fig. 6: Differences Netherlands and Germany before and after quasi-geoid correction.

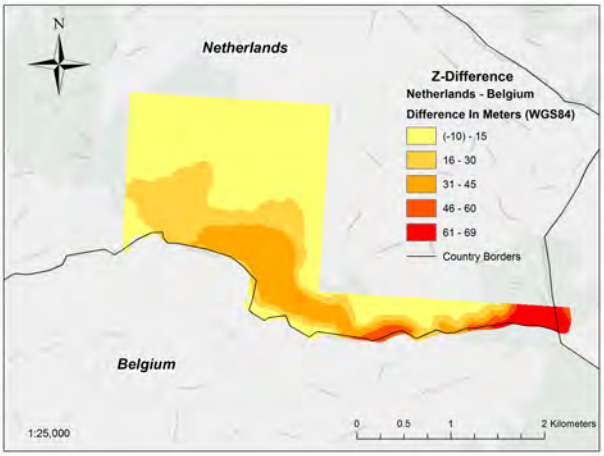
falls almost entirely within the extent of the Netherlands. The difference that is displayed, is almost everywhere approximately 45-46 meters. This is because Proj.4 did not take the conversion of the German vertical CRS into account. This conversion in vertical CRS is related to a concept known as a *quasi-geoid*. The quasi-geoid that is used in the German dataset is the DHHN92, one of the many realizations.

The German Federal Agency for Cartography and Geodesy (FACG) provides a free service to compute quasi-geoidal heights in DHHN2016, a system that differs only a few centimeters in height from DHHN92. Another online service provides the differences between DHHN2016 and DHHN92, so this could be taken into account. Doing this for a whole dataset, is not free of charge. Therefore, a sample set of 40 points at random locations is used. The method cannot be used directly, because the service only provides conversion from geoid hieght to quasi-geoid height and not the other way around, and therefore some iterations need to be made. The result is a set of 40 points that are corrected on the quasi-geoid of Germany. It can be used to interpolate the German dataset and this result is shown in Figure 7b. The majority of the points have an error that is less than one meters, except for some outliers. This is quantified in Figure 12 in the Appendix.

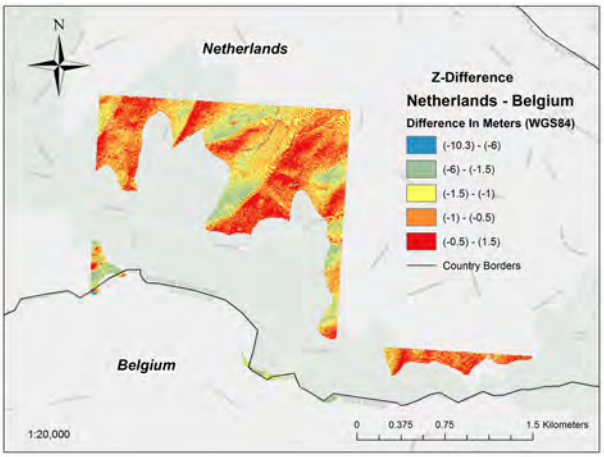
Figure 7 shows the deviation between the Dutch and the Belgium dataset. Unlike the previous study, the error is not systematic and ranges from -10 to +68 meters. Almost all values between -10 and 15 meters actually fall in the range of -3 to +3 meters. The datasets deviate the most from one another in the southern portions of the overlapping area, where differences in the elevation values can reach the highest.

To explain the data difference, firstly the datasets are compared to their benchmarks. The Netherlands provides Normaal Amsterdams Peil (NAP) benchmark points, that have been surveyed with respect to their X, Y, and Z RD-NAP coordinates. These all fall within the Dutch border. A deviation can be found of 0 - 1.75 meters, which is not enough to explain the differences of +68 meters.

The same is done with the Belgian NGI Geodetic Benchmarks and the Belgian dataset. Equivalent to the Dutch data, the benchmarks only exist within the Belgian border. The

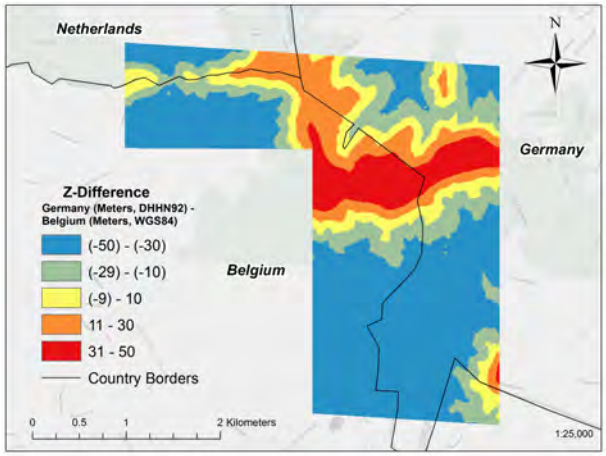


(a)

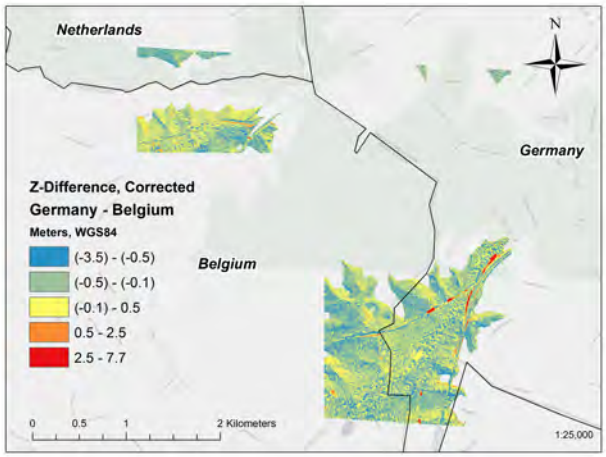


(b)

Fig. 7: Netherlands minus Belgium before and after error removal.



(a)



(b)

Fig. 8: Differences Germany and Belgium before and after quasi-geoid correction and error removal.

deviation that is found, is no more than in between -1 and +1 meters, which means neither of the datasets are wrong within the border of their own country.

As the overlapping area of the two datasets is almost entirely within the territorial extents of the Netherlands, this provides a strong reason to state that the accuracy of the Belgian dataset becomes poorer in this region. This can be illustrated by using a NAP benchmark that is converted to the Belgian datum (Ostend Height). Converting a benchmark with ID 062Do117, which is at (195.550;308.640;275,203) in RD/NAP coordinates, returns 276.62 meters with Proj.4 and 277.56 with PCTRANS. While they are close to each other, they are not close at all to the value contained in the GeoTIFF file of Belgium (255 meters). With some more research it can be found that all values that are not resembling the Dutch dataset at all, are of value 255.

With a first glance at figure 8a, one can see that the datasets of Belgium and Germany do not align at all. About half of the German data is more than 42 meters below the Belgian. The remainder is spread equally between -42 and 42 meters below/above the Belgian data. Using our prior knowledge from the previous comparisons, an assumption can be made about the differences in datasets. The German data needs to be corrected on its quasi-geoid and the Belgian data has some incorrect values, that are detected quite easily because they are of value 255.

Using the same method as described in the part related to the differences between the German and Dutch dataset, a file that has a correction of the German quasi-geoid is constructed. The remaining deviation is between -4 and 90 meters. The areas where the Belgian dataset has values of 255 meters are almost identical to the areas of maximum deviations between the two datasets. Thus it can be con-

cluded that there is a strong belief that this is the cause of the remaining discrepancies between the two datasets.

IV. Data Visualization

The end product for the GEOloc project is to establish an open point cloud web viewer containing harmonized datasets of our three countries. The are several options and tools to use in order to visualize our data (Potree, Cesium and Three.js). Because our point cloud data is very small about 10.000-50000 points per hexagon the most appropriate for our case is Three.js. Three.js creates a simple 3D environment using Javascipt library can also handle many files at once and the processing time is not much (Ltd, 2016). It gives accessibility to many users and it can be supported by many browsers. Three.js is an open source library that can simplifies the WebGL tools and environment (Ltd, 2016). To initialize making a 3D visualization first the scene, the camera must be set. Also some external libraries must be defined to be able to use some functions for making our wed viewer. Our main input are the files with the hexagons and the points inside and also the transformation parameters for each hexagon. The files contain the points in local coordinates and by applying the transformation parameters for each hexagon we transform on the fly from local to ECEF system to be able to visualize the hexagons and display them in their initial place. About the interface it is made user-friendly with some basic information displayed. While clicking on a point on the canvas the coordinates on WGS'84 are displayed to the user transforming again on the fly.This is because The hexagons with the points from the three countries could be seen. Each country is represented in different color (Belgium:Red, Netherlands: Green, Germany: Blue) in order to understand the boundaries and the overlapping areas between the countries are also represented them with the combined colors as it could be seen in the Figure 9 below.

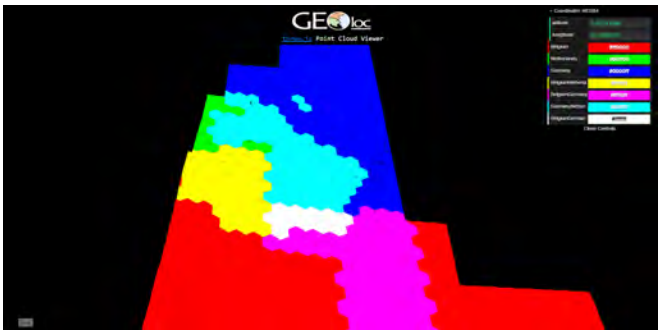


Fig. 9: Final interface of the Web Viewer

V. Conclusions

The general objective of this project is to research the harmonization of different point cloud datasets that connect and overlap at a common geographic area and to visualize the integrated data in a web viewer. The research question specified for the research was answered based on various GIS techniques and an extensive literature study.

During the data collection, several differences in the acquired point cloud data appeared, such as the data accuracy, file formats, and different CRSs. Therefore, the datasets had to be filtered to allow for the CRS transformation in a later step.

The main challenge of the scientific research in this project was the determination of an ideal CRS for different point cloud datasets. The Geodetic Coordinate System was chosen as a common CRS, as it represents the Earth's surface using the simplest rotational shapes that match the true the Earth. However, this CRS is not very suitable for spatial analyses and data visualization in web viewers. In order to store the point cloud data efficiently and to be able to visualize the datasets in a web viewer, a DGGS was used as a spatial reference for dividing the Earth in a multitude of small regions. The DGGS was generated by specifying the design parameters (polyhedron, cell type, polyhedron orientation, and inverse map projection) in the DGGRID software.

After having defined an ideal CRS for the point cloud data, the CRS of the distinct datasets was converted into the new CRS. The point clouds were clipped with the hexagons and these were used to construct a local CRS for each file in order to reduce data storage.

Key part of the research was the analysis of any height discrepancies in the overlapping areas. There are mainly two reasons for the height differences between the three datasets. Germany uses the quasigeoid as vertical CRS that was not taken into account by the software used for the CRS transformation (Proj.4). This was solved by computing the quasigeoid for a sample of 40 points using an online service provided for that. As to Belgium, it can be concluded that the Belgian dataset deteriorates significantly in accuracy in the overlapping area, as this area is almost entirely within the territorial extents of the Netherlands.

As a result of the point cloud harmonization, a web viewer was created to allow for visualization of the integrated data. Three.js was used due to the variety of options it offers for visualizing 3D objects. In order to be able to display the point cloud data, the latitude and longitude and the centroid of the hexagons along with the X,Y,Z ECEF coordinates of the same point were used to construct transformation and rotation matrices to get to the local CRS. Then, the translation and rotation matrix was applied to each hexagon so that it can be shown at the correct place in the viewer.

VI. Further Recommendations

As expected, several essential issues should be considered for advancement in terms of the DGGS and web viewer. In order to make the research applicable in a global scale, the following recommendations for the DGGS generation are provided:

- **Hexagonal grids**. Further research is necessary to find out how the icosahedron grids are actually created in

terms of the equations used, computing limitations, accuracy, etc.

- **Inverse map projection**. ISEA was used, but it has the disadvantage that it produces changes in the directions (Furuti, 2015), which are visible as breaks in the lines connecting vertex to face centers. Therefore, other possible equal area projections need to be found.
- **Indexing**. A hierarchical indexing method for the hexagonal grids should be developed, as a single resolution is not sufficient to constitute a DGGS (Stroble, 2016).

Three.js uses a JavaScript library and offers a wide range of functionalities to visualize 3D objects and process point cloud data. However, there are several recommendations to improve the performance of the web viewer in the future:

- **Interactive**. The web viewer could be made more user-friendly by adding various options like turning on/off layers according to the area or country that the user wants to examine.
- **Analysis tools**. It would also be useful to add some tools on the interface that allow the user to calculate the distance and give some statistics about the input data, such as the quality and the density of the point cloud.
- **Open data**. In order to make the web viewer truly open, download and upload functions could be added so that the users are able to work with the data.

APPENDIX

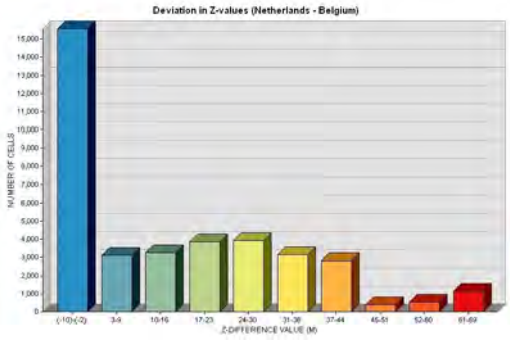


Fig. 10: Histogram Netherlands minus Belgium

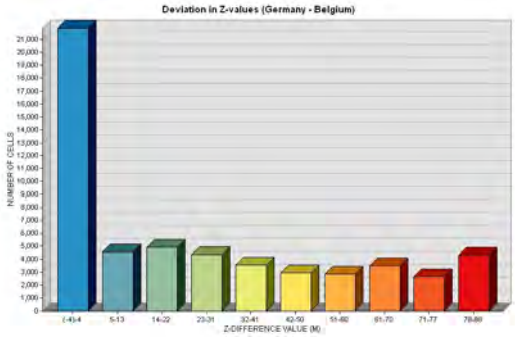


Fig. 11: Histogram Germany minus Belgium

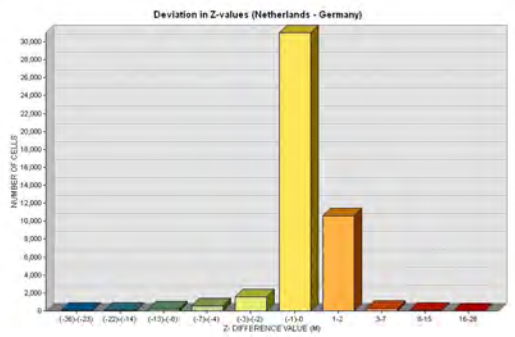


Fig. 12: Histogram Netherlands minus Germany

REFERENCES

Amiri, A. M., Samavati, F., and Peterson, P. (2015). Categorization and conversions for indexing methods of discrete global grid systems. *ISPRS International Journal of Geo-Information*, 4(1):320–336.

Dutton, G. (1996). Encoding and handling geospatial data with hierarchical triangular meshes. In *Proceeding of 7th International symposium on spatial data handling*, volume 43. Netherlands: Talor & Francis.

Featherstone, W. and Vanicek, P. (1998). The Role of Coordinate Systems, Coordinates and Heights in Horizontal Datum Transformations.

Furuti, C. A. (2015). Map Projections: Polyhedral Maps - part 1.

Isenburg, M. (2017). Nrw open lidar: Merging points into proper las files.

Ltd, A. P. (2016). Animating scenes with webgl and three.js.

Marel, v. d. (2016). *Reference Systems for Surveying and Mapping*. Delft University of Technology.

NRW, G. (2017). Digitales Oberflächenmodell mittlerer Punktabstand 1m.

PDOK (2012). Actueel hoogtebestand Nederland (gefilterd) (AHN2).

PYXIS (2006). Discrete Global Grid System.

Sahr, K., White, D., and Kimerling, A. J. (2003). Geodesic discrete global grid systems. *Cartography and Geographic Information Science*, 30(2):121–134.

Stroble, P. (2016). Discrete Global Grid Systems for handling big data from space.

Vanicek, P., Kingdon, R., and Santos, M. (2012). Geoid versus quasigeoid: a case of physics versus geometry. *Contributions to Geophysics and Geodesy*, 42(1):101–118.

CONTENTS

ACRONYMS

AOI Area Of Interest

CRS Coordinate Reference System

DEM Digital Elevation Model

DGG Discrete Global Grid

DGGS Discrete Global Grid System

DSM Digital Surface Model

DTM Digital Terrain Model

ECEF Earth Centered Earth Fixed

GSP Geomatics Synthesis Project

GEOTIFF Georeferenced Tagged Image File Format

GDAL Geospatial Data Abstraction Library

ISEA Icosahedral Snyder Equal Area

KML Keyhole Markup Language

LAS LASer

NRW North-Rhine Westphalia

OGC Open Geospatial Consortium

SGL Second General Levelling

TCP Three-Country Point

UTM Universal Transverse Mercator

WebGL Web Graphics Library

1 INTRODUCTION

This report provides an overview of the GEOloc project as part of the Geomatics Synthesis Project (GSP) 2017. The general scope of the project is the accessibility, interoperability, and quality of different point cloud datasets. The project combines scientific research with practical work. In this way, the students have the opportunity to combine the skills gained in the following courses as part of the Master's programme **Geomatics for the Built Environment**:

- Geographical Information Systems (GIS) and Cartography
- Positioning and Location Awareness
- Sensing Technologies for the Built Environment
- Geo Datasets and Quality
- Geo Database Management Systems
- Python Programming for Geomatics
- Geoweb Technology
- Geo-information Organization and Legislation

The aforementioned courses covered several aspects of the project's scope. The GIS and Cartography course gave a solid basis to the students of how to use GIS in order for solving real-world problems. Moreover, the course encompassed basic concepts of cartography for visualizing spatial data on maps efficiently and explained the different CRSs, projections, and transformations. In addition, the Positioning and Location Awareness course expanded the gained knowledge about the CRSs by giving more practical experience in performing re-projections and transformations. The Sensing Technologies course provided essential theoretical knowledge of data acquisition methods including Light Detection and Ranging (LIDAR) techniques to obtain point cloud data, which is directly connected to the GEOloc project.

The Geo Datasets and Quality course was useful for the project with respect to the assessment of the quality of spatial datasets using several statistical methods (e.g. systematic errors) and their harmonization. One of the topics in the Geo Database Management Systems course was storing point cloud data in an efficient way using spatial indexing and clustering, which was necessary within the scope of the GEOloc project. Python programming skills were applied for the conversion of the coordinates to another system, and for the creation of rotation and translation matrices of the points in the point cloud datasets that were used for the data storage and visualization. The content of the Geoweb Technology course was invaluable, as it covered the basics of front-end programming with HTML, CSS, and JavaScript. This knowledge was used for creating the web viewer.

The Geo-Information Organization and Legislation class was relevant for the project in terms of finding open data from the three countries bordering

the TCP. Furthermore, the web viewer of the integrated point cloud datasets was developed as open data so that everyone can access it without any restrictions.

1.1 PROBLEM DEFINITION

Point cloud datasets, usually acquired using LIDAR techniques, are becoming increasingly popular nowadays. Different countries have acquired their own national point cloud datasets and many have made them available as open data. These datasets are usually immense with respect to size, and need to be stored efficiently. Although a significant amount of research has been done on point cloud data management solutions, the research on the subject of harmonizing point cloud datasets of different countries and making them interoperable is very limited. The datasets originate from different sources and have different properties, such as CRSs and data quality. As more and more point cloud datasets become available, there is an ever growing need to make them accessible to end users through the Internet.

This research seeks to address the problem of combining different datasets that connect and overlap at a common geographic area into a unified point cloud dataset. As a use case, the Three-Country Point (TCP) where Germany, Belgium, and the Netherlands share a border will be evaluated. The challenges in this task are researched and solutions are provided. The end product would be an open point cloud data viewer in which the harmonized datasets of the case study countries are visualized, i.e. the CRS differences have been resolved and the datasets "line up" with each other.

The harmonization of the point cloud data of different origins implies the two big parts on which the project is focused:

a) The development of a method that allows for integration, easier and quicker access than the existing solutions, processing and visualization of massive point clouds from different sources.

b) The analysis of the point clouds in the overlapping areas of the three countries that aims to explain the possible differences - and if feasible - to harmonize (minimize) them.

1.2 RESEARCH OBJECTIVES

The main goal of this project is to research the problems, challenges, and potential solutions involved in the process of harmonizing point cloud datasets of different origins. Consequently, the integrated point cloud data will be incorporated in a web viewer which allows for visualization and/or analysis, something similar to an "Open Street Map for Point Clouds". Using appropriate methods, the following research question will be answered:

How can differences between point cloud datasets of various origins that connect and overlap at a common geographic area be harmonized to allow for data interoperability between these varying datasets?

Each country of the case study had its own point cloud data hosted on its own server. Therefore, it was necessary to analyze how to store the data efficiently and how to incorporate the point cloud datasets in a web viewer to allow for analysis. Hence, the main research question for this project is composed of four sub-questions:

- *How can point cloud datasets from different sources be combined?*
- *What is the ideal CRS for point cloud datasets?*
- *How can these point cloud datasets be stored efficiently?*
- *How can the differences in the overlapping areas be harmonized?*

The project covers a broad research scope, the majority of which is derived from the requirements set forth by the main client - Fugro GeoServices B.V. Fugro is a global leader in offshore survey, offshore geotechnical and seabed (subsea) geophysical services. The end product, the open point cloud viewer, should meet the requirements of the main client regarding its look and functionality. Ultimately, the web viewer will be provided by Fugro to its clients. A flowchart of the project's workflow is shown in Appendix F.

1.3 RELATED WORK

There is very little written about point cloud harmonization, regarding different CRSs. Much more is written about both subjects separately. Budge and Niederhausern (2011) did research about combining point clouds with low-cost GPS/IMU Systems.

There has been much research on the subject of CRSs and coordinate transformations. A good overview of different kind of reference systems and transformations is made by Marel (2016).

Ultimately a Discrete Global Grid System (DGGS) will be used in the project. This will be explained in detail in Section 2.2. The idea of a DGGS is around there of quite some time. One of the first researches is made by Dutton (1996), that wrote about a specific DGGS: O-QTM (an Octahedral Quaternal Triangulated Mesh). A few years later, Amiri et al. (2015) gave a more holistic approach. They compared multiple ways to design a DGGS and explained the advantages and disadvantages of each way. OGC set up some requirements that a DGGS has to take into account (Purrs, 2015, Strobl et al., 2016).

There are already several point cloud web viewers. For instance, the AHN2 viewer shows the whole AHN2 dataset of the Netherlands. However, this dataset is still in the RD-system, since this is the coordinate reference system the data is acquired in.

1.4 REQUIREMENTS ANALYSIS

After having defined the main objectives of the project, it is necessary to determine the conditions and needs that must be met by an end product based on the requirements of the involved stakeholders. The project is carried out

in a team of five students and involves stakeholders from both TU Delft and Fugro.

1.4.1 Key Requirements

Although there are various types of requirements, some important ones are the functional and non-functional requirements and boundary conditions (Mochal, 2007). The functional requirements identify the necessary task, action or activity that must be accomplished for the project to run successfully and the non-functional requirements specify the criteria for assessing the outcome. The boundary statements help to separate the things that are applicable to the project from those outside its scope. One of the simplest methods for identifying boundary conditions is answering the five "W"s along with "How" (Product & Process Innovation, 2017).

- Functional requirements:
  - Accuracy assessment
  - Data collection
  - Data pre-processing
  - CRS determination
  - CRS transformation
  - Selection of a web viewer for visualization

- Non-functional requirements:
  - Datasets harmonization
  - Quality of the integrated dataset
  - Usability of the web viewer and visualization method
  - Setting different colors for countries to distinguish the point clouds
  - Scalable solution, applicable to other case studies

- Boundary conditions:
  - what: web viewer of the harmonized point cloud datasets
  - who: Fugro and its clients as primary customers/stakeholders
  - when: 24/04/2017 - 30/06/2017 (10 weeks)
  - where: mainly at TU Delft; several meetings at Fugro's office
  - why: accessibility to the leading spatial datasets
  - how: scientific research, time, web rendering software knowledge

1.4.2 MoSCoW Rules

The MoSCoW method is a task planning and prioritization technique used in project management to classify the tasks of a project into those that "must", "should", "could", and "won't" be performed. This allows the project performers to reach a common understanding with stakeholders on the delivery of each requirement (Miranda, 2011). The relevant MoSCoW tasks for the scope of the GEOloc project are listed in Table 1.1.

To meet the key requirements of the client, the GEOloc team keeps in contact with the stakeholders and other experts and conducts interactive discussions with the supervisors throughout the research.

Must	Should
Harmonize the data of different point cloud datasets.	Use efficient spatial clustering techniques to store massive point clouds
Determine an ideal common CRS for point cloud visualization on web	Compare the suitability of the CRS transformation software programs.
Incorporate the point cloud dataset in a web viewer for analysis and better understanding of the CRS parameters.	Assess the quality of the input and output data.
Could	**Won't**
Have extra attributes than the X,Y,Z coordinates.	Research versioning and archiving of point cloud data in the database.
Compare and determine an ideal database management system (DBMS).	Research multi-user access and security issues in the database.
Find solutions for filling gaps between different point cloud datasets.	Build a web viewer.
Create switch on/off point cloud datasets functionality based on country.	Classify points that do not have classification.
Create download functionality in the web viewer.	

Table 1.1: MoSCoW Rules.

1.5 RISK MANAGEMENT

Risk assessment is an essential part of each project, as it helps for the identification of risks and their impact on the project. Risk can be defined as any uncertainty that, if it occurs, would have a direct effect on the project objectives and functioning. In that sense, risk includes both threats to the project success and opportunities for improvement (PMI, 2000).

Project risk management is usually associated with the development and evaluation of alternative plans that support plans of the project activities (Chapman and Ward, 2003). Risk management planning consists of risk identification, analysis, and response (PMI, 2000).

First, the risks that occurred in the GEOloc project were identified and divided into technical and organizational risks, which in turn could be influenced by internal and external factors. After having analyzed the uncertainties in the project, specific actions were determined to reduce any threats to the project objectives (see Table 1.2).

The internal risks refer to the tasks that need to be accomplished by the GEOloc team to meet the client's requirements and the organization between the team members. For instance, the team has to find a way to deal with any problems that may occur in the data filtering and/or CRS transformation process. Moreover, the team members need to have a good communication to allow for the project to run smoothly.

The external risks are determined from many factors outside the team, and thus, these are more difficult to control. In case of the GEOloc project, the availability of the point cloud datasets and their quality are external issues that can influence the success of the project directly. Furthermore, the cooperation between academic and professional stakeholders depends mainly on the availability of both bodies in the period when the project is running.

	Risks	Solutions
	Technical	
Internal	Data filtering	Using a DTM instead of a DSM to reduce additional filtering Using the existing classification of the German point cloud dataset
	CRS transformation	Comparing the datasets and researching about ways to reduce the distortions
	Data visualization	Selecting a web viewer that uses a JavaScript library which is familiar to the team members
External	Data availability	Contacting the institutions responsible for the point cloud data in the given country/ province
	Data quality	Conducting own research on the quality of the acquired data Contacting some experts responsible for the point cloud datasets
	Software usage	Selecting software programmes that the team members are familiar with and can
	Organizational	
Internal	Communication	Giving weekly update to the team leader and other team members Working in smaller groups of 2-3 people
	Team meetings	Organizing weekly meetings including more discussion on the work flow
External	Involvement of Fugro in the project	Meeting the specified requirements to their full extent
	Cooperation between academic and professional stakeholders	Arranging meetings with the mentors from the TU Delft and from Fugro

Table 1.2: Risk management of the GEOloc project.

2 METHODOLOGY

2.1 DATA PRE-PROCESSING

2.1.1 Data Collection

Within the scope of the GEOloc project, point cloud data have been collected from the three countries bordering the TCP: the Netherlands, Germany, and Belgium. During the data collection process many differences between the datasets with respect to the file format and CRS have been found. The Netherlands provides its data as LAZ (compressed LAS) files, whereas the German point cloud dataset is an XYZ-file and the Belgian one a GeoTIFF raster.

A common feature of the countries is that they all provide their point cloud data as either a Digital Surface Model (DSM) or a Digital Terrain Model (DTM). A DTM represents the bare-ground, and does not include any above-ground objects such as buildings or vegetation, whereas a DSM includes the above-ground features. Using a DSM could be advantageous, because it provides more detailed information about the real world. Buildings could be used to find certain spatial errors between different countries. However, as the positional accuracy of the Belgian dataset is only 1 m, it is questionable if those spatial errors can really be found. Furthermore, the DSM is unfiltered in all cases, while the DTM is filtered for all three countries. Filtering a DSM to remove outliers (points with excessively high or low elevation values) is a challenging task in itself and is outside the scope of this project. Therefore, a DTM is more suitable for the point cloud harmonization.

The case study countries use different CRSs, which is another challenge in the GEOloc project. An overview of the accuracy and formats of the three datasets as well as the distinct CRSs can be found in Table 2.1.

The CRSs of the point cloud datasets are provided along with their EPSG codes. In the case of all three countries, the datasets used a separate horizontal CRS and a separate vertical CRS. Furthermore, the datums that the datasets are based upon are all different. A datum is a set of reference points from which surveys and measurements can be made (NOAA, 2017). Datums act as starting points for carrying out surveys and measurements. The presence of multiple datums means that datum transformations will need to be performed while reprojecting the datasets into a common CRS. Usually, most GIS or geodetic software programs are able to perform transformations between all kinds of datums and CRSs (Vanicek et al., 2012).

	Netherlands	Germany	Belgium
Data Accuracy			
Position Accuracy (m)	<0.5	0.5	1
Height Accuracy (m	0.1	0.2	0.12
Point Density (pt/m2)	6-10	1-4	1
Format	LAZ	XYZ	GeoTIFF
Coordinate Reference Systems			
Compound CRS	EPSG:7415	EPSG:5555	EPSG: 6190
Horizontal CRS	EPSG:28992 (Amersfoort/RD New) Datum: Amersfoort	EPSG:25832 (ETRS89/UTM Zone 32) Datum: ETRS89	EPSG:31370 (Belgian Lambert 72) Datum: Reseau National Belge 1972
Vertical	EPSG:5709 NAP Height ***Datum:*** Normaal Amsterdams Peil (NAP)	EPSG:5783 DHHN92 Height ***Datum:*** Deutsches Haupthöhennetz 1992	EPSG:5710 Ostend Height ***Datum:*** Ostend

Table 2.1: Data Accuracy and Coordinate Reference Systems

The Netherlands

The Dutch government organisation Rijkswaterstaat created together with 'de Waterschappen' the AHN (*ActueelHoogtebestandNederland*) dataset. The data was primarily used for water management. The first version (AHN) was made between 1996-2003 and had a point cloud with a density of about 1 point per 16 m^2. The second AHN point cloud is more dense and has been available since 2012/2013. The density is between 6 and 10 pt/m^2 (Zon, 2013). There is an even denser point cloud that is being captured, called AHN3. It has classifications, but is not yet available for the TCP area (see Figure 2.1). Therefore, the AHN2 point cloud is used in this study.

Figure 2.1: Data availability of AHN3

As can be seen in Table 2.1, the horizontal accuracy of the Dutch dataset is 0.5 meters or better and the vertical accuracy is at least 0.1 meters. 5 centimeters are systematic and 5 centimeters are stochastic errors. The CRS of the dataset is in EPSG:7415. This is a combined system, with its horizontal coordinates in *Amersfoort / RD New* (EPSG:28992) and elevation in *Normaal Amsterdams Peil a.k.a. NAP* (EPSG:5709) (PDOK, 2012).

Germany

Contrary to the Dutch AHN2 dataset, most of the German point cloud data is not provided as open data. The federation of North Rhine West-

phalia (NRW) however, provides both its DTM and DSM as downloadable ASCII XYZ-files. Those are provided in EPSG:5555, which combines the 2-dimensional UTM coordinate system on the ETRS89 datum with the German height values above sea-level, based upon the NHN (*Normalhöhennull*) system.

According to NRW (2017), the data has an accuracy of 0.5 meters in the horizontal direction and an accuracy of 0.2 meters in the vertical direction. The specification does not provide a division into what are systematic or stochastic errors. The resolution is 1-4 points per meter squared and the DTM has not been post-processed (further explained in Section 2.1.2) .

Belgium

The data of Wallonia has to be requested in order to obtain access to it. It uses EPSG 6190, a compound CRS consisting of the Belgian Lambert 72 horizontal CRS with the Ostend height (EPSG: 5710) as vertical system. The position accuracy is 1 meter, while the height accuracy is 12 centimeters. The point density is the same as that for the German dataset (1 pt/m^2), but it has already been processed and is delivered in GeoTIFF format. Therefore this dataset is in the form of a raster grid file, in contrast to the data of Germany and the Netherlands.

2.1.2 Data Filtering

To be able to work with the point cloud data of the three countries, several pre-processing steps had to be applied to the datasets to prepare them for the coordinate conversion. Firstly, as the particular use-case of this research is the TCP, where the borders of the three countries meet, a small study area containing the TCP and extending into the interiors of the three countries was outlined. Point cloud datasets were acquired for the regions of the countries overlapping the study area, and if their extent fell outside the study area bounds, then the datasets were clipped to the study area. Figure 2.2 below provides an outline of the study area chosen for the research.

The sequence of steps followed to pre-process the point cloud data are described in detail below, separately for each country. For most of the data processing, the *LAStools* suite of tools for LIDAR data processing available from `www.rapidlasso.com` was utilized. LAStools was chosen due to the wide range of tools available for working with point cloud data, low memory requirements, and generally fast speeds.

The Netherlands

Point cloud data in the LAZ (.laz) format was acquired from the Dutch PDOK geoportal for the part of the study area that falls within the Netherlands. LAS (.las) is a public file format allowing for the interchange of 3-dimensional point cloud data between users, and created by the American Society For Photogrammetry and Remote Sensing (ASPRS). A binary file format, it provides a simple yet comprehensive support for the storage of (usually) LIDAR point cloud data in a standardized and exchangeable format. Its lossless compressed twin is the LAZ format. Two tiles from the AHN2 dataset were acquired, numbered "g69gn2" and "g69hn1" according

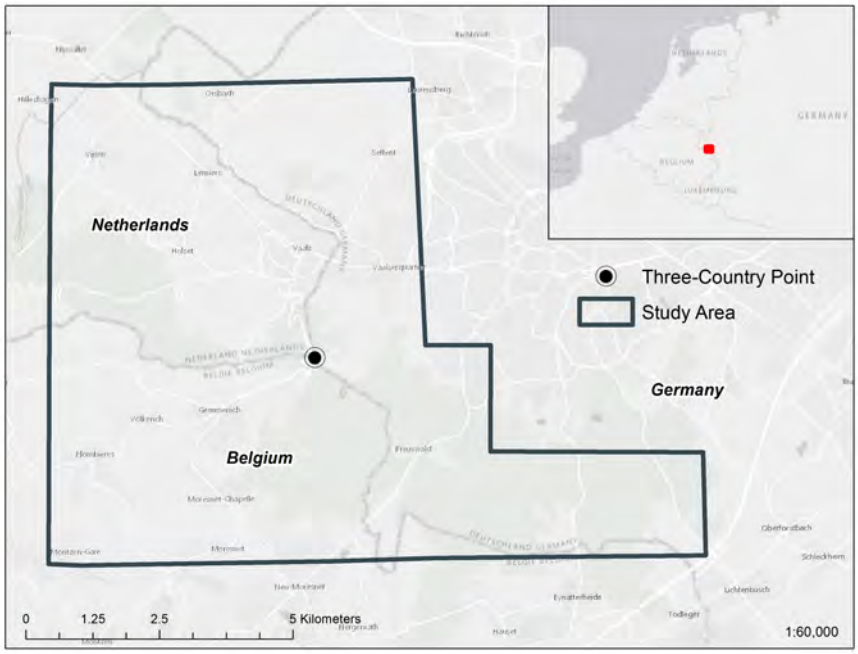


Figure 2.2: The boundary of the study area of the research, containing the TCP

to the AHN2 tile numbering scheme. As the DSM dataset has not been filtered, it contains many points with excessively high elevation values that have not been removed yet. Using the DSM data as an input into the coordinate conversion could later result in problems, as these outliers would also get transformed. The outliers are essentially 'noise' in the data, and need to be removed. Therefore, the DTM dataset was chosen over the DSM dataset precisely because it has already been filtered from outliers, and outlier removal was judged to be outside the scope of the research.

The LAZ files downloaded from PDOK had missing spatial reference information. This could be seen with a simple call to *lasinfo*, a command that reports the contents of the header of a LAZ file (that contains the spatial reference information, if there is any). As no coordinate system or map projection information could be seen, the datasets were 'stamped' with their proper spatial reference; the las2las command can be used to assign the proper horizontal coordinate system and vertical units of measure to the dataset. The horizontal coordinate system assigned is the *Rijksdriehoekssstelsel - New* (EPSG: 28992), with vertical units of measure of meters.

All of the LAStools functions can be run from the Command Prompt, allowing for fast processing on LAS/LAZ format data. Then, as the datasets covered large areas on the map, they were subdivided into smaller, more-manageable tiles. A tile size of 1000 meter was used to subdivide the bigger LAZ files (with projection information) into smaller tiles using the *lastile* command. For the Netherlands, this resulted in a total of 29 tiles of 1000 m X 1000 m each. Finally, the *lassort* command can be used to re-order the points inside each LAZ tile according to a space-filling curve such as a Z-order curve. A Z-order curve maps an n-dimensional dataset into a one-dimensional structure, so that points close together spatially are also stored closer together in memory. This is a utilization of the technique of clustering-

storing objects close together which are also often selected together, because they are nearby in space. Before running *lassort*, the points were stored in the LAZ files with no inherent order; *lassort* rearranges these points into a 1-dimensional data structure (defined by a Z-order space filling curve) that could later be used for storage and searching. Sorting the points in the files this way allows for rapid further processing.

Finally, it is unknown whether there are any duplicate points in the LAZ files. This refers to points with the same X, Y, and Z coordinates. It is wise, therefore, to check and rectify such situations using the *lasduplicate* command with the "unique_xyz" option. This will remove any duplicate points wherever found.

The above pre-processing steps have resulted in a set of tiles that are clean, have the correct projection information, and sorted. They are now ready to be utilized as input into the coordinate conversion.

Germany

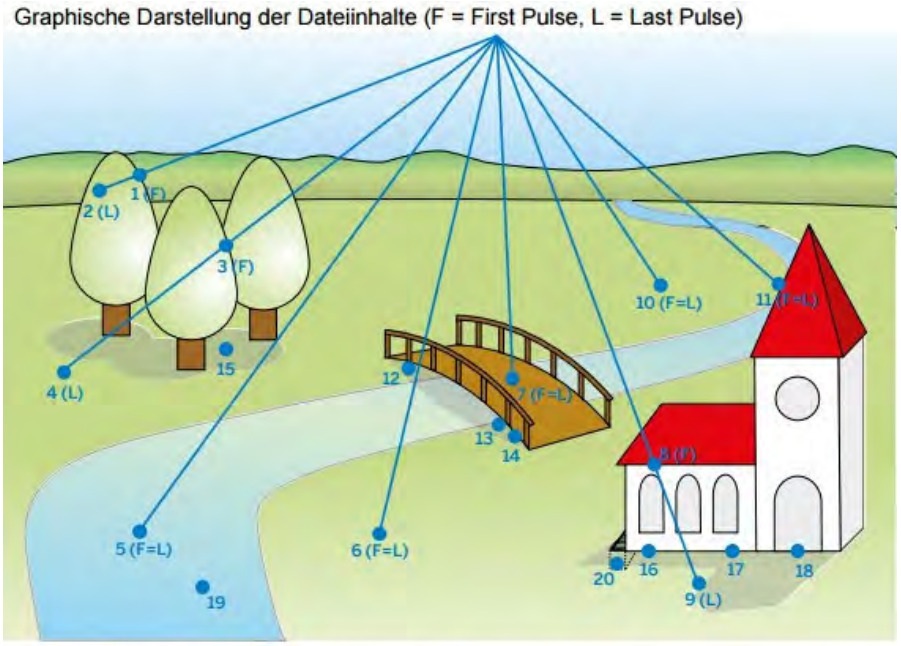
The German province of North-Rhine Westphalia (NRW) has become the first province to offer download links for hundreds of gigabytes of open LIDAR data. Due to some recent "freedom of information" legislation, it's open data organization - OpenNRW - made its LIDAR data available with an extremely permissible license. This is extremely valuable for this research, as NRW is the only German province that borders the TCP.

In comparison with the available point cloud data of the Netherlands, the German point cloud datasets required a lot more pre-processing. They have been provided in ASCII XYZ (.xyz) format as 1 kilometer X 1 kilometer tiles. However, there is more than one tile for each square kilometer as the LIDAR points have been split into different tiles based on point classification and return type. For example, there is a different file for each of the classes "Building", "Bridge", or "Water" for the same 1 km X 1 km tile. Therefore, these different files need to be merged into one common file for each tile. Due to some issues mentioned below, this is not as straightforward as merging the files for a tile into a common file (Isenburg, 2017b).

The city of Aachen, Germany, shares borders with both Belgium and the Netherlands and touches the TCP. Therefore, only the XYZ files of this city were downloaded. There were a total of 1,218 files for the city, out of which less than 150 were useful as they fell inside the area of the city considered relevant for the research. As each XYZ file represents a different class and there are multiple XYZ files for each tile, in total these files cover 34 separate tiles. The types of classifications and/or return types found for the points falling into these tiles, along with brief descriptions of these classifications are provided below; the names of the classifications are German acronyms:

- "Lpb" - the last return points classified as "ground"
- "Lpnb" - the last return points classified as "non-ground"
- "Lpub" - the last return "underground" points with very low elevation
- "Brk" - the points that reflected off of bridges
- "Ab" - synthetic points created under bridges
- "Ag" - synthetic points created under or beside buildings
- "Aw" - synthetic points created on water body surfaces

All the points with “ab”, “Ag”, or “Aw” classifications are therefore “synthetic points” that fill up ground areas that could not be reached by the laser pulse from the LIDAR scanning device; the “a” stands for “ausgefüllt” (German for “filled up”). These points were created by the land survey department that generated the LAZ files to replace LIDAR under bridges or next to buildings, areas where the laser pulse could not reach (Isenburg, 2017b). The goal of creating synthetic points is to provide an accurate surface depiction in these low point density areas, and to provide an uninterrupted transition in point density from surrounding reachable areas to these inaccessible areas. The distinct classifications of points in the dataset are illustrated in Figure 2.3 below, obtained from OpenNRW.



Dateibezeichnung	synthetische Punkte
-ab_	12, 13, 14, 15
-ag_	16, 17, 18
-aw_	19

Dateibezeichnung	gemessene Punkte
-fp_	1, 3, 5, 6, 7, 8, 10, 11
-lpb_	4, 5, 6, 9, 10
-lpnb_	2, 11
-lpub_	20
-brk_	7

Figure 2.3: Information on point classifications and return type in the German LIDAR dataset, provided by OpenNRW. Points 12-19 are synthetic (or manually surveyed) points that are located in areas that could not be reached by the laser pulse. F = first return, L = last return. As ours is a DTM dataset, there are no first returns (class “Fp”). This means that all pulses have only a single return (a last return), as the German data contains no intermediate returns.

The German LIDAR datasets contain either only a single return or two returns (a first and a last return). However, since we are dealing with a bare-Earth DTM, there are no first returns. There are only last returns. It is only if we are dealing with a DSM that first returns also need to be incorporated.

To prepare the XYZ files for subsequent data processing, they need to be compressed into smaller sizes. Furthermore, as the datasets do not contain spatial reference information in their header, they need to be assigned this

information. The *laszip* command was used to reduce the files to one-tenth of their original sizes without any loss of information and save them as LAZ files (Isenburg, 2017a). From now onwards, further operations would be applied directly to the LAZ files. Also, the proper horizontal and vertical coordinate system information was assigned to the LAZ files using this command.

To visualize the boundaries of these tiles in Google Earth, the *lasboundary* command was used to convert the boundaries into Keyhole Markup Language (KML) (.kml) files. KML provides an internationally standardized XML-based format for encoding geographic data that can be consumed by Internet maps and 3D visualization environments such as Google Earth. Converting the boundaries of the tiles into KML allows for a quick glimpse into the extent and characteristics of the surface of each tile.

Although separate files have been provided for each classification and return type of points, the points within each file have not been 'stamped' with their proper classification code. This was done as an extra precautionary step, in case it would prove useful in the future when the data is loaded into a web viewer. These classification codes have been predefined by ASPRS and can be found on the web (ESRI, 2017) . Furthermore, the synthetic points created by the survey agency were assigned a synthetic flag; a classification flag allows an additional classification attribute to be assigned to a point. The type of flag (synthetic) indicates that this point was created by a means other than LIDAR collection. The *las2las* command can be used to assign the correct classification code and classification flag to the points.

The classification codes assigned to the points of a respective class are:

- "Brk" - code 17 for "Bridge Deck"
- "Lpb" - code 2 for "Ground"
- "Lpnb" - code 1 for "Unassigned"
- "Brk" - code 7 for "Low Point"
- "Ab" - code 2 for "Ground", and synthetic flag 1
- "Ag" - code 2 for "Ground", and synthetic flag 1
- "Aw" - code 9 for "Water", and synthetic flag 1

Now, each point within a file has the proper classification code and/or flag assigned. The next step is to combine the different files for each tile into a single file. Using *lasmerge*, these files were merged into a common LAZ file. As each original file had the row and column number of the tile embedded into its name, it was not difficult to merge the matching files for a tile into one. Then, as an additional QA/QC check, the *lasduplicate* command with the parameter "-unique_xyz" was used to remove any duplicate points - points with the exact same x, y, and z values (if any). Next, *lassort* was used to store the points in a spatially coherent manner, using a Z-order space-filling curve. The output files were saved as LAZ files.

The German point cloud dataset contains five classifications; that is, each point has been assigned one out of five classification codes: Bridge Deck (17), Ground (2), Unassigned/Unclassified (1), Low Point (7), and Water (9). Upon closer inspection of Figure 2.3 above, however, it can be seen that points that are part of the 'Unclassified' class can also fall on top of buildings, trees, or other above-ground objects. For example, points 2 and 11

in the figure are located on top of a tree and building, respectively. These cannot be considered as depicting the bare-Earth surface and hence should be excluded from further processing. The 'Unclassified' class in the ASPRS LAS format has classification code 1. The las2las tool within LAStools with the ' —drop_class 1' parameter was used to remove points in this class from the LAS files. Now, all of the points contained within the point cloud represent a DTM.

The above sequence of steps have resulted in LAZ files that are clean, merged, sorted, and indexed. These files are now ready for the coordinate conversion.

Belgium

In contrast to the Netherlands and the German province of NRW, the Belgian region of Wallonia (which borders the TCP) does not provide any publicly available LIDAR data in the form of LAS/LAZ files. However, the Geoportal of Wallonia does provide a dataset in GeoTIFF (.tiff) or ESRI file geodatabase (.gdb) formats that is a terrain model of Wallonia resulting from acquisitions from LIDAR sensing carried out between December 12, 2012 and March 9, 2014. The dataset had to be requested, and after a time-consuming process to acquire the data, it was received as a set of separate GeoTIFF files covering the requested extent of coverage (the north-eastern portion of Wallonia bordering the Netherlands and Germany). The GeoTIFF files have a one-meter resolution and already have the proper spatial reference information assigned to them.

Even though LAS/LAZ files were not available for Wallonia, the GeoTIFF files can be converted to LAZ format with a few steps. ESRI ArcGIS Desktop 10.3 was used to perform most of the subsequent processing of the data. The first step is to merge the delivered rasters (tiles) into a single big raster; the *Mosaic to New Raster* tool was used to create this raster. Then, a study area polygon representing the extent of the GeoTIFF raster needed for Belgium was digitized, and the mosaicked raster clipped to the extent of this study area using the *Clip* tool. A side-effect of this process was that the cells of the raster without any elevation values at all got assigned a value of 0 by default. These values had to be reassigned as 'NoData' cells using the *Reclassify* tool. Therefore, after running this tool, all cells having actual elevation values (¿ 0) in the input GeoTIFF files had a value, and all other cells had values of 'NoData'. Next, to create an ASCII gridded XYZ (.xyz) file from this data, the *GDAL_translate* tool was used. GDAL, the Geospatial Data Abstraction Library, provides a suite of tools for transformation of raster and vector data formats. *GDAL_translate* specifically can convert raster data to many other formats. Finally, once an XYZ file has been created, it can be converted into a LAZ file using the *laszip* command provided within LASTools.

There is no need to tile this file, since the number of points contained within the file is relatively small and therefore the file has a very small size. The above sequence of steps have resulted in a LAZ format file of resolution 1 meter spatial resolution that is now ready for the coordinate conversion.

2.2 DEFINING AN IDEAL COORDINATE REFERENCE SYSTEM (CRS)

The desire is to have an ideal CRS for point cloud dataset interoperability. The last word indicates that the aim is to define a solution that could work for sources stemming from different sources and for global scales. This is also related with the method with which the point cloud datasets are ultimately visualized. To do so, there are 2 options; either by using a Digital Earth (Virtual Globe) or by using a flattened view of the Earth (i.e. *projection*).

2.2.1 Projection or Virtual Globe

There are mainly two different ways to describe a position that is related to the Earth: the Cartesian Coordinate System or the Geodetic Coordinate System, also referred to as the Curvilinear Coordinate System (Featherstone and Vanicek, 1998) (see Figure 2.4). The 3D Cartesian Coordinate System is mainly used by satellites to measure locations on the Earth. It has some disadvantages: it measures positions in X, Y, and Z coordinates with its origin at the center of the Earth if it is an Earth Centered, Earth Fixed (ECEF) system. This results in large numbers that take up a lot of storage space. To store one point on the surface of the Earth, it takes at most 7 digits for each axis, without any decimal numbers, when storing coordinates in units of meters. (Marel, 2016). Moreover, it is difficult to represent the height above the geoid in a Cartesian Coordinate System.

In comparison, the Curvilinear Coordinate System has the capability to communicate information about the surface of the Earth (height values), which is important for point cloud data (3d information). It represents the surface of the Earth using an ellipsoid, using ϕ as the geocentric latitude, λ as the longitude, and h as the height above the ellipsoid. There are no other simple rotational shapes that match the true Earth better than an ellipsoid and therefore it is the most common representation to describe points on the Earth (Marel, 2016).

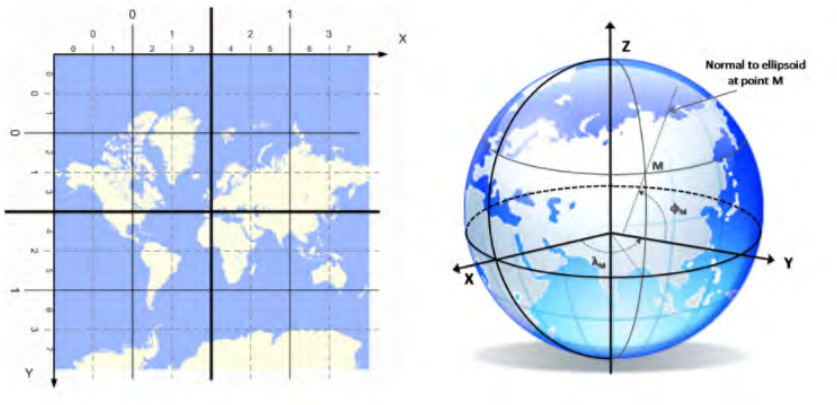


Figure 2.4: Cartesian Coordinate System and Geodetic Coordinate System

Doing spatial analysis on a sphere is very complicated. It is even more complicated when using an ellipsoid (Marel, 2016). Since these kinds of

operations take less time on a Cartesian (2D-)grid, a projection is used to acquire Cartesian coordinates. There are different kind of projections. Marel (2016) gives the most common (miscellaneous projections disregarded):

- Cylindrical map projections: This projection is constructed by wrapping a cylinder around the Earth. An example is the well-known Mercator projection.
- Conic map projections: A cone wrapped around the Earth.
- Azimuthal map projections: The Earth is projected on a plane tangent to the Earth. An example is the Dutch RD Amersfoort projection, the projection of the point cloud of the Netherlands.

Projections are acceptable for local scales, but not so for global scales since the curvature becomes more significant. Projecting a map globally will result in large distortions. There are three types of distortions, of which a projection has at least one. The projection can distort in area or scale, in distance or in shape. A conformal projection preserves local angles. The Mercator projection is an example of a conformal projection. An equal-area projection preserves the area, but therefore distorts shape. An equidistant projection preserves the distance, but can only achieve this in some directions (Marel, 2016).

This research is attempting to achieve a point cloud map on a global scale. A Geodetic or Cartesian Coordinate System both have their advantages and disadvantages. While geodetic systems do not have (projection) distortions, they are hard to do spatial analysis on. On the other hand, projected (Cartesian) coordinates always have distortions in either shape, area and/or distance. A DGGS deals with both problems. It uses the geodetic system to divide the Earth in a multitude of small regions, which results in having only small distortions in every area. The DGGS is explained in the next paragraph.

2.2.2 Digital Earth Frameworks: Traditional and Geodesic Discrete Global Grid Systems (DGGS)

The Digital Earth Frameworks can be divided in two categories; the traditional DGGS (see Figure 2.5) and the Geodesic DGGS, as named by Sahr et al. (2003).

The traditional DGGS are already used by many applications, algorithms and software. They use simple and efficient algorithms and they can make use of ordinary data structures and visualization devices. Many users are more familiar with these kind of representations (Sahr et al., 2003) instead of the Geodesic DGGS. The traditional DGGS are based on latitude - longitude coordinate systems (Amiri et al., 2015) as they represent the Earth's surface using the meridians and parallels (Ottoson, 2001). Therefore, the constructed grid subdivides the Earth in equal degree steps along latitude and longitude. Thus, the cells in the grid of a traditional DGGS are not of equal-area, which leads to the disadvantage that data analysis (Amiri et al., 2015) and statistical analysis (Sahr et al., 2003) is rather complicated. Apart from the cell distortions regarding the area, the shape and the distance between the cell centroids are getting larger as one moves from the equator to the poles. In general, the angular and areal distortions are considered high when implementing this global grid type (Amiri et al., 2015). In addition,

Figure 2.5: Traditional DGGS - The surface of the Earth is divided by meridians and parallels.

as can be seen in Figure 2.5 used by Ottoson (2001), on the poles the cells of these kind of global grids become triangles. This has obliged many relevant applications to use special grids for these regions. Lastly, even the adjusted traditional DGGS that manage to achieve more regular cell region areas have the disadvantage that the cell shapes are more irregular and finding the neighboring cells constitutes a more composite procedure (Sahr et al., 2003).

In comparison, a Geodesic DGGS is a spatial reference system that uses a hierarchical tessellation of cells to partition and address the globe (Strobl et al., 2016). Therefore, a Geodesic DGGS is considered as a sequence of discrete grids that are consisted of grids of finer resolution (Sahr et al., 2003) (see Figure 2.6). A grid of a single resolution is not considered to be a DGGS (Strobl et al., 2016). Therefore, this is one of the advantages of the geodesic over the traditional DGGS.

In addition, the grids of a DGGS are used to address the Earth's surface, (partly) by using inverse map projections (see Section 2.2.3); firstly the planar cells of the grid are created and then they are converted to spherical or ellipsoidal surfaces. According to OGC (Strobl et al., 2016), a Geodesic DGGS must use an 'equal-area' projection (i.e. areas of features are preserved). The reasons are that 'equal-area' cells ensure uniform data coverage in an area of interest (AOI), statistical analysis becomes efficient and this also allows for interoperability when multiple sources have to be integrated. This is due to the way data are projected on the planar grids. Namely, the same transformation parameters are needed whether an area of interest (AOI) lies in Amsterdam or in New York. Here comes a difference between the Geodesic DGGS and the Universal Transverse Mercator (UTM) coordinate system. Apart from the fact that UTM constitutes a projected reference system (so it cannot be used to shape a Digital Earth as desired), it also requires the specification of different parameters when an AOI is placed in Europe and when in USA.

Due to the reasons described a DGGS constitutes a solution that allows big data (such as point cloud) to be easy to access, store, sort, process, transmit, integrate, visualize, analyze and model. On top of that, OGC is on the way to standardize a common analytical framework for fusion and analysis

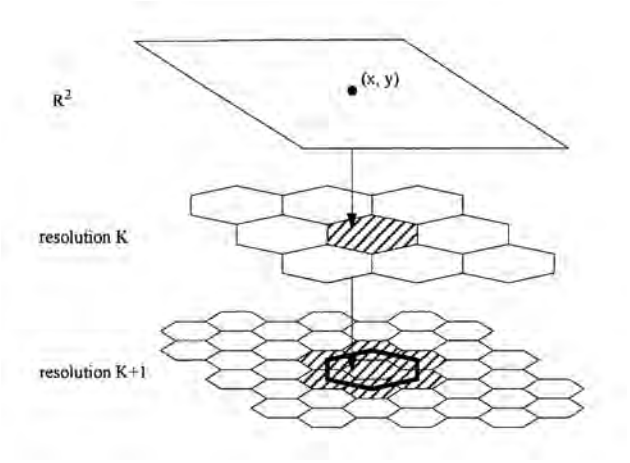


Figure 2.6: Grids consisting of grids of finer resolution.

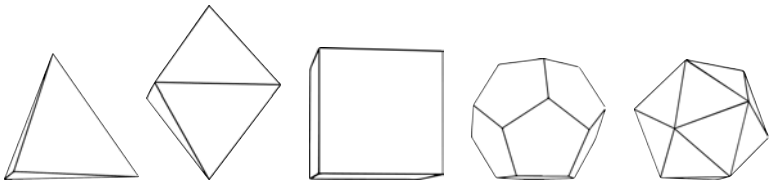


Figure 2.7: Possible polyhedrons to approximate the Earth's surface.

using DGGS.

The comparison provided and the advantages of the Geodesic DGGS over the traditional one, lead to the decision to choose a Geodesic DGGS as the grid reference system for point cloud data representation. Especially, the fact that they can support multi-resolution partitioning of the Earth's surface is important since the density of point cloud datasets varies in multiple areas and for multiple reasons.

2.2.3 Ideal DGGS for Point Cloud Datasets

This section includes some small modifications as a result of further research. Particularly, an 'equal-area' projection is preferred instead of the Dymaxion projection for the reasons explained in the *DGGS design parameter: Transformation* part of this section.

Here, the different options for the construction of a DGGS as a reference grid for point cloud data representation are provided and compared in order to define the optimal solution.

As Sahr et al. (2003) noted, there is no one Geodesic DGGS suitable for every application. Thus, in this chapter the design parameters to construct a Geodesic DGGS, are given and the possible options for each parameter are compared to choose an optimal reference grid for point cloud representation.

The Geodesic DGGS can be constructed by defining 5 design parameters. Firstly, it approximates the Earth by spherical or ellipsoidal polyhedrons (Amiri et al., 2015) and thus consists of several planar faces (see Figure 2.7). Also, the polyhedron -with its faces, edges and vertices - needs to have a fixed orientation which will resemble the real Earth's surface (PYXIS, 2006). In addition, it requires a method that will subdivide the polyhedron in a way that will allow for a multiple resolution discrete grid (Amiri et al., 2015) and a method to transform the planar faces to a corresponding spherical or ellipsoidal surface. Lastly, a method to assign points to each grid face is needed.

Choice of polyhedron

The approximation of the sphere (or of the ellipsoid) can be achieved with the icosahedron, cube, tetrahedron, octahedron (Dutton, 1996) and dodecahedron (Amiri et al., 2015)(see Figure 2.7) .

All the polyhedrons have their strong and weak points with respect to the DGGS construction. The tetrahedron (consisting of triangles) represents one of the simplest solutions (Amiri et al., 2015). The cube (consisting of quadrants) can be efficiently handled (Sahr et al., 2003) due to the fact that its faces can be subdivided into square quadtrees (Amiri et al., 2015). The octahedron can be oriented with vertices at the north and south poles, and with vertices at the intersection of the Prime Meridian and the Equator (Dutton, 1996). Thus, a better approximation of the Earth can be defined this way. An icosahedron can give less angular and areal distortions (that are relatively small) in comparison with all the pre-mentioned polyhedrons. The smaller the size of a polyhedron's faces, the less the distortions introduced when transforming between a planar face to the spherical surface. The pentagon of each dodecahedron though, can be further subdivided into 5 isosceles triangles, which have sizes even smaller than the triangles created in an icosahedron. That means that the distortions in comparison with the icosahedral triangles, are even less (Sahr et al., 2003).

However, the faces of the dodecahedron are pentagons and there is no possible refinement to get to a greater resolution for those (Amiri et al., 2015). Besides this, another negative point of the dodecahedron is that when it is subdivided into isosceles triangles, the most important property of a Geodesic DGGS is lost; the edges of the faces are no longer of equal lengths, which leads to inconsistencies along the edges between faces (Sahr et al., 2003). The disadvantage derives from the fact that regular tree-based algorithms cannot directly be used.

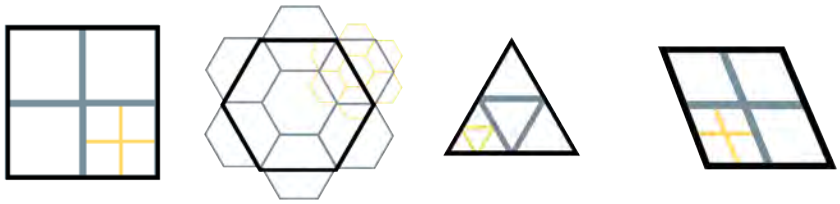
Regarding the octahedron, since its faces are larger than the icosahedron, this means that the distortions involved are even larger. The same holds for the tetrahedron and the cube, but the distortions are even more noticeable in these cases (Amiri et al., 2015).

In Appendix C.1 the comparison of the design options are given. From there, it is easy to see that the optimal polyhedron to use for the construction of a DGGS would be the icosahedron. This is because it introduces the least possible distortions and does not have drawbacks that other polyhedrons have. The fact that the areal distortions are minimal benefits point cloud datasets at a country level, because if the areas are preserved, then the

datasets can fit efficiently to the approximated (to the Earth's surface) country boundaries. In addition, since point clouds are very accurate spatial data, they can be used for many applications, including measuring procedures. If the distortions introduced on the reference grid are high, then they cannot be used for such applications.

Cell type

The partitioning of the polyhedron refers to the cell type chosen for the creation of multiple resolution grids and the refinement (or aperture) parameter. The partitioning can be achieved with 4 partitioning topologies (see Figure 2.8): squares, triangles, hexagons and diamonds. The refinement defines how many times the edge of a cell is split into finer cells.



(a) squares **(b)** hexagons **(c)** triangles **(d)** diamonds (rhombuses)

Figure 2.8: Partition of cells (faces of a polyhedron can be achieved with multiple shapes. A finer resolution is made by color. Modified figure from Amiri et al. (2015).

The squares are used for a cube polyhedron subdivision. The triangles and the hexagons are used for polyhedrons with triangular faces, such as an octahedron, icosahedron and the modified dodecahedron (Amiri et al., 2015).

One of the disadvantages of squares is that their geometry makes them unsuitable for triangulated polyhedrons and the desired polyhedron - the icosahedron - belongs to this family.

The coarsest diamonds though can do that and can also make use of the quad-tree based algorithms. Their disadvantage though is that they do not provide uniform adjacency (Sahr et al., 2003) and thus the centroids of the neighboring cells cannot be found in the same distance from the centroid of an original cell, since there are two kinds of neighbors; those that are sharing an edge and those that are touching at a vertex (PYXIS, 2006). This according to Amiri et al. (2015) results in more complicated data analysis than when uniform adjacency applies. Nevertheless, this is not a good enough reason to exclude diamonds as an option, since efficient indexing of the diamonds' centroids can help with finding the neighboring cells and simplify data analysis.

The triangles as the partitioning topology have the same main disadvantage that squares have, that they do not provide uniform adjacency. Besides that, they also do not support uniform orientation, and this might affect algorithms that take into account the orientation of the triangles.

Hexagons discretize/quantize the plane with the smallest average error, provide the greatest angular resolution and have uniform adjacency. The latter not only means that the centroids of all the 6 neighbors of a hexagon are located at the same distance from that hexagon's center, but it also means that its neighbors share an edge with it. According to Frisch et al. (1986) this is efficient for simulations of spatial discrete (instantaneous) events. This means that events are separated by intervals of time which is also the case with point clouds. The main drawback of using hexagons is that it is unattainable to fully partition the spherical/ellipsoidal Earth with hexagons as soon as a polyhedron with triangulated faces is used. Even more critical and negative characteristic is that the multi-resolution capability of the Geodesic DGGS cannot be achieved using congruent hexagons (Gutierrez, 2007) as they cannot be precisely divided into smaller hexagons (Sahr et al., 2003). Thus, a hexagonal subdivision either needs to be single-resolution, not hexagonal or not congruent, which means that a hexagon cannot exactly be decomposed into smaller hexagons.

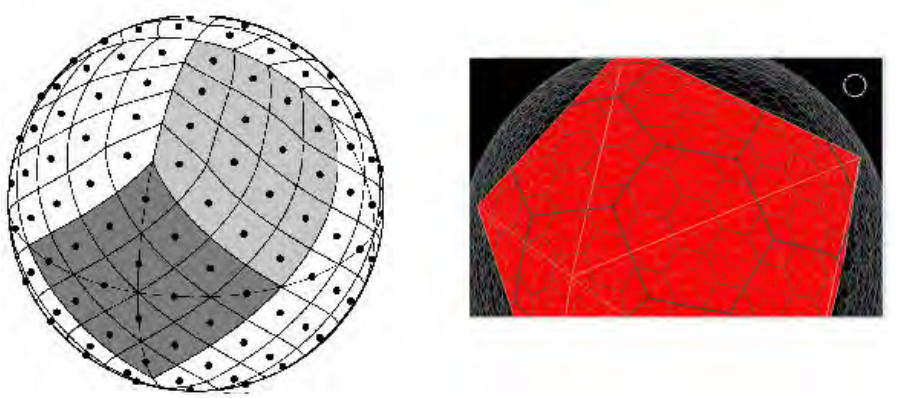


Figure 2.9: Comparison between congruent diamonds (left) and non-congruent hexagons (right) for multi-resolution.

Multi-resolution is considered essential for the representation of point clouds, since the sources have different origins and thus it is highly likely that the density will differ. This could also apply in areas where different scanning methods were used due to less detailed environments. But multi-resolution can be achieved with non-congruent hexagons. The disadvantage though is that regular tree-based algorithms cannot directly be used on these hexagon hierarchies. Sahr et al. (2003) states that it can be better to use an incongruent hexagon system, instead of a congruent system with another polygon subdivision.

Small apertures (not as a term of number) provide more resolution grids. This represents an advantage especially for point cloud datasets in which the information is rather dense. The aperture equals n to the power of 2, where n equals the pieces and the edge is split to create congruent (equilateral) polygons. An odd aperture is also possible and then aperture equal to n to the power of 2, while n = 2m and m represents a positive integer. In this case though the subdivided grid consists of non-congruent polygons (Sahr et al., 2003).

The research has shown that hexagons have most of the advantages over the other polygons, but their disadvantage is that they cannot provide multi-resolution level support with congruent hexagons. This is possible though with not-congruent hexagons. However, in that case the beneficial quad-tree algorithms will require further processing to allow for use. Diamonds could be used as an alternative, but they have other disadvantages such as the fact that they do not provide uniform adjacency and therefore the data analysis required to be performed is more complex (see Figure 2.9). This can be cumbersome especially when handling big data such as point clouds. Besides this, hexagons are more popular and there is also good relevant documentation, which is also an advantage in case of time limitations.

Orientation

The orientation of the polyhedron needs to be specified to ensure that a proper approximation of the Earth will be achieved.

The orientation of a platonic solid, i.e. a polyhedron consisting of congruent faces, can be defined with one vertex of the polyhedron and one azimuth from that vertex to the close by vertex.

For an icosahedral polyhedron there are at least 3 possible solutions for orientation (see Figure 2.10). One option is that the vertices of the triangles meet at the poles and one edge is aligned with the Prime Meridian (Sahr et al., 2003). The drawback of this orientation is that there is no symmetry on the Equator, which is though the actual case. Second option is to use a specific orientation called Dymaxion orientation, in which all the vertices of the triangles fall in the oceans. The advantage of this is that the icosahedron can be unfolded without any borders on the land. Lastly, the triangles can be oriented in a way that there is symmetry on the Equator which already constitutes an advantage. Another benefit of this orientation is that the number of the vertices of the icosahedron that fall into land can be minimized.

By comparing the three already existing orientations for icosahedra, the 3rd option is chosen as the optimal. That will be able to provide an actual representation of the Earth around the Equator (symmetry), in combination with the least possible amount of points that connect polyhedron cells on land.

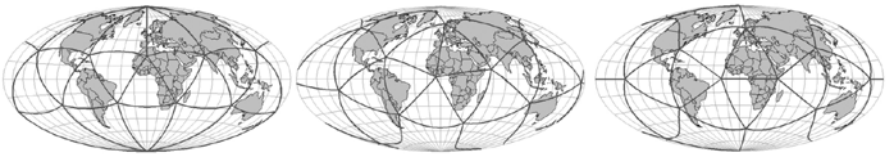


Figure 2.10: Three possible orientations for icosahedron.

Transformation

A transformation is needed to convert each planar partition of the polyhedron to its corresponding spherical or ellipsoidal part of the Earth (see Figure 2.11). To do so, there are two methods. Firstly, the method *direct spherical subdivision*: as the name implies, this directly creates spherical partitions from the planar ones. The other method, is to apply an inverse map projection (Sahr et al., 2003).

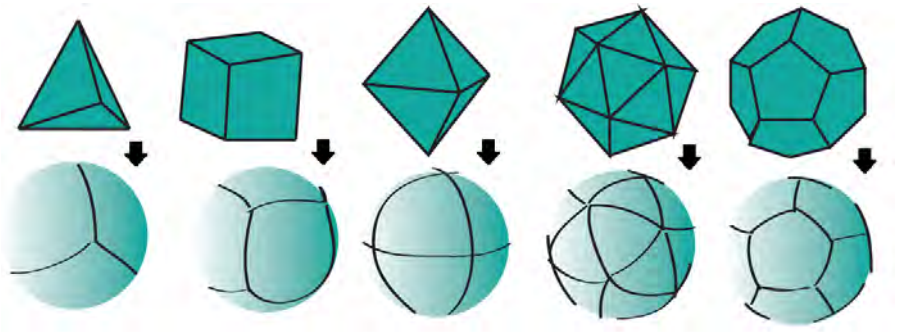


Figure 2.11: Visualization of the conversion result from planar grid cells to the corresponding spherical cells. (Image from (Amiri et al., 2015))

There are at least two ways to implement the method of direct spherical subdivision, while acquiring simultaneously equal area for each grid cell. The first one uses two great circle arcs adjusted to the mid-point of each new edge, which is formed by using that break point (mid-point). The other one uses small circle arcs (Song et al., 2002).

However, Song et al. (2002) stated that inverse map projections may have higher efficiency than using the two pre-mentioned methods. Each inverse map projection must fulfill this property to be used for this purpose: map the straight-planar face edges to the great circle arc edges of the related spherical face. There are a few of such projections.

Regarding an inverse map projection, Snyder's ISEA was used. However,its disadvantage is that it produces changes in directions (Furuti, 2015), which are visible as breaks in the lines connecting vertex to face centers 2.12). Therefore, further research should be done to find other possible equal area projections that do not introduce these errors.

The gnomonic one can implement that property for all the polyhedra, but it has the disadvantage that it introduces high distortions on the areas and shapes. The equal area projections that Snyder created produce changes in the directions, which are visible as breaks in the lines connecting vertex to face centers (see Figure 2.12). The projections of Snyder are applied to all the platonic solids, out of which for the truncated icosahedron the areas are preserved almost exactly (Furuti, 2015). Fuller's Dymaxion (Fuller, 1975) projection is implemented only for icosahedrons, it gives less area and shape distortions than the gnomonic one and also less shape distortion than Snyder's related projection (Snyder, 1992).

However, the Dymaxion projection does not provide equal area for each spherical cell. By comparing the transformations from planar cells to spherical (or ellipsoidal) cells, the decision initially was to use Fuller's Dymaxion method, mainly due to the minimum distortions that it provides. As stated in the bibliography though, its drawback is that the spherical cells will not be of 'equal area'. In addition, in a presentation of 2016 from OGC (Strobl et al., 2016) regarding the standardization of DGGS with respect to big data handling, it is mentioned that Global Grids that do not have 'Equal Area' cells (i.e. no distortions on areas and the grid cells are of equal area) are not sufficient to be described as a DGGS. This is due to the benefits that can be provided by using such a projection. Specifically, by using an equal-area

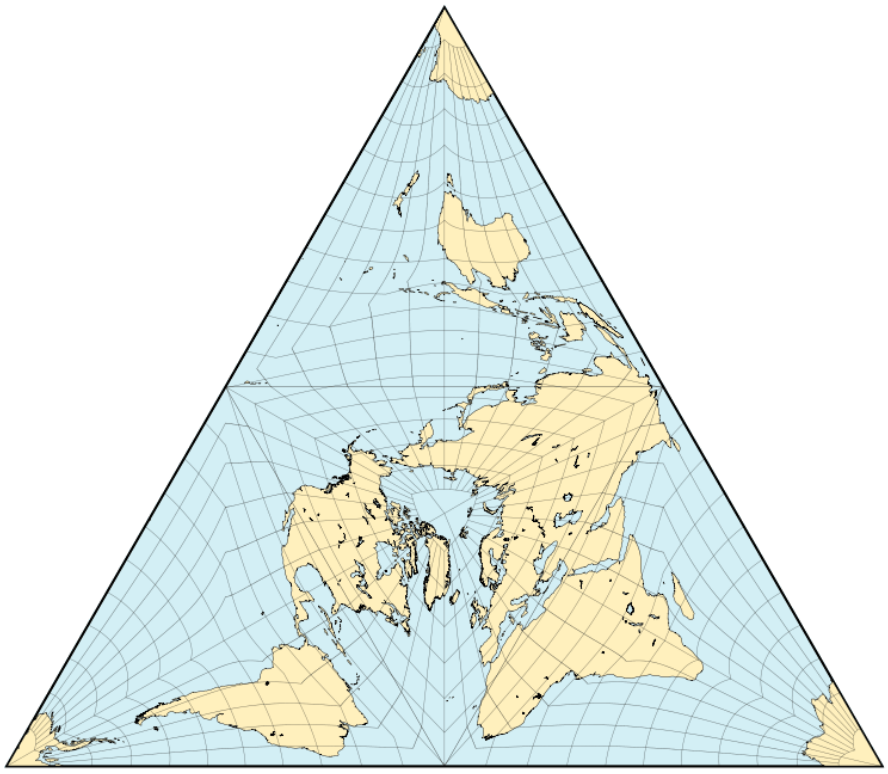


Figure 2.12: Breaks in directions when Snyder's equal-area projection is applied to a tetrahedron.(Image from (Furuti, 2015)

projection it is then certain that the data coverage is uniform. For example, analysis of data is not applied in a smaller/larger area in Belgium than the truth. Since the areas are preserved, the statistical analysis becomes efficient and lastly, this allows people to use DGGS in combination with other data infrastructures. Furthermore, in the same material it is clearly stated from OGC that the Dymaxion projection is considered invalid (for the reasons given above and because it is not an equal-area projection) under the new OGC DGGS Core Standard. Thus, the Snyder equal-area projection for an icosahedron is ultimately chosen as the optimal solution.

To summarize, an ideal DGGS for the purpose of the current project should be constructed using an icosahedron with hexagonal cells. The triangles of the polyhedron should be oriented in such a way that there is symmetry at the Equator and the least number of points on land. Lastly, Snyder's equal-area map projection should be used to convert the planar faces to the ellipsoidal ones.

2.2.4 Existing Solutions for DGGS Implementation

There are some small modifications in comparison with the mid-term report as a result of further research. It was first stated that only the rHEALPix DGG follows the draft standards of OGC, but in fact ISEAH3 follows these as well. Next to that, the ideal transformation is again ISEA and not a Dymaxion projection. On top of that, the DGGRID software (Sahr et al., 2015) was not known when writing this section. This software makes easier im-

plementing a DGGS (that uses icosahedron to represent the erath's surface).

In the previous section we specified the ideal DGGS in our use case. However, developing a DGGS implementation is rather hard to accomplish. It is time consuming to assemble personalized equations and to implement this in code. With a limited time-frame of only 10 weeks, there is a need to find already existing solutions that use DGGS. There are multiple existing infrastructures that have already implemented DGGS and we will elaborate on two: the rHEALPix DGGS and the PYXIS WorldView Studio.

rHEALPix DGGS is an extension from HEALPix DGGS. It uses a planar projection, which has a low average angular and linear distortion. Thereby, the area of the grid cells is equal at every iteration level, which makes statistical analysis easier. On top of that, the planar projection of rHEALPix consists of horizontal-vertical aligned nested square grids. It is not an icosahedral, but a cubic DGGS (Gibb, 2016).

The code is open source and written in Python and the DGGS map projection is implemented in Proj.4. This projection is equal-area. And lastly, its implementation meets the OGC DGGS standards. There are a number of requirements to meet these standards. Here are the most important OGC specifications (Purrs, 2015). A DGGS should:

a) exhaustively cover the globe without overlapping cells

b) define multiple grids forming a system of hierarchical tessellations

c) preserve the area

d) have equal area cells in each level of precision.

While rHEALPix is only an implementation, PYXIS WorldView Studio is a commercial client application. Therefore it is easier to use, but cannot alternate to have our own functionalities and is thus limited to what PYXIS provides. Like rHEALPix, the PYXIS projection is equal-area, but it uses ISEA Hexagon Aperture 3 (ISEAH3) DGG, a grid that was originally developed by the Oregon State University (Gibb, 2016). This means it is part of the family that is described as the ideal DGGS in the previous section. It is a multi-resolution solution, and therefore uses incongruent hexagons (see Figure 2.13).

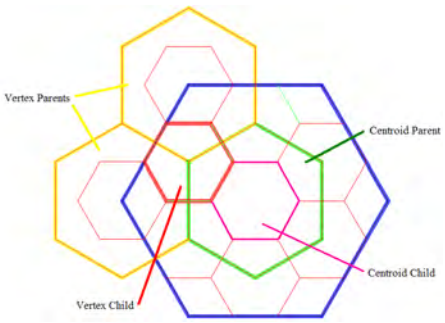


Figure 2.13: Incongruent Hexagons

PYXIS WorldView Studio can handle a bunch of file formats, like WFS, WMS, GML or SHP files. It supports raster data with an elevation per cell. Unfortunately, this client application doesn't support LAS-files or other point cloud formats and since it isn't open-source software, there is no way to change that (WorldView Studio, 2016). However, there are some R implementations that use the same grid (ISEAH3).

2.2.5 DGGS Determination

So far, there are three options from which we can choose. Since PYXIS doesn't support LAS-files, this option is not suitable. The ISEAH3 implementation however is. The second option is the rHEALPix implementation. The last option is using the parameters that were chosen best in Section 2.3.3. Table 2.2 shows the different options.

Table 2.2: DGGS implementation comparion

DGGS Parameter	rHEALPix	ISEAH3	Ideal DGGS
Polyhedron	Cubic	Icosahedron	Icosahedron
Cell type	Square	Hexagonal	Hexagonal
Orientation	Meridian & Equator Symmetric	Equator Symmetric	Equator Symmetric
Projection	Equidistant Cylindrical Projection	Area preserving planar projection	Area preserving
Transformation	Unspecified	ISEA	ISEA
OGC	Conform	Conform	-

The table shows that the implementation of rHEALPix is deviating the most from the ideal DGGS in this project. Because it is a cube, it can be Meridian and Equator symmetric. Previously though, the main advantage to use rHEALPix is that it is made according to OGC draft standards, but now we know that ISEAH3 is also following these. However, there is another software that can implement many kind of grid systems. Since it can construct the ideal DGGS of this use case and it is open source software, this is the method that is going to be used in the project.

2.2.6 Base CRS

In order to harmonize the point cloud datasets of the Netherlands, Germany, and Belgium, they need to transformed into a common CRS based on a common datum. They are all in different CRS's and datums at this point, and therefore the coordinates of every point in the LAS files need to be converted into a common CRS and datum. For example, the dataset of the Netherlands is in the Amersfoort/RD New horizontal CRS that is based on the Amersfoort datum and the Normaal Amsterdams Peil (NAP) vertical CRS that is based on the NAP datum; the point cloud dataset of Germany is in the Universal Transverse Mercator (UTM) Zone 32 North horizontal CRS based on the European Terrestrial Reference System (ETRS) 1989 datum and the Deutsches Haupthoehennetz 1992 (DHHN92) vertical CRS based on the DHHN92 vertical datum; and the point cloud dataset of Belgium is in the Belgian Lambert 1972 horizontal CRS based on the Reseau National Belge 1972 datum and the 'Ostend height' vertical CRS based on the Ostend datum. Therefore, it is critical to convert these datasets into a

common horizontal and vertical CRS based on a common datum.

With the arrival of INSPIRE, there is a discussion going on in European countries to head towards a common reference system. INSPIRE shares a goal of this project, to make data interoperable. While their objective is to combine seamless spatial information from different sources across European borders (INSPIRE, nd), the difference is that our aim is worldwide. It is a difficult and time-consuming task to bring all data to a common system, since all databases of government structures are based on their own systems (Geonovum, 2015). To follow the INSPIRE implementing rules, ETRS89, with vertical component EVRS (European Vertical Reference System) should be chosen (INSPIRE, 2008). Another option is to choose for WGS84.

The World Geodetic System of 1984 (WGS84) is a standard for use in cartography, geodesy, and navigation, including Global Positioning System (Wikipedia). A host of key parameters are defined as part of the WGS84 standard, including the choice of an ellipsoid, geoid, and a standard coordinate system. An oblate ellipsoid (an ellipsoid representing the size and shape of the Earth that bulges out at the Equator and is slightly flattened at the poles) is chosen as the datum surface for WGS84. The Earth Gravitational Model (EGM) 96 geoid is chosen as the reference geoid to define the surface of the sea level. The origin of the coordinate system is defined to be the Earth's center of mass. Due to the universal nature of this system, it has gained immense popularity especially with respect to being the reference CRS used by GPS.

The recommended coordinate system for inter-European projects, as stated by INSPIRE, is the European Terrestrial Reference System (ETRS89) (INSPIRE, 2008). The German system is already in ETRS89 and the Dutch system is on its transfer to it. ETRS89 is more internationally oriented than the RD system and is valid in the sea (Geonovum, nd). It is, like WGS84, a geodetic coordinate system and is fixed to the Eurasian tectonic plate. This plate moves 2 centimeter a year and is therefore more accurate than the other system in this area (Geonovum, 2015).

The system that is chosen is WGS84. The main reason for this is the global coverage of the system, while ETRS89 moves together with the Eurasian plate. ETRS89 is more applicable at the specific study area, but the aim of this project is to find a global solution. The need for a centimeter positional accuracy is low, since the data itself is only of maximum 50 centimeter (positional) accuracy. Therefore, as it defines a standard CRS that is applicable to the entire Earth, WGS84 was chosen as an intermediate CRS and datum into which the point cloud datasets would be converted. The WGS84 datum with latitude/longitude coordinates has EPSG code 4326.

2.2.7 Discrete Global Grid Generation

The icosahedral discrete global grid needs to be generated according to the design parameters determined in the Section 2.2.3. For the optimal visualization of large datasets of points (extent of countries) and for the reasons explained in that section while comparing a DGGS with the

alternatives, the icosahedral grid is created as explained below.

For that purpose, the software DGGRID (Sahr et al., 2015) was used, which is an implementation mainly by Sahr (2015), but also with other contributors. The software has the capabilities to create the icosahedral grid with the planar hexagonal topology and inversely project the grid (hexagons) onto the WGS84 datum.

Since the DGGRID software can be run only through Unix command line (Sahr, 2015), a VirtualBox was installed to support the aforementioned operating system. The main requirement of the DGGRID software is that a *metafile* needs to be created, which contains the operation to be applied and all the parameters to determine the output of the corresponding operation.

The operation *generategrid* can be used to provide both the hexagonal grids, their centroids and their codes (indexes), while the code for each centroid is the same as its corresponding hexagon. In Figure 2.14, the related metafile with all the desired parameters is given and explained.

```
# specify the operation
dggrid_operation GENERATE_GRID

# specify the DGG
dggs_topology HEXAGON
dggs_res_spec 14
dggs_proj ISEA
dggs_aperture_type PURE
dggs_aperture 4
dggs_orient_specify_type SPECIFIED
dggs_vert0_lat 58.28252559
dggs_vert0_lon 11.25
dggs_vert0_azimuth 0.0
dggs_num_placements 1


# control the generation
clip_subset_type SHAPEFILE
clip_region_files ./InputThreeCountries/LASthreeCountries.shp

geodetic_densify 0.0

# specify the output
cell_output_type SHAPEFILE
point_output_type SHAPEFILE
cell_output_file_name /home/geoloc/OutputThreeCountries/Res14/GridCellsThreeCountriesRes14ISEA
point_output_file_name /home/geoloc/OutputThreeCountries/Res14/GridCentroidsThreeCountriesRes14ISEA
```

Figure 2.14: Metafile created to specify the desired DGG generation parameters, i.e.an icosahedron with triangle orientation of Sahr (2015), a hexagonal grid, an inverse map projection of equal-area and aperture 4.

Prior to the chosen parameter explanation, it can be stated that the DGGRID software can generate only polyhedrons with 20 triangular faces, i.e. icosahedrons, and thus this is not a parameter to be specified. After the operation definition (*generategrid*), the desired DGG is defined. Firstly, the hierarchical spatial partitioning method for the subdivision of every face of the icosahedron is set, by defining the hexagons as the cell type. Part of that parameter is also the *aperture*, which is set to 4 and of type *pure* since it is not mixed-aperture (i.e. a combination of different aperture values is not used). This variable is related to the way the hexagons are constructed at each resolution. In particular, that means that the ratio of areas of the planar hexagons at a resolution k and k+1 is equal to 4 (Sahr et al., 2003). Next, the orientation (of the faces) of the icosahedron relative to the Earth (Sahr, 2015) is specified so as to provide symmetry at the Equator and to allow

only one vertex of the triangular faces of the icosahedron to fall on land (see Section 2.2.3). The definition of the orientation requires that the latitude and the longitude of a vertex of the polyhedron faces is defined and also the azimuth to an adjacent vertex. The desired orientation is determined with longitude 11.25, latitude 58.28252559 and azimuth 0.0 according to Sahr et. al (2003). In addition, the inverse map projection - allowing to transform the initially created planar hexagonal cells to spherical - is set with the parameter *dggsproj* to ISEA, which refers to the equal-area cell projection of Snyder (see Section 2.2.3). Furthermore, the resolution for which the cells are created is specified with the parameter *dggsrecspec*. Lastly, the *dggsnumplacements* parameter defines that the desired operation is performed on a single DGGS (the one defined).

The second step is to specify the area for which the grids will be created. For that reason, a convex hull was created around the point cloud datasets of the three countries (see Figure 2.15) and extracted in shapefile format. This was used as an input to the grid generation operation to indicate the clipped area. Apart from the grid generation, there is also the possibility to generate the centroids of the hexagonal cells. Both cells and centroids come with indexes that are the same for corresponding entities (i.e. the centroid of a hexagon has the same ID as the hexagon itself). Lastly, the output format of the generated grid can be specified. Then, using the command *./dggrid nameOfTheMetaFile.meta* in Unix, the grids can be generated. The same command needs to be specified for all the desired subdivisions as soon as the metafile is modified.

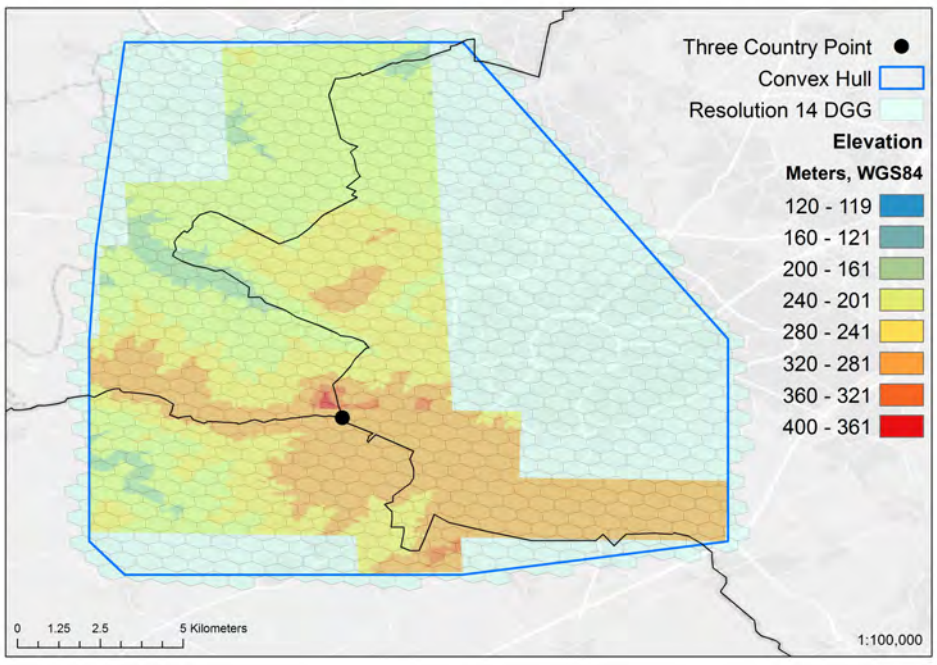


Figure 2.15: The convex hull of the point cloud datasets of the three countries that defines the area for which the hexagonal grid will be created.

For the current project only one resolution is used for further processing. This is because of two reasons; firstly the purpose is to create an appropriate methodology and not to apply it on as many data as possible. Secondly, multi-resolutions have only a meaning if a hierarchical indexing system

is implemented. Some related methods in the DGGRID software were tested, but those did not seem to give the desired results (see Section 2.2.8). Despite that, it can be recalled that many of the advantages of a DGGS (as specified in Section 2.2.1) are due to its multi-resolution support.

To determine which resolution to be used for the purpose of the project, grids were created for resolutions 8 to 18 and grid statistics (on the plane) were calculated using the DGGRID software. These refer to the number of the generated cells, their area, the distance between their centroids, but also the time needed for the DGGRID software to create the grids. As can be seen in Table 2.3, the process was in general executed quite fast.

Icosahedron Hexagonal grid and ISEA inverse map projection				
Resolution	Number of cells	Inter-cell distance (km)	Cell area (km^2)	Processing time (s)
8	3	27,553330	778,298373	0,090
9	5	13,776665	194,574593	0,040
10	9	6,888333	48,643648	0,008
11	24	3,444166	12,160912	0,071
12	71	1,722083	3,040228	0,065
13	255	0,861042	0,760057	0,080
14	952	0,430521	0,190014	0,286
15	3662	0,215260	0,047504	0,479
16	14385	0,107630	0,011876	1,480
17	56972	0,053815	0,002969	5,634
18	226774	0,026908	0,000742	24,208

Table 2.3: Statistics for the hexagonal grids created for a DGG that uses an icosahedron, aperture 4 and ISEA inverse map projection. The statistics refer to a plane and not a sphere and were created using the DGGrid software of Sahr et al. (2015).

However, since these results refer only to a small area and not to an entire country, it is useful to mention that they were generated in a machine with an I3 processor. If a better machine was used the results could be produced even faster. In addition it can be that the first resolution creates very few cells. The grids of resolutions 8, 9 and 10 have been created to visualize this (see Figure 2.16). As it can be observed, this has no meaning since the area of interest is a lot smaller than the created grid. If the number of cells and the cell area among the resolutions are compared, than it can be seen that as soon as a resolution k becomes bigger (i.e. k+1) then the relation between the first and the next resolutions is about 1:4 respectively. This is logical, because this is how hexagonal grids are subdivided. The constructed grids of resolutions 13 and 14 can be seen in Figure 2.17 to make this easier to visualize. The hexagon with index code 135945289 in resolution 13 is split into hexagon 54376672 and 3 other hexagons that are parts of 6 hexagons in total in resolution 14. So, since the aim is not to cause an overhead, resolution 14 (see Figure 2.18) is considered as good enough to be used since only 952 total hexagons are created, which would have been 4 times more if resolution 15 was chosen and exponentially larger at higher resolutions. At resolution 14, according to the sizes of the hexagonal cells and the density of the thinned point clouds, it can be estimated that each hexagon will contain approximately 10000 to 50000 points.

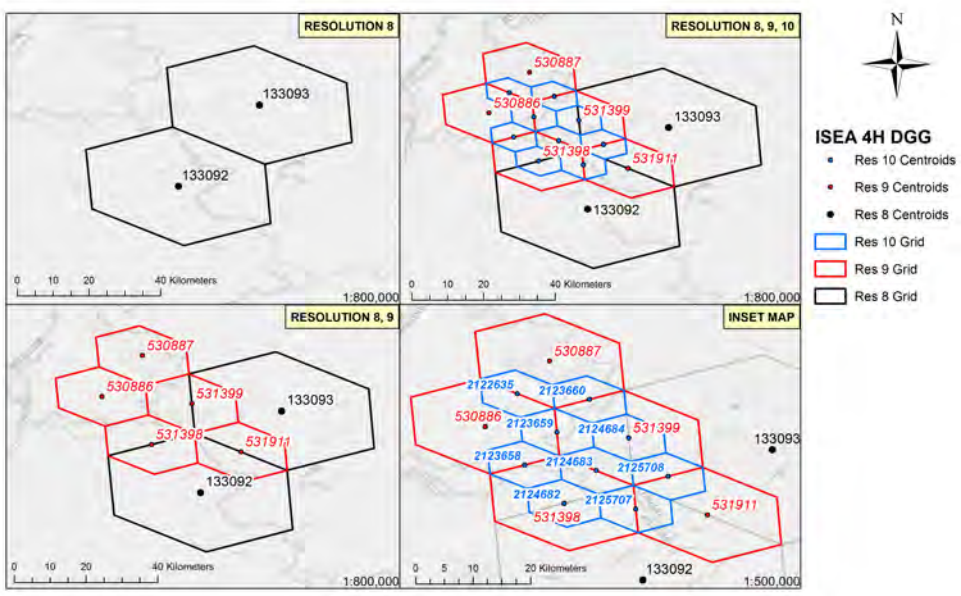


Figure 2.16: The (large) hexagonal grids of resolution 8, 9 and 10 with parameters of an icosahedron DGG created using ISEA as inverse map projection and aperture 4.

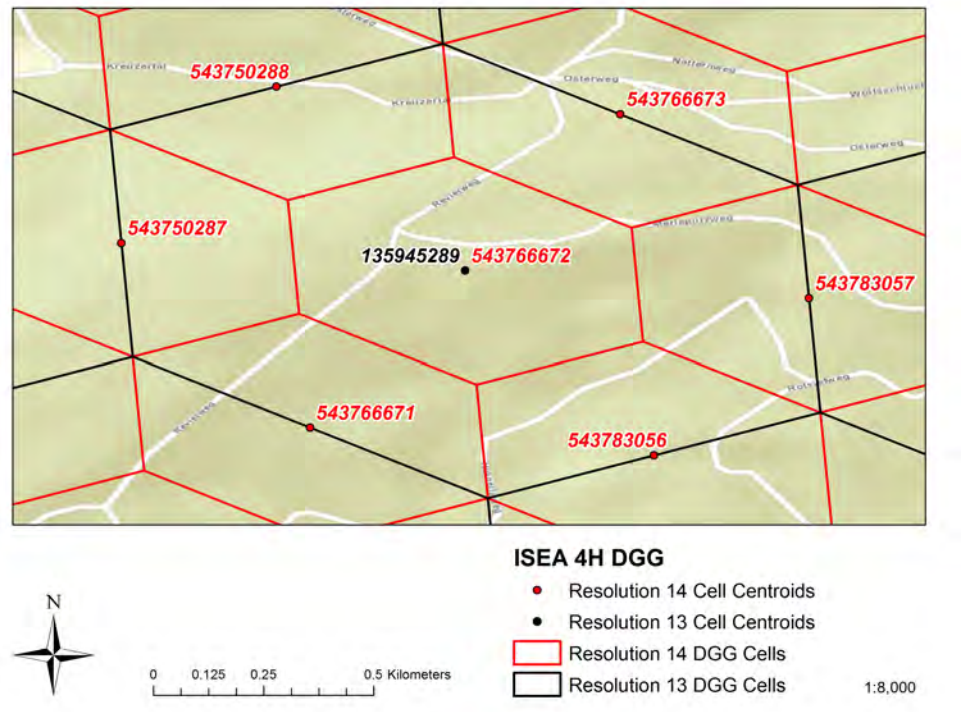


Figure 2.17: A hexagon in resolution k is split into 4 smaller hexagons in resolution k+1.

2.2.8 Indexing of the DGG

The indexing provided for the hexagons (and their centroids) using the DGGrid software (from the generation of the grids) is not hierarchical as it can also be seen in Figure 2.17. For example, the hexagon with index 135945289 in resolution 13, is subdivided into hexagons with indexes 543750288, 543766673, 543783057 and so on. Therefore, the parent and child hexagon ID's do not have any relation to each other. Ideally, according to a proper hierarchical indexing scheme, the ID's of the child hexagons should be an extension of the ID's of their parent hexagons.

The relation that exists in the indexing is only between hexagons at the same resolution. As one can see in Figure 2.18 the numbering is sequential and increases diagonally from left to right. The numbers of the hexagons are quite big since they are part of a global indexing scheme, which occurs because the *output_cell_label_type* is by default set to *global_sequence* (Sahr, 2015).

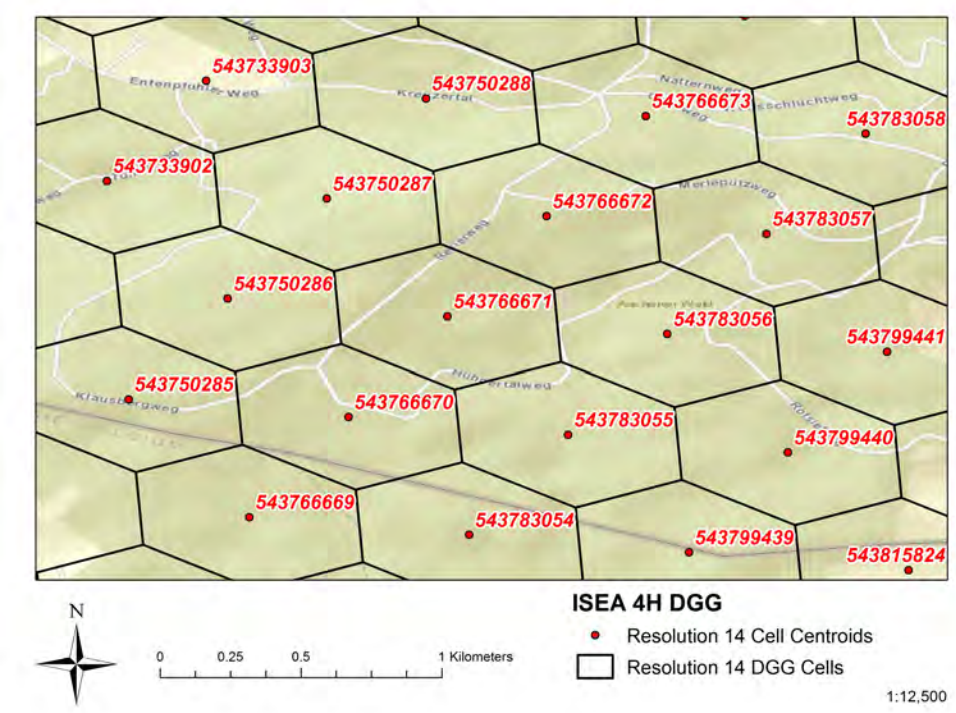


Figure 2.18: The hexagonal grids of resolution 14 (icosahedron, ISEA, and aperture 4 generated by the DGGRID software) and their indexes, which are sequentially numbered according to a global indexing scheme.

Other capabilities of the software related to indexing methods can be tested to figure out if there is a fast way to create the hierarchical indexing since this is ideally desired, but it is a 'should have' and not a 'must have' requirement. To do so, the operation *transform_points* can be assessed, of which the main aim is to assign points to the cells of the grids. There are though 7 alternative methods for its implementation. Their difference lies in the way the location ID of the point is stated. The Table 2.4 shows the 7 methods, a short description of them and the delivered output once these methods are executed using the existing latitude, longitude and height values in WGS84.

Three of these methods *(plane, projtri, vertex2DD)* do not provide hexagon indexes; they use other ways to distinguish where a point is placed (e.g. triangle number and (x,y) coordinates on the triangle). The *Seqnum* method identifies the hexagons with sequential numbering, the methods *Q2DI* and *Q2DD* generate indexes based on quads and not on the subdivision cells. It was also not possible to apply the method *interleave* (although it uses quads).

	Q2DI	Seqnum- linear address	Interleave	Q2DD
Description	Quad number and (i, j) coordinates on that quad	Linear indexing address from 1 to the size of the DGG	Digit-interleaved form of Q2DI	Quad number and (x, y) coordinates on that quad
Output	4, 71, 49, 252.649	214835, 252.649	Value 'INTERLEAVE' is not one of the allowed values: geo, q2di, seqnum, plane, q2dd, projtri, vertex2dd, aigen	4, 0.1829835, 0.1672156, 252.649
	PLANE	**PROJTRI**	**VERTEX2DD**	
Description	(x, y) coordinates on unfolded ISEA	Triangle number and (x, y) coordinates within that triangle on the ISEA plane	Vertex number, triangle number, (x, y) coordinates on ISEA plane	
Outpu	4.2363047, 1.6571903, 252.649	8, 0.7636953, 0.0748605, 252.649	4, 8, keep, 0.1829835, 0.1672156, 252.649	

Table 2.4: Comparison of methods for the identification of a point's location in the DGGS by using the DGGRID software.

The next and last option could be to use another *dggs_type* called *Superfund_500m*. According to Sahr et al. (2003) this method generates icosahedral hexagonal DGG with approximately 500 meter distance between neighboring hexagonal cell centers and uses hierarchical indexing. It uses the Fuller (Dymaxion) projection and it is a mixed aperture sequence (aperture 3 and 4). Although this projection does not provide equal area cells (like ISEA does), the Superfund method was tested to find out if it provides appropriate results with prospects of modifying the source code if needed, since the DGGRID software is not proprietary. In Figure 2.19 the generated hexagons and indexes in resolutions 2, 4, 5 and 6 using the DGG type *Superfund_500m* are given. When moving from resolution 2 to 4 it seems that the method works properly as there is a hierarchical relation between the indexes. In resolution 2 the hexagon has index 2028 and in resolution 4 the hexagons have indexes 202886 and 202871. Therefore, the first four digits of these 3 numbers are the same and every time a resolution is created, a digit is appended to the right side of the number. However, there is no relation between the hexagon with index 202871 in resolution 4 and its children in resolution 4 (2028855,2028856, 2028854). The same holds true for the children of the parent hexagon 2028856. A tree explaining the parent-child relationship and the seemingly non-hierarchical indexing is provided in Figure 2.20.

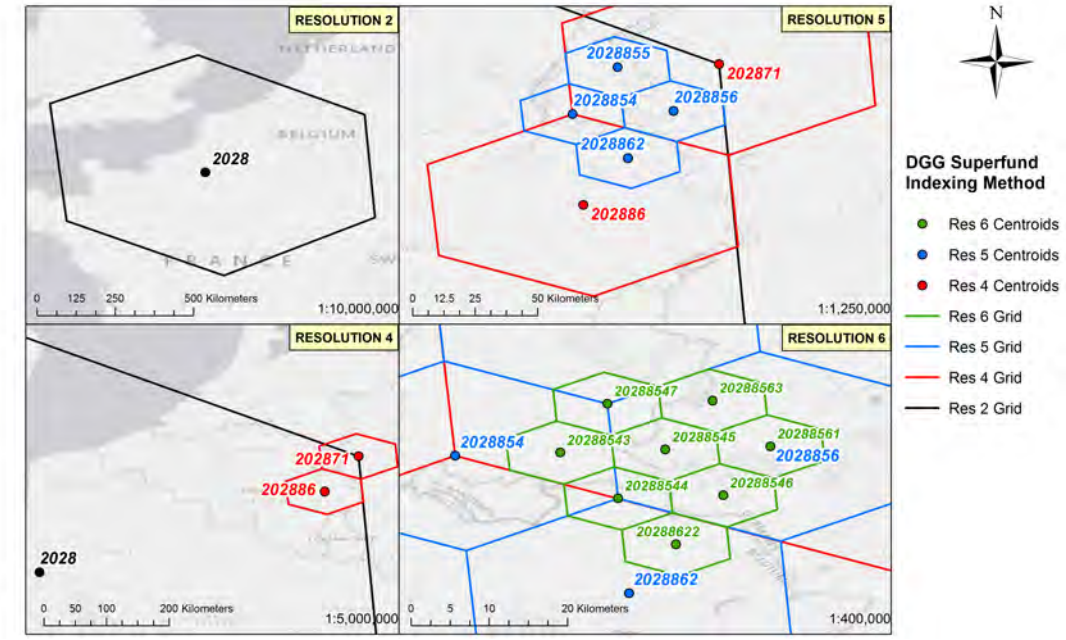


Figure 2.19: Relation between the indexes of the hexagons in resolutions 2, 4, 5 and while using the DGG type *Superfund_500m*.

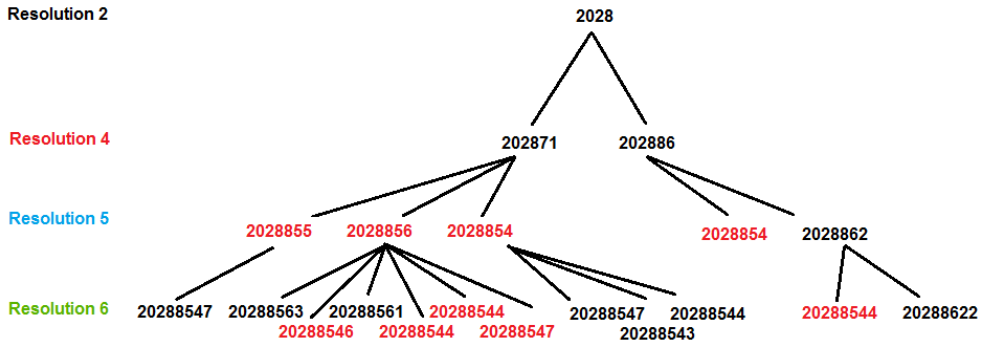


Figure 2.20: Parent - child relationship showing the non-hierarchical indexing obtained using the method *Superfund_500m*.

2.3 DATA PROCESSING

2.3.1 Software

Different type of software were used to transform the data from their initial coordinate reference system and datum to the ETRS89 datum in order to compare and choose the most appropriate one for the research. Brief descriptions of each of the software used are provided below.

PROJ. 4: Proj.4 is a coordinate conversion library which converts geographic (longitude and latitude) coordinates into Cartesian coordinates (and vice versa). It uses a wide range of selectable projection functions. The user is also able to download separate grid files that contain the datum transformation information, in case they are needed. The *pyproj* Python library can be used to work with most of the Proj.4 functions.

FME: Feature Manipulation Engine (FME) is a premier data interoperability tool that can harmonize data from hundreds of data formats. Moreover, it can handle data from different coordinate systems or combine datasets with those in other coordinate systems. Usually, spatial datasets will have a coordinate system defined, but if that is not the case, then FME can be used to assign the proper coordinate system to the dataset. The re-projection itself is not an easy task as it a complex mathematical operation and can also produce slightly different results for the same re-projection. For this reason, FME has two major re-projection transformers.

The 'CsmapReprojector' transformer uses the CS-MAP re-projection engine to do horizontal and vertical transformations. CS-MAP is a general re-projection engine that is included with FME and includes definitions for thousands of coordinate systems, with a large variety of projections, datums, ellipsoids and units.

The 'Reprojector' transformer is a simpler version of the CsmapReprojector transformer. The main difference is that it is meant for horizontal reprojection only. Therefore, as it does not allow the conversion of vertical coordinate systems, the 'CsmapReprojector' transformer was used for the purposes of this research.

PCTRANS: PCTrans is provided for free by the Hydrographic Service of the Royal Netherlands Navy. PCTrans is a program for geodetic and hydrographic calculations. The program can be used for the computation of datum transformations, various map projections, geodetic lines, rhumb lines, and surface areas while taking into account the curvature of the Earth. It also contains the correction grid for the Netherlands, and this make it more accurate than many GIS software such as QGIS that do not take into account the correction grid when transforming data from the Amersfoort datum (used in the Netherlands) to another datum.

Many different map projections are in use all over the world for different applications. However, having many different types of map projections and grid coordinates, may sometimes also result in confusion about what coordinates are actually used or given. Some software packages may support many of these map projections, but it is impossible to support them all.

Other software are specifically written for one specialized map projection, and give incorrect results when using coordinates from a different type of projection. By comparing some of the available software we can conclude which performs the best for our particular use case- the Three-Country Point. This does not mean that is the best option in general, only that for our particular use case it is the best one to use. The results of this testing are provided in the next section.

2.3.2 Software comparison

To determine the appropriate software to perform the coordinate conversion, a number of tests on a small dataset were carried out. The goal is to compare the original coordinates of a point to its coordinates after being converted into two systems: 1) EPSG 5555 (a compound datum consisting of a horizontal coordinate system (UTM Zone 32 North) on the ETRS89 datum, along with a vertical coordinate system on the German DHHN92 datum) and 2) EPSG 4258 (geographic coordinates a.k.a. latitude/longitude on the ETRS89 datum along with the ellipsoidal height). Comparing the transformed coordinates with the original input coordinate reference system using various software will allow us to determine which is the better software to perform the coordinate conversion.

From each country's point cloud data, a sample test point with X, Y, and Z coordinates in the respective country's original coordinate reference system was chosen. Then, three different programs and/or programming tools were used to perform the conversion: 1) Proj.4 along with the pyproj Python library for working with Proj.4 2) PC Trans 3) and FME. In the case of Belgium, an additional software package named CConvert was used to perform as well the conversion.

In general, it can be observed in the table in the Appendix that there are only minute differences in the X and Y output coordinate values among the the different software. However, we can conclude that Proj.4 is the better software solution because 1) it can work directly with LAS/LAZ files through the use of laspy (a Python library to work with LAS files) and pyproj 2) therefore, it can be automated with scripts 3) it supports most of the needed datum transformations without the need to download separate grid files 4) it is open-source and 5) and most of the existing GIS software packages (such as QGIS) utilize it for performing their conversions. Although there are many other software/tools that can be used to perform the conversion, only the above four were tested for the purpose of this research.

2.3.3 Coordinate Conversion

In order to use the point cloud datasets of the Netherlands, Germany, and Belgium in a DGGS, they need to transformed into a common CRS based on a common datum. The chosen CRS is WGS84, which is explained in Section 2.2.6. The whole transformation is visualized from beginning to end in Appendix F.

Several tools were used to perform the coordinate transformation. Firstly, a toolset to read and write LAS(.las) files is needed; laspy was used to

perform these tasks. Pyproj can be used to work with Proj.4 functions through Python. Numpy can be used to store the converted X, Y, and Z coordinates in the form of arrays. The combination of the three allows to read in the points from a LAS file, perform the coordinate transformation, and write the results to a new output LAS file.

FME and ArcGIS Desktop 10.3 were used to perform the pre and post processing. Due to the immense size of the original LAS files, thinning the files made sense to save time, computer processing, and computer memory. Each of the original tiles of 1000 meter X 1000 meter was thinned to contain 100000 points using FME. However, even with 100000 points per tile, the coordinate transformation took hours to run. A small test of the time taken to run for successively large number of points was carried out. More specifically, the amount of time taken to convert 100, 1000, 10000, and 100000 points at a time was recorded. The results, for the German dataset, are shown in Table 2.5.

Number of points	Time taken to convert
100	Less that a second
1000	Less that a second
10000	4 minutes
1000000	7 hours, 58 minutes

Table 2.5: Summary of the time taken to convert successively larger numbers of points shows that the relationship is not linear

The results indicate that the relationship between the number of input points and the amount of time taken to convert these points to another CRS using the Python script is not linear at all. In fact, for relatively small to medium numbers of points, the conversion is instantaneous, whereas for large number of points the conversion can take many hours. Hence, the tiles of 100000 points were broken down into separate tiles of 10000 points each, saved as a LAS file, and then the coordinate transformation was performed on those tiles. This allowed for quick response, fast conversion, and less utilization of computer power and memory. Then, the resulting tiles of 10000 points each in LAS format were merged into the original common tiles of 100000 points each.

A sample Python script that performs the coordinate transformation for Belgium is provided in the Appendix. An important thing to note in the script is the 'outfile.header.scale' parameter that controls the precision of the output coordinates. The precision is specified in the form of an array containing the X, Y, and Z coordinate precision. This is simply the number of digits to the right of the decimal point that will be present in the output converted file. X and Y have been set to a precision of 6 digits to the right of the decimal, whereas Z has been set to 4. If X and Y were set to 4 as well, then the output points would appear as a regular grid as the digit at the 4th place to the right of the decimal represents the seconds that are contained within a degree. Therefore, the precision of the original coordinates would be lost as the points would essentially 'snap' to a regular grid. To preserve the irregular nature of the input points, the precision of the X and Y coordinates was set to a higher number, 6. As for the height, the precision does not matter, as the height is expressed in units of meters and not degrees and 4 is a relatively good precision for the height. The figures below illustrate this issue with the precision of the coordinates.

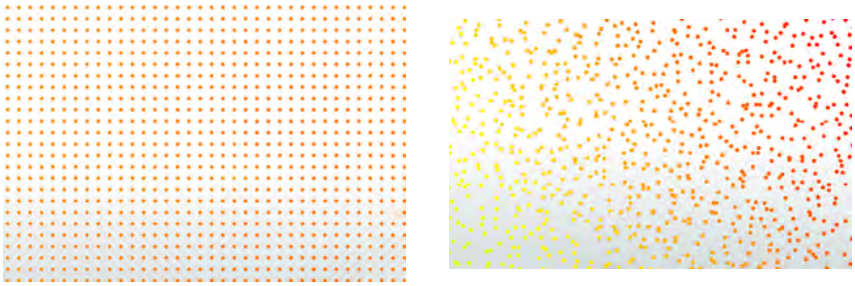


Figure 2.21: Using a scale (precision) value of 4 snaps the points to a regular grid (left), whereas using a precision value greater than 4 (in this case 6) keeps the irregular nature of the points in the point cloud (right)

The Python script writes the converted coordinate values to a new LAS file, but does not assign the output file a CRS. The correct CRS to use is EPSG: 4326, referring to WGS84. Therefore, the output files need to be 'stamped' with their proper CRS and this was done with the las2las command found within LASTools.

The tables below provide some statistics about the three datasets after conversion to WGS84. Please note that a comma is used as a separator character instead of a dot.

Dataset	Point Count	File Size
Netherlands	2620267	52kb
Germany	4842644	96kb
Belgium	4827453	96kb

Table 2.6: Summary of the LAS files of the WGS84-transformed datasets

	Degrees, WGS84				*Meters, WGS84*	
Dataset	**Xmin**	**Xmax**	**Ymin**	**Ymax**	**Zmin**	**Zmax**
Netherlands	5.9541	6.0293	50.7519	50.801	167.8024	368.2524
Germany	5.9618	6.0537	50.7118	50.8562	118.58	341.07
Belgium	5.9523	6.1255	50.7209	50.7758	177.08	299.2328

Table 2.7: Spatial statistics about the WGS84-transformed point cloud datasets

2.4 LOCAL COORDINATE CONSTRUCTION

WGS84 is an ideal CRS to store global point clouds, since it has worldwide coverage. The hexagon grid, that is defined with the DGGS software, has its boundaries displayed in this system. Only clipping the point cloud on the hexagons would be of no use; that would simply be a method of tiling. The main disadvantage, as described in Section 2.2.1, is that there would still be no easy way to visualize the data. All the points need to be converted to ECEF coordinates in order to be visualized. For big point cloud data, doing this on-the-fly would take too much time, since every point has to be converted separately. Another option is storing the whole point cloud

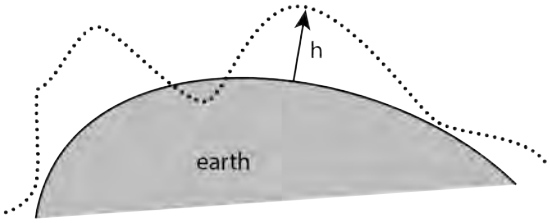
in ECEF coordinates, but this gives issues with either the digital precision, since it has to store up to 7 digits per coordinate for meter level resolution or with the storage of data which would cause considerable overhead and memory. A proper way to reduce storage, and to keep a way to visualize the data, is to define a three-dimensional coordinate transformation for each hexagon, that constructs a *local coordinate system* defined on each hexagon. A transformation is conformal when the angles between point A, origin and point B are preserved and distance ratios are preserved as well. In this way the shapes are not changed due to the transformation (Marel, 2016, p. 9).

The transformation is based upon 7 different parameters: 3 translation parameters, 3 rotation parameters and a scale parameter (Marel, 2016), since the hexagons are very small. In our case the transformation projects points from ECEF coordinates to a plane related to the centroid of the hexagon. It uses this centroid as the origin of the coordinate system. The local x and y are respectively to the east and north direction and in meter units. The local z-value is the height in meters orthogonal above the plane of the coordinate system. In this way, the height above the ellipsoid equals the height of the hexagon above the ellipsoid plus the height above the hexagon. In this way, the data that is stored can differ a few centimeters from the "true" height, but this implies that there is no distortion in the data that is visualized (see Figure 2.22).
There are a number of steps that need to be done, in order to construct the transformation parameters and get the local coordinates. The ECEF X,Y,Z coordinates of the centroid are together comprise the *shift* to the local coordinate system. The rotation parameters are derived from the latitude (ϕ) and longitude (λ) of the centroid (see 2.2) (Marel, 2016). To convert all points that are clipped within a hexagon to local coordinates, the shift needs to be subtracted and the result needs to be multiplied by the inverse of the rotation. See 2.1 and 2.2. The conversion takes on average 15 seconds for each hexagon, with outliers to 5 minutes (of a degree) for the largest hexagons and less than 1 second for the smallest.

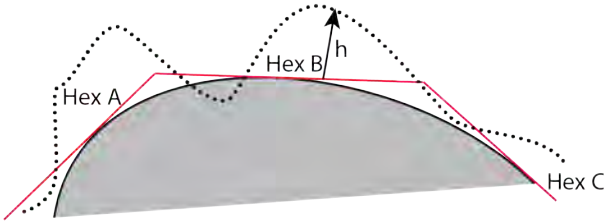
$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} X_{point} \\ Y_{point} \\ Z_{point} \end{bmatrix} - \begin{bmatrix} X_{centroid} \\ Y_{centroid} \\ Z_{centroid} \end{bmatrix} \tag{2.1}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} * \begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix}^{-1} \tag{2.2}$$

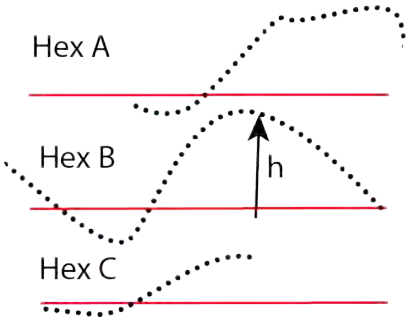
The output is a point cloud per hexagon with coordinates ranging from -250 - 250 x/y (which is about the diameter of the hexagon) and with the height that is about the same as the original height. If the hexagons are smaller, the range of the local coordinates will also be smaller, but the overhead (transformation parameters per hexagon) will be larger. Therefore the computation for viewing the point cloud will be slower. To visualize the point cloud correctly, the plane of each hexagon needs to be multiplied by the rotation and the shift needs to be added again. The use of local coordinates, therefore, allows for more efficient storage and this provides one of the strongest advantages of using a DGGS to store global point cloud data.



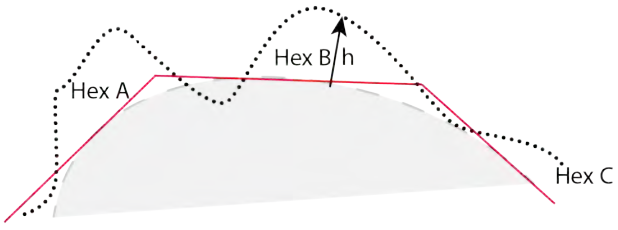
(a) Height ϕ, λ, h



(b) Clipping on hexagons



(c) Coordinates referring to hexagons



(d) No distortion when displaying in ECEF

Figure 2.22: constructing local coordinates

3 DATA ANALYSIS

3.1 INITIAL FINDINGS

Now, all three datasets' horizontal CRS have been converted into geographic coordinates (latitude/longitude in decimal degree format) and their vertical CRS converted into ellipsoidal height on the WGS84 ellipsoid (datum). There are areas in which two or all three of the datasets overlap one another. These areas can now be studied for any discrepancies in X, Y, and Z coordinate values stemming from the three datasets. ArcGIS Desktop, FME and LAStools were used to perform most of the analysis and visualization of the overlapping areas as they provide a diverse set of geospatial tools to work with both LIDAR LAS files and rasters. Three cases need to be analyzed, one per country pair: 1) Netherlands and Germany, 2) Netherlands and Belgium, and 3) Germany and Belgium.

In order to get a first impression regarding the discrepancies of the data in the 3rd (Z) dimension, the point cloud datasets are converted to raster Digital Elevation Models (DEM's) and then the WGS84 heights are classified in 8 classes, almost all of which are equal interval. As it can be seen in Figure 3.1 the change in the heights is not at all gradual where the point clouds meet (i.e. the boundaries of the LAS files). If there were no (or minor) differences in the heights of the overlapping areas between the datasets and if the point clouds from the different countries were perfectly aligned, then there should not be such abrupt changes at the point cloud boundaries. For example, there is a sudden change in elevation value from the range 161-200 meters to the range 201-240 meters at the Dutch-German border. There is also an almost instantaneous change in elevation values at the German-Belgian dataset border, from the range 201-240 meters to the range 281-300 meters, respectively.

Working with the LAS files directly would consume a significant amount of processing time, resources, and computer memory. To compute differences in height values between the various datasets, a natural approach would be to convert the LAS file height values into raster Digital Elevation Models (DEM's). A DEM is a digital model or 3D representation of a terrain's surface created from terrain elevation data (Wikipedia, 2017). A DEM representation lends itself naturally as a model to store height values. Each cell contains one value, and the value is the average of the height values of all points that fall within the cell. Depending on the chosen cell size, the quality of the DEM can be controlled. For example, choosing a cell size of 5 meter by 5 meter (as was done in this research, see below) would result in a coarser representation of point cloud heights than a cell size of 1 meter by 1 meter. However, given the objectives of this research, it is deemed sufficient to compute in general terms the Z differences; therefore, the choice of a 5 meter by 5 meter cell size is sufficient. Then, by a simple subtraction of heights of cells, the differences in height values can be obtained. FME

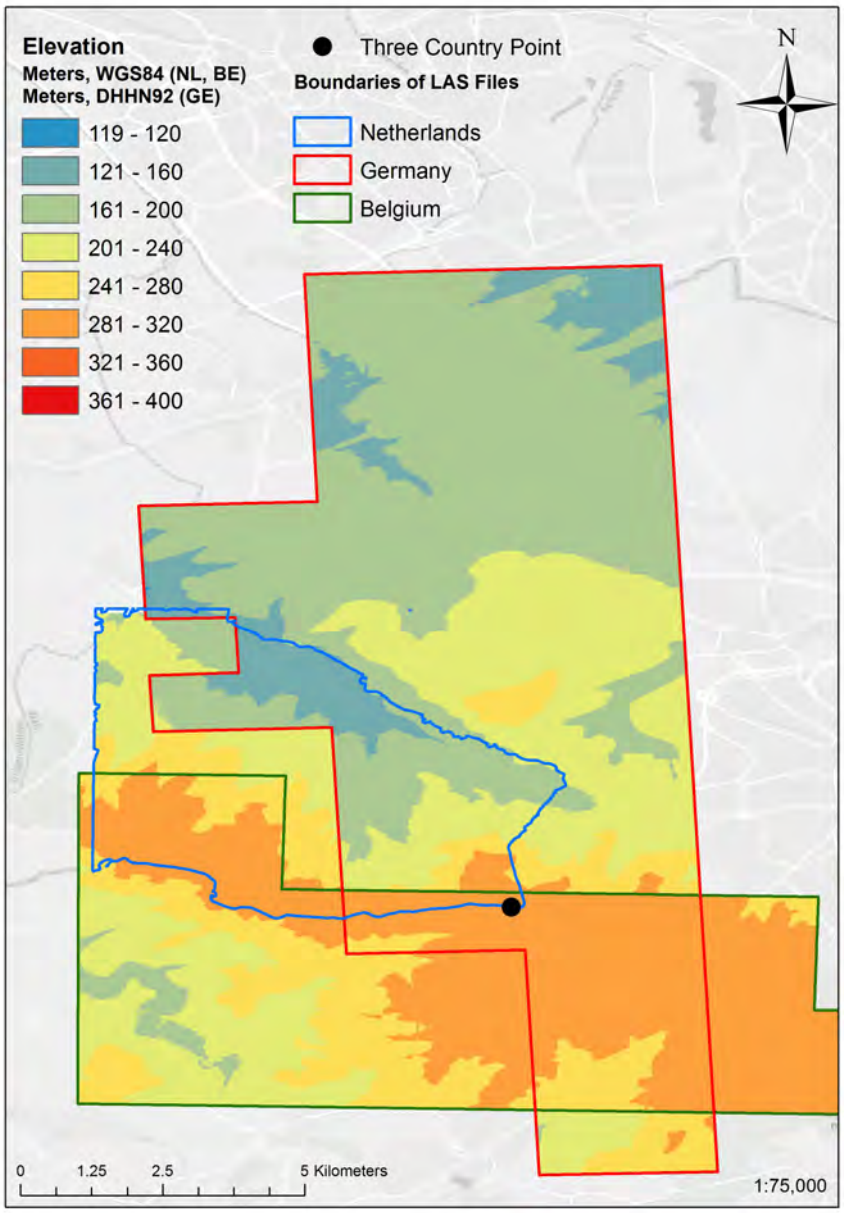


Figure 3.1: An overview of the DEM's created using the point cloud datasets of Germany, Netherlands and Belgium and classified according to their heights in 8 classes. It can be easily seen that the heights between the datasets do not change gradually, which prompts the investigation of these height differences.

can be utilized to perform these tasks. First, since the LAS files are in a geographic coordinate system (i.e. latitude/longitude), in order to generate a raster DEM they need to be projected into a projected coordinate system using the *CsmapReprojector* transformer. This is because FME cannot directly create a DEM from the unprojected LAS files. The projected coordinate system chosen was *Universal Transverse Mercator (UTM)* Zone 32 North on the WGS84 datum. Then, the *RasterDEMGenerator* transformer can be used to perform a planar barycentric interpolation of the input points and the result resampled to a 5 meter by 5 meter cell size raster grid. Resampling refers to changing the spatial resolution of a raster dataset and usually involves transforming the dataset from a finer to a coarser resolution. Finally, the result is written to an output GeoTIFF file in the WGS84 UTM Zone 32N coordinate system/datum. This will result in three separate raster DEM's of cell size 5 meter by 5 meter, one for each country's LAS dataset.

Now, to compare and contrast the datasets with one another, their overlapping areas need to be delineated. The *Raster to Polygon* tool within the *Conversion Tools* in ArcGIS Desktop can generate a vector polygon layer from a DEM. Then, the *Intersect* tool can be used to find the spatial intersection of any two polygon shapefiles, and the result saved to a new output shapefile. This will result in three separate shapefiles, one for each country pair's area of overlap. Then, the original raster DEM's can be clipped on the vector overlapping areas using the *Clip* tool by clipping the raster to the vector geometry. To find the exact amount of overlap between the datasets, the *Calculate Geometry* operation can then be performed within each shapefile. The areas of overlap among the three countries' datasets are shown in Table 3.1.

Country	Area Of Overlap (Square Meters, UTM)
Netherlands and Germany	10,059,900
Netherlands and Belgium	5,021,127
Germany and Belgium	9,804,271
All countries	656,169

Table 3.1: Area of overlap between the point cloud datasets of the three countries

Now, discrepancies between the overlapping raster DEM's of the three countries can be analyzed. A simple *Minus* operation can be performed to subtract one DEM from another DEM on a cell-by-cell basis. However, the input DEM's are not cell-aligned. That is, each 5 meter by 5 meter cell in one DEM does not align precisely with a 5 meter by 5 meter cell in the other DEM (the cell borders do not align). In ArcGIS Desktop, this can be fixed easily by setting a *Snap Raster* environment; in this case, the Netherlands DEM was used as a 'Snap Raster' to snap the other dataset's cell corners to those of the Netherlands dataset. Once this is done, one of the two DEM's can be subtracted from the other on a cell-by-cell basis. The output of the *Minus* operation is then saved as a new raster layer. Three such *Minus* operations were performed, one for each country pair: 1) Netherlands minus Germany (shown in Figure 3.2) 2) Netherlands minus Belgium (shown in Figure 3.3) and 3) Germany minus Belgium (shown in Figure 3.4).

Histograms of each of these Z-difference maps are provided below. These give a glimpse into the distribution of the Z-difference values between the datasets, and allow us to know where most of the values lie. The X-axis rep-

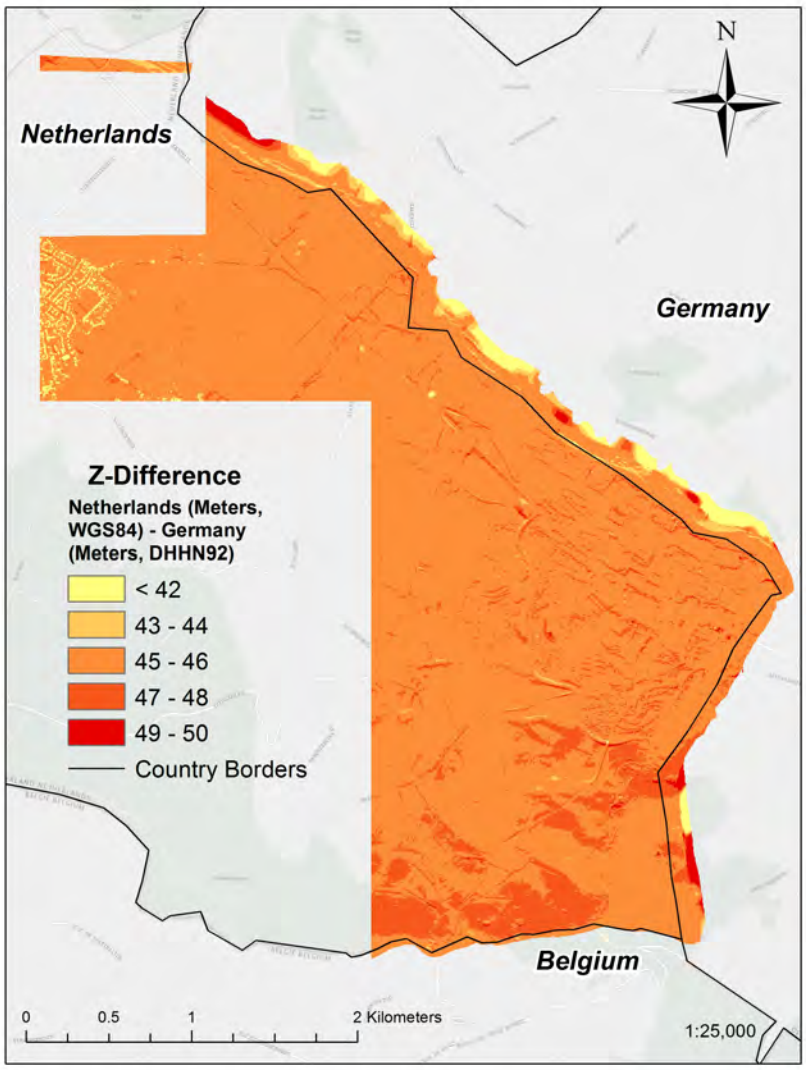


Figure 3.2: The Germany DEM subtracted from the Netherlands DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. This area falls almost entirely within the extent of the Netherlands.

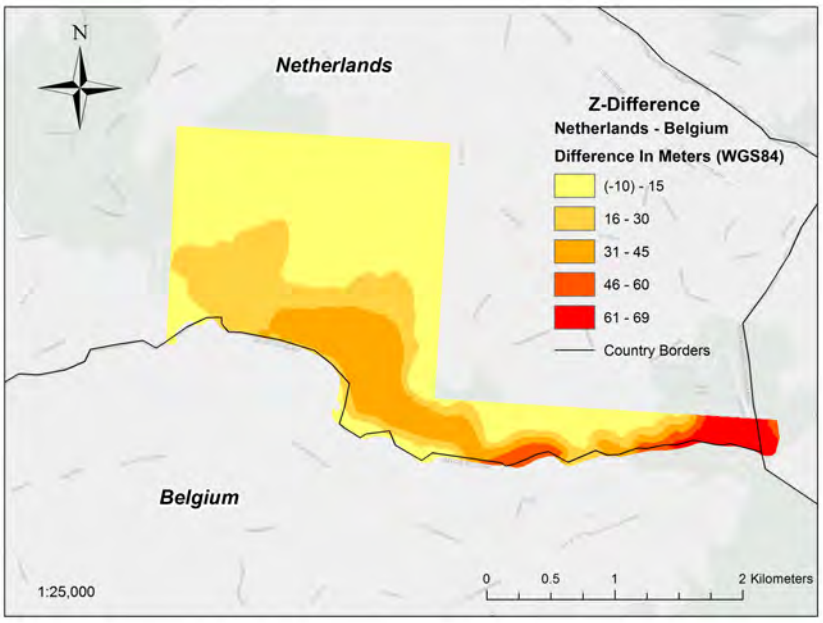


Figure 3.3: The Belgium DEM subtracted from the Netherlands DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. This area falls almost entirely within the extent of the Netherlands.

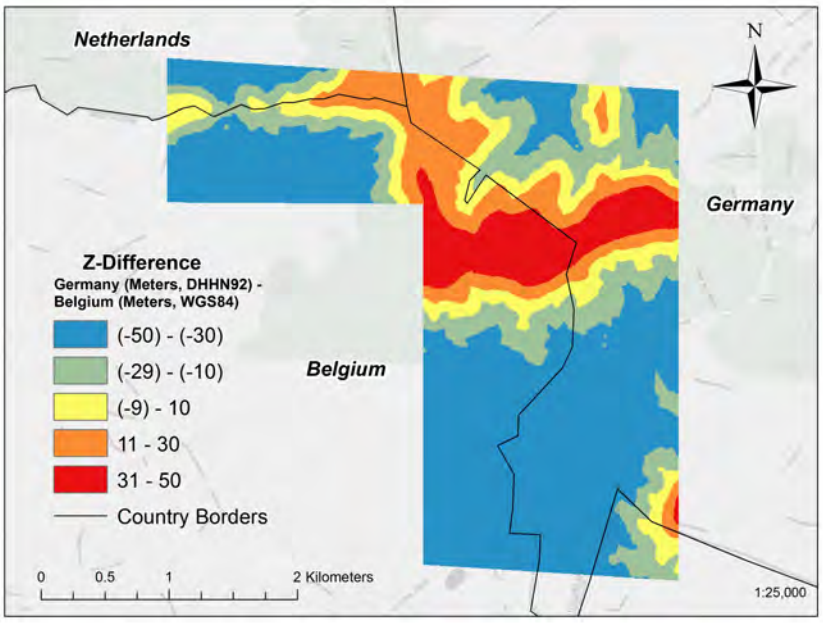


Figure 3.4: The Belgium DEM subtracted from the Germany DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets.

resents the Z-difference value and the Y-axis represents the number of raster cells having this difference value. For example, most of the Z-differences between the Netherlands and Germany fall in the 45-46 meter range; for Germany and Belgium, in the (-50) - (-42) meter range; and for the Netherlands and Belgium in the (-10) - 2 meter range. The histograms indicate that for the Netherlands and Germany, there spread of the Z-difference values is minimal, as more than 90 percent of the values are contained within the 45-48 meter range. For the other two cases, however, there is considerably more spread in the difference values: deviations of up to 42 meters can be found between the Germany and Belgium datasets (see Figure 3.7), and up to 69 meters between the Netherlands and Belgium datasets (see Figure 3.6). Potential causes of these large deviations are explained in the next section.

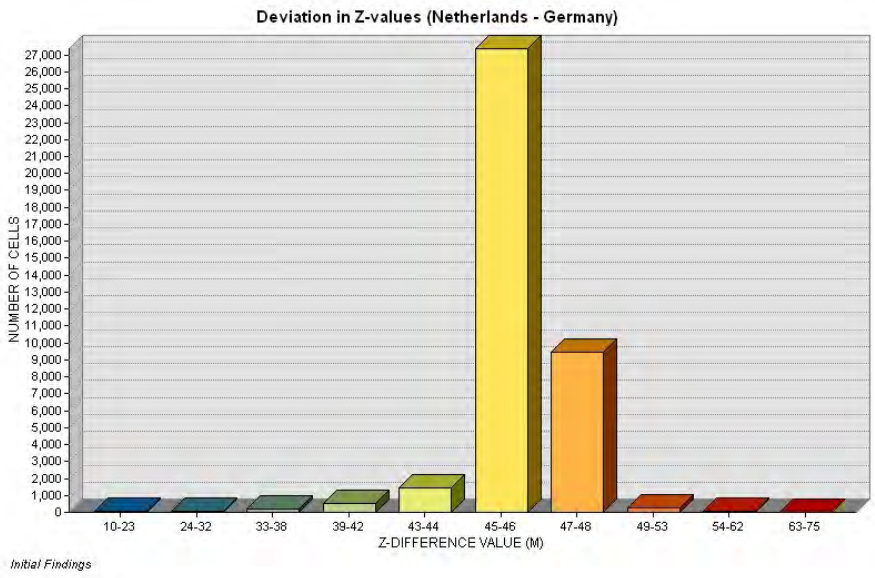


Figure 3.5

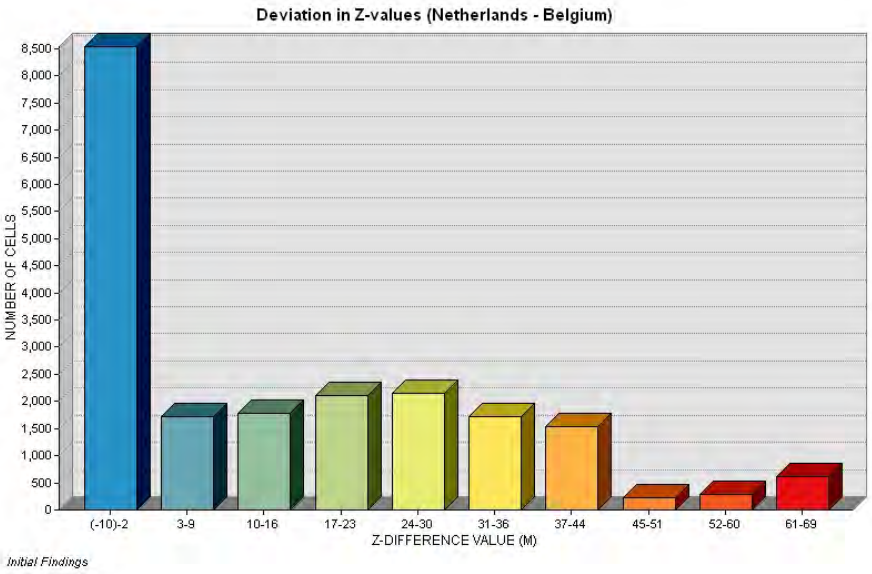


Figure 3.6

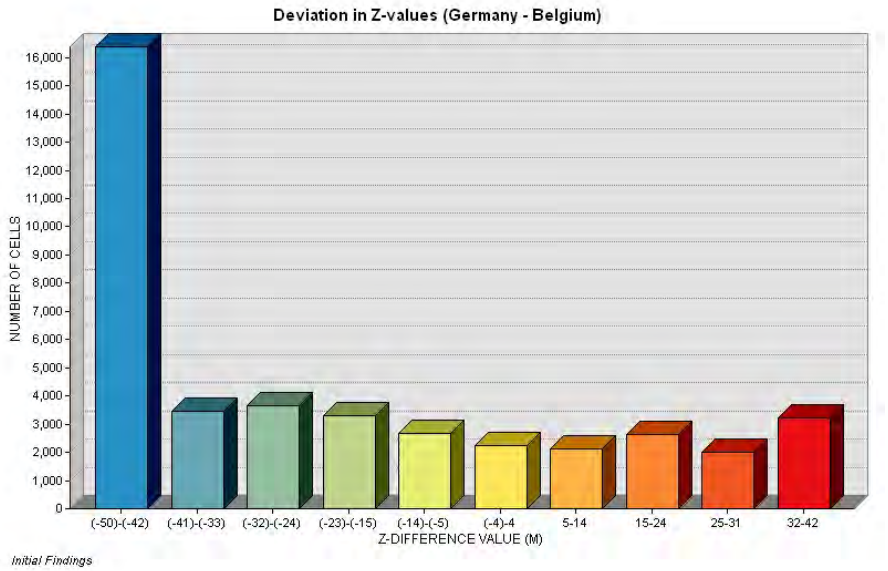


Figure 3.7

An effort will be made to explain the potential causes of inconsistencies between the point cloud datasets originating from the different countries and overlapping at a common geographic area. The maps are meant to provide a visual overview of the discrepancies between the point cloud datasets. An explanation follows, for each country pair at a time.

3.1.1 Netherlands and Germany

Z-Deviation

It can clearly be seen that the German dataset is consistently off from the Netherlands dataset by about 45-46 meters almost throughout the entire overlapping area (see Figure 3.2). Why this is the case is a matter of investigation. After much research, it can be concluded that this is due to the fact that Proj.4 did not take into account the conversion from the German vertical CRS (*DHHN92*) to the WGS84 ellipsoid. This conversion in vertical CRS is related to a concept known as a *quasigeoid*. Unlike a geoid, which is an equipotential surface (a surface of constant gravitational potential), the quasigeoid is not an equipotential surface of the Earth's gravity field and thus has no physical meaning ((Heiskanen and Moritz, 1967)). Computing the quasigeoid is easier than computing the same with the geoid because the former requires no prior knowledge of the density of the mass distribution of the Earth ((Sadiq and Ahmad, nd)) , unlike the latter.

The German combined quasigeoid is the height reference surface of the Working Committee of the Surveying Authorities of the provinces of Germany. It can be used to transform ellipsoidal heights determined by GNSS measurements on the ETRS89 datum (GRS80 ellipsoid) into physical (geoidal) heights determined by leveling and vice versa (Federal Agency for Cartography and Geodesy). The geoidal heights are based on the *Deutsche Haupthöhennetz (DHHN)* vertical CRS used in Germany. As the German point cloud dataset's vertical CRS is based on a realization of the DHHN datum (namely *DHHN92*, it too can be transformed into the correct ellipsoidal

heights in ETRS89. The formula to convert height measurements between the two systems is:

$$H^{DHHN} = h^{ETRS} - \zeta^{ETRS}_{DHHN}$$

where ζ^{ETRS}_{DHHN} is the ETRS89 ellipsoid – DHHN2016 quasigeoid separation. In other words, this value represents the vertical distance between the quasigeoid (on which the DHHN2016 system is based) and the reference ellipsoid (ETRS89) and is known as the *quasigeoidal height* or *height anomaly* ((Vanicek et al., 2012)).

For the quasigeoid to have some use in practice, it has to have a meaningful system of heights associated with it. These heights are known as *normal heights* and refer to heights of a topographical surface above the reference quasigeoid, measured along the normal plumbline ((Vanicek et al., 2012)).

The *DHHN* is a precise network of measured heights in Germany. It has had many "realizations", such as DHHN85, DHHN92, and DHHN2016. The German point cloud data is based on DHHN92; however, DHHN92 and DHHN2016 are off only by up to a few centimeters (Wikipedia, DHHN). Moreover, ETRS89 and WGS84 are nearly identical, being off by only less than a meter (KilletSoft) in position. For the current research objectives, this is a negligible difference. Therefore, the above formula can also be used to convert between WGS84 ellipsoidal height and DHHN92 normal height. The German Federal Agency for Cartography and Geodesy (FACG) has published a map indicating the values the variable ζ^{ETRS}_{DHHN} takes across the country. This map is shown in Figure 3.8.

As can be seen, the ellipsoid-geoid separation is approximately 45-46 meters at the TCP. This explains why the German point cloud data is consistently off by this amount and lower than the Netherlands point cloud data, because Proj.4 did not take into account the quasigeoidal height in its conversion. The reason the German dataset is lower than the Dutch dataset is because the Dutch dataset's vertical CRS was correctly transformed to WGS84 ellipsoidal height, whereas the German dataset's vertical CRS was not. Subtracting the German elevation values from the Dutch values yields positive numbers, indicating that the Dutch dataset is higher than the German dataset. Once the values for ζ^{ETRS}_{DHHN} are added to each point in the German point cloud dataset, the dataset will align with the Netherlands dataset.

The German quasigeoid has been determined through many methods: 1) gravity disturbances derived from 860,000 point gravity values from various governmental and private organizations 2) a network of benchmark points surveyed using GNSS for their ETRS89 ellipsoidal heights as well as normal heights in DHHN2016 3) DTM's and 4) global gravity field models (FACG, 2016). Files containing the values that the quasigeoidal height takes at locations across the country can be obtained from the FACG in ASCII text, binary, or some other proprietary formats. However, this data is not provided free of charge, and will cost at least 250-300 euros to acquire. As this is outside the budget for this project, data acquisition will not be performed. However, a free online service to compute quasigeoidal heights is provided on the FACG website at (Anonymous, 2017) and this can be used to corroborate the results at selected points across the overlapping area.

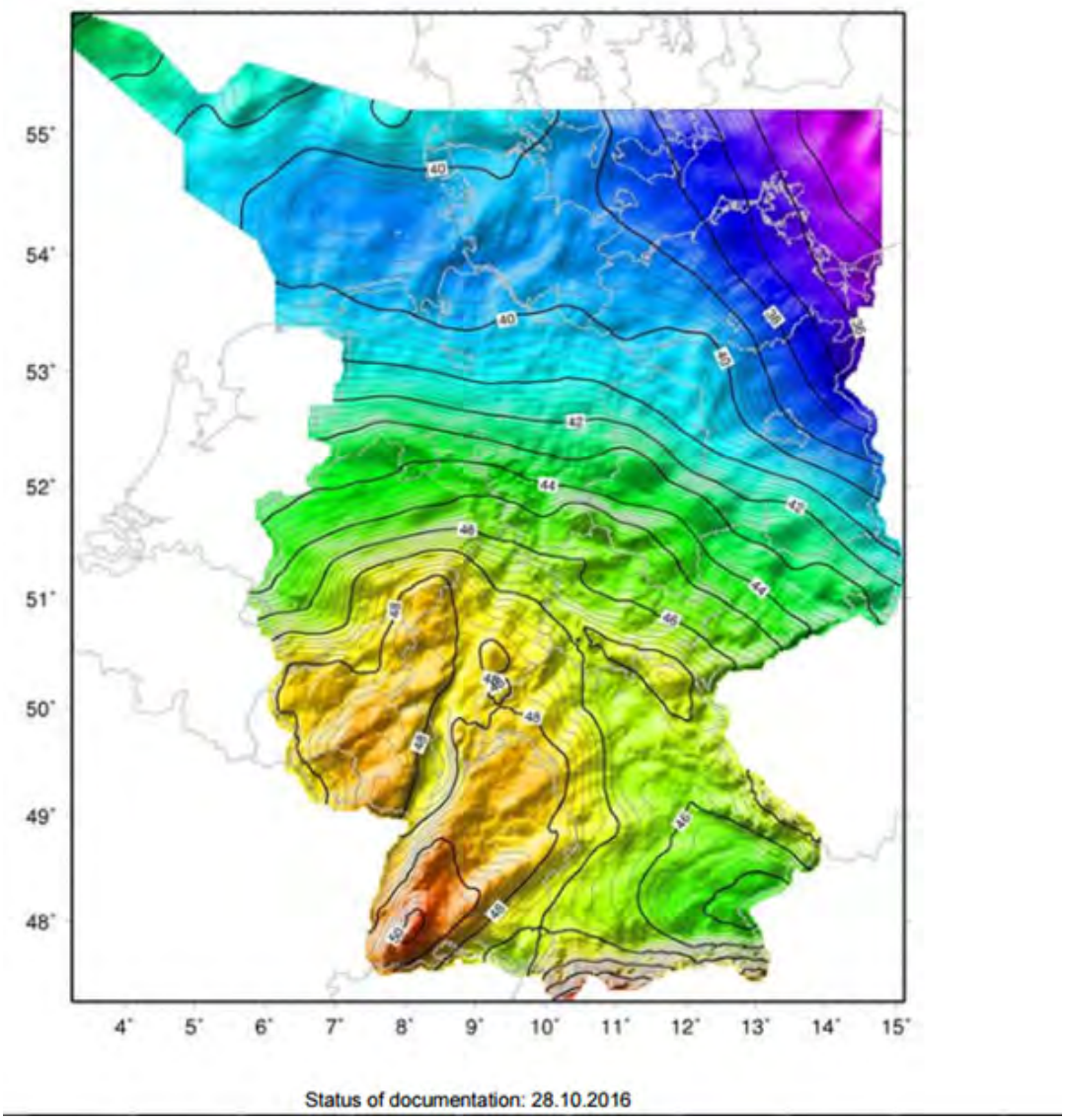


Figure 3.8: Contour line map of the quasigeoidal heights, i.e. separation between DHHN2016 and ETRS89 (Bundesamt für Kartographie und Geodäsie, 2016).

Since the online service can calculate quasigeoidal heights only at single points and does not allow us to compute these heights for in batch mode, these values will not be acquired for the entire overlapping area. Instead, about 40 random points will be tested in the overlapping area. These points are also constrained to be spread out across the area. The values of the 'difference DEM' of Figure 3.2, which was created by subtracting the DEM of Germany from the DEM of the Netherlands in the overlapping area, will be recorded at these 40 random point locations. These two DEMs resulted from the transformation of the original LIDAR data to WGS84, and its conversion into a DEM. It can be recalled that at that moment the height values (in DHHN92) from the original LAS file for Germany, did not get converted to WGS84 ellipsoidal height properly due to Proj.4's ignorance of the quasigeoidal height. The free conversion tool requires geographic coordinates (latitude/longitude) and ellipsoidal height as input. Although it requires geographic coordinates on the ETRS89 ellipsoid as input, WGS84 geographic

coordinates could also be used, since the two are considered almost identical (Killet, 2012). With these three input parameters, the tool outputs the normal height in DHHN2016 and the quasigeoidal height.

The 40 random points can be queried for their latitude/longitude coordinates, however the WGS84 ellipsoidal height at these locations is unknown. To use the tool, there is thus incomplete information (ellipsoidal height is missing). As an alternative the output that the software provides is used to compensate for this lack of data. As the tool provides the quasigeoidal heights in DHHN2016 and not DHHN92, the deviation between these two systems also needs to be taken into account. This deviation can be obtained from another online service that outputs the difference between DHHN92 and DHHN2016, after the provision of latitude and longitude values in ETRS89 (AdV, 2017). Then, the DHHN92 values from the original DEM of Germany are joined to the sampling points and the e-tool is used to acquire the difference. In order to find out if this has to be subtracted or added to the DHHN92 heights that exist to obtain the corresponding DHHN2016 heights, data for the height benchmarks of the city of Aachen (bordering the TCP) can be used (Fachbereich, 2017). In this dataset, heights have been measured in both DHHN2016 and DHHN92 and from this it can be observed that DHHN16 height values are always larger than those of DHHN92. Therefore, the DHHN2016-DHHN92 difference can be added to the DHHN92 coordinates, to obtain the corresponding height values in DHHN2016. *DHHN2016* is the newest realization of the German height reference system, and is meant to be introduced by June 30, 2017 (Wikipedia, 2017).

Then, the e-tool can be used by using the existing latitude and longitude values of the random points on the WGS84 datum and approximating the ellipsoidal height (i.e. testing various ellipsoidal height values) until the output DHHN2016 height is equal to the obtained DHHN2016 height from the previous step. Finally, the quasigeoidal heights can be obtained. However, these values represent the height separation between the WGS84 ellipsoid and the DHHN2016 quasigeoid. Since the input data are in DHHN92, the height differences between the DHHN92 and DHHN2016 systems need to be *subtracted* from the calculated quasigeoidal heights.

Next, the values of the 'Difference DEM' at the 40 random sample point locations in the overlapping area can be subtracted from their obtained quasigeoidal heights. This way, the quasigeoidal height from the DHHN92 system is taken into consideration in the observed differences in the overlapping area of the two countries. To 'depict' this:

Difference of quasigeoidal heights on sampling points = (WGS84 heights Netherlands - DHHN92 heights Germany) - (Quasigeoidal heights from online service)

The histogram in Figure 3.9 shows the frequency distribution of the differences in quasigeoidal height values at these sample point locations. It can be clearly seen that most of the sampling values give differences close to 0 m; the minimum value is equal to -0.13 m and the maximum difference value equal to 3.17 m. The point that gives the highest value is located inside the extent of Germany, but very close to the border with the Netherlands (see Figure 3.2). However, the AHN2 source data do not extend as much outside

the Netherlands' boundaries. It is possible that this big difference arises due to the fact that when creating the DEM from the AHN2, the heights of many areas along the boundary of the dataset were interpolated from the nearest points. Therefore, it is safe to assume that these values are not valid and that they should be excluded from the analysis (overlapping) area. No other points were tested since the accuracy of the online service is not known and the method followed to extract the quasigeoidal heights might include many steps that could introduce small errors in the calculations.

Therefore, it is futile to make any bold conclusions here; that could be possible only if the quasigeoidal correction grid was obtained, though as stated earlier this would cost a significant (for this project) amount of money. Regardless, it is noted that the differences in the overlapping areas between Netherlands and Germany at the sampling points decrease significantly once the quasigeoidal height is subtracted from them, i.e. once the quasigeoidal height is taken into consideration (see Figure 3.14). Most of the values that are close to om, but not precisely om, could have arisen because of the decimeter-level deviations present between the ETRS89 datum (what is required as input for the e-tool) and the WGS84 datum (what was actually provided as input to the e-tool), or due to possible interpolation that could have been performed by the e-tool or that has been performed as a part of the DEM creation from the LIDAR.

The average of the quasigeoidal values for *all* of the sampling points was 46.202 meters. These values are provided in Table 3.2 below.

Point ID	Quasigeoidal Height	Point ID	Quasigeoidal Height
1	46.117	21	46.236
2	46.130	22	46.206
3	46.141	23	46.194
4	46.121	24	46.226
5	46.139	25	46.236
6	46.137	26	46.26
7	46.133	27	46.251
8	46.152	28	46.247
9	46.151	29	46.279
10	46.143	30	46.277
11	46.158	31	46.269
12	46.144	32	46.262
13	46.158	33	46.283
14	46.153	34	46.268
15	46.174	35	46.293
16	46.150	36	46.291
17	46.166	37	46.293
18	46.162	38	46.303
19	46.195	39	46.309
20	46.187	40	46.079
AVERAGE	46.202		

Table 3.2: The quasigeoidal heights at the 40 random sample point locations

Therefore, the average of these values could be added to the original heights of Germany (in the DHHN92 system) to get the WGS84 ellipsoidal heights. Then both datasets would have a height based on a common datum. Lastly, the differences between the datasets of the two countries should be recalculated.

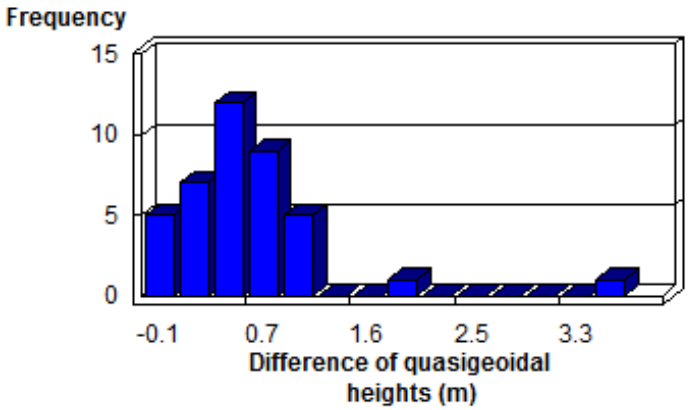


Figure 3.9: Frequencies of values that describe the *differences* between a) the quasi-geoidal height calculated by subtracting DHHN92 heights from WGS84 heights and b) the quasigeoidal height calculated using the e-tools *Onlineberechnung von Quasigeoidhöhen mit dem GCG2016* (Anonymous, 2017) and *HOETRA2016* (AdV, 2017)). These values refer only to the sample points in the overlapping area of Germany and the Netherlands.

3.1.2 Netherlands and Belgium

Z-Deviation

It is evident after looking at the deviation between the two point cloud datasets after being converted into raster DEM's in Figure 3.3 , that the Dutch and Belgian datasets align sufficiently well with one-another across most of their area of overlap. Almost all of the values of the Z-deviation in the range (-10) to 15 actually fall within plus or minus 3 meters from 0 (see Figure 3.5). This is an exceptionally good level of Z-alignment between the datasets. The datasets deviate the most from one another in the southern portions of the overlapping area, where differences in the elevation values can reach as high as 68 meters. However, it can be concluded that the Belgian dataset is not accurate enough in these areas, for reasons explained below. The Belgian dataset significantly deteriorates in height accuracy in these areas, and especially within the area 1 kilometer from the Belgian border inside the Netherlands. As the overlapping area of the two datasets is almost entirely within the territorial extents of the Netherlands, this provides a strong reason to state that the accuracy of the Belgian dataset becomes poorer in this region; the overlapping area shown in Figure 3.3 is almost completely within the extents of the Netherlands.

One of the major goals of the Institut Géographique National (NGI), the national geodetic, photogrammetry, remote sensing and surveying organization of Belgium, is to establish and maintain high precision planimetric and leveling networks across the country (NGI). Both the horizontal and vertical networks have been updated between 1980 and 2000, and an exhaustive reevaluation of the precise leveling network with 19,000 geodetic benchmarks has been completed (NGI). X, Y, and Z coordinates for each of these benchmark points in different reference systems (including the *Second General Levelling (SGL)* that is used by the Belgium point cloud data) are freely available on the NGI website (`http://www.ngi.be/gdoc/index.html?lang=fr&x=506826.00&y=6539342.00&zoom=7&baseLayer=ngi.cartoweb.topo_bw.be&layers=alti_coord,plani_coord`). Therefore,

these ground control points (GCP's) can be used to verify the validity of the Belgian point cloud dataset. As the altimetry reference of the dataset is also the Second General Levelling based on the Ostend datum (*EPSG 5710*), these GCP's can be used to ascertain that the elevations contained in the dataset are correct. The standard deviation of the altitudes of these GCP's is less than 5 millimeters (NGI), so they have exceptionally high height accuracy. The coordinates of these GCP's are given in Lambert 1972, Lambert 2008, and ETRS89 latitude/longitude coordinates. Table 3.3 compares the height values of some GCP's located within Belgium to the height values of the point cloud data at the same location. None of these points fall inside the overlapping area, they are all contained within the territory of Belgium; however, they give us a general impression of the validity (accuracy) of the Belgium point cloud data inside Belgium. Accuracy can be defined as the degree or closeness to which the information on a map matches the values in the real world (Lounge, 2011). Therefore, in the current research, this refers to the degree of closeness of the observed X, Y, and Z values of the points to their true (*real-world*) X, Y, and Z values. It should be noted, however, that *there exist no NGI benchmarks in the overlapping area with the Netherlands as well as in the overlapping area with Germany*. Moreover, there also exist no benchmarks in some large areas inside of Belgium. The validity of the data in these places also cannot be determined, unless external information is acquired.

	NGI Geodetic Benchmarks			***Point Cloud Data***	
	Belgian Lambert 1972 projection		*Ostend Height*	*Ostend Height*	*Deviation*
Point ID	**X**	**Y**	**H**	**H**	
Aa49	263394	160571	196.317	196	-0.317
Aa50	264016	160375	207.716	207	-0.716
Aa51	264802	160726	206.094	205	-1.094
Aa53.1	264453	159553	197.848	199	1.152
Aa54	264994	158981	206.233	206	-0.233
				Average	-0.2416

Table 3.3: The geodetic benchmarks in Belgium used to check the validity of the acquired GeoTIFF file created from a LIDAR survey

As is evident, the point cloud data in the form of a GeoTIFF file matches very well with the geodetic benchmarks in terms of height values. Although there are some deviations of about a meter, these can be ignored as the altimetric accuracy of the GeoTIFF file itself is on the order of 0.12 meters absolute over the whole territory of Wallonia and it has been resampled to a 1 meter by 1 meter resolution. Also, some very small parts of the territory could not be acquired using LIDAR, so the elevation values in those areas were interpolated from the surrounding values. Furthermore, many of the geodetic benchmark points are actually 20-80 centimeters higher than the ground (they are placed on the walls on the sides of buildings). Moreover, GeoTIFF file contains elevation values rounded to the nearest meter and as integers and not floating-point values. These could all be reasons that there are these minor deviations of up to a meter or more between the benchmarks and the GeoTIFF file. Finally, it should be noted that the height accuracy of the original points in the GeoTIFF file is 0.12 meters. Therefore, this should also be taken into consideration when comparing the points to these benchmarks. This could also partly explain the causes of the deviations in

between the NGI benchmarks and the point cloud heights in Table 3.3. For all practical purposes, these deviations can be ignored.

Similarly, there also exist Normaal Amsterdams Peil (NAP) benchmark points in the Netherlands that have been surveyed with respect to their X, Y, and Z RD-NAP coordinates. These can be found on the Rijkswaterstaat website (https://www.rijkswaterstaat.nl/zakelijk/open-data/normaal-amsterdams-peil). In total, there are 35,000 such benchmarks above-ground and 400 underground. An NAP-height of 0 is equal to the mean sea level of the North Sea (Rijkswaterstaat). General users have free access to *NAPinfo*, an online Rijkswaterstaat application with information about these NAP benchmarks. These benchmarks are usually 30-50 centimeters above the local ground level. Data about some of these benchmarks can be compared to the raster DEM derived from the AHN2 point cloud to ascertain the validity of the height values. A couple of these geodetic benchmark points were compared alongside the height values of the AHN2 point cloud dataset to test the validity of the AHN2 data. All of these points fall within the study area of this research that is a part of the Netherlands. The benchmarks were not compared to the point cloud dataset directly, but to a raster DEM representation of the point cloud that was obtained from PDOK (PDOK, 2017). This raster layer has a 0.5 meter by 0.5 meter spatial resolution, and represents a DTM of its area of coverage. It can be downloaded for specific tiles of the AHN2 directly in the form of a GeoTIFF file. For the purposes of this study, therefore, it was downloaded for the two tiles of the AHN2 that were used.

	Rijkswaterstaat NAP benchmarks			***Point Cloud Data***	
	Amersfoort - RD New projection		*NAP*	*NAP*	*Deviation*
Point ID	**X**	**Y**	**H**	**H**	
062D0068	198530	307790	234.122	233.667999	-0.454001
062D0028	198050	308920	180.858	180.781998	-0.076002
062D0030	197510	310260	169.748	168.940994	-0.807006
062D0029	199100	308820	220.359	220.154007	-0.204993
062D0027	197110	308750	240.921	239.151001	-1.769999
				Average	-0.6624002

Table 3.4: The geodetic benchmarks in the Netherlands used to check the validity of the acquired AHN2 LIDAR point cloud

Some of these benchmarks fall inside and some outside of the overlapping area with Belgium. As can be seen, the AHN2 dataset does not deviate significantly from the geodetic benchmarks set up by Rijkswaterstaat. Therefore, the AHN2 can be considered as sufficiently valid with respect to its height values.

It should be noted, however, that the height accuracy of the original points in the AHN2 is 0.1 meters. Therefore, this should also be taken into consideration when comparing the points to these benchmarks. This could also partly explain the causes of the deviations in between the NAP benchmarks and the AHN2 heights in Table 3.4.

Furthermore, according to technical specialists from AHN, there are no indications that the AHN2 in this region is incorrect. The height of the AHN is

based on the NAP and there are no indications that the soil level decreases or something similar occurs in this region. The height of the "*Drielandenpunt*" (TCP) itself is 322 meters above NAP.Therefore, it can be concluded that there are no problems with the height validity of the AHN2. So, the question still remains as to why the Belgian dataset does not align with the Dutch dataset in the overlapping area.

It can be concluded that the Belgian dataset simply is not accurate outside the territory of Belgium, where the overlapping area falls. A simple coordinate transformation can illustrate this reasoning. The Rijkswaterstaat NAP benchmark with ID *062D0117* falls within the overlapping area and has surveyed coordinates in RDNAP. The X-RD is 195550 m, Y-RD is 308640 m, and NAP is 275.203 m. These coordinates can be converted into the Belgian CRS (*EPSG 6190*) and the height-value of the same point in the Belgian point cloud data can be compared to the converted value. Proj.4 and the program PCTRANS (Defensie, 2017) were used to perform this conversion. PROJ.4 returned an Ostend height value of 276.62 meters, whereas PCTRANS returned 277.56 meters. Although these values are close to one another, they are not close at all to the value contained in the original GeoTIFF file for Belgium at the exact same location: 255 meters. This exercise indicates that the height value at that location should be a number such as 276-277 meters, but in the GeoTIFF file it has been found to be only 255 meters (see Figure **??**). Moreover, according to Pierre Voet, a geodetic specialist from NGI, the NAP and Ostend Height (SGL) should differ from one another by a maximum of only *2 meters and 30 centimeters*. This is simply not the case with the acquired data, so this allows for the viability of the stated conclusion.

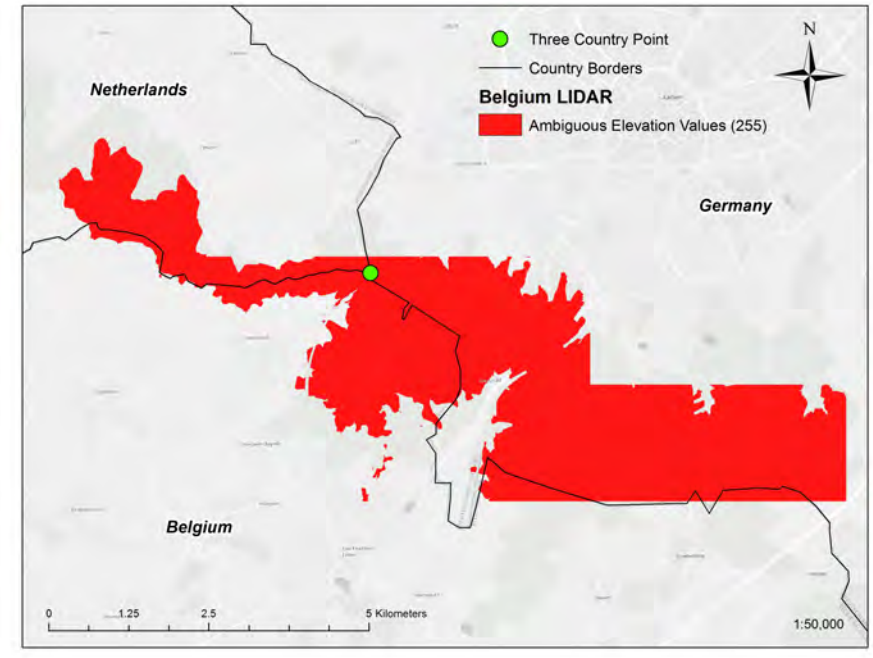


Figure 3.10: Extensive amount of ambiguous elevation values of 255 meters in the overlapping area between Belgium and the Netherlands. Units are meters.

The conclusion that can be drawn is that the Belgian point cloud dataset that was created from the GeoTIFF file contains valid height values only at

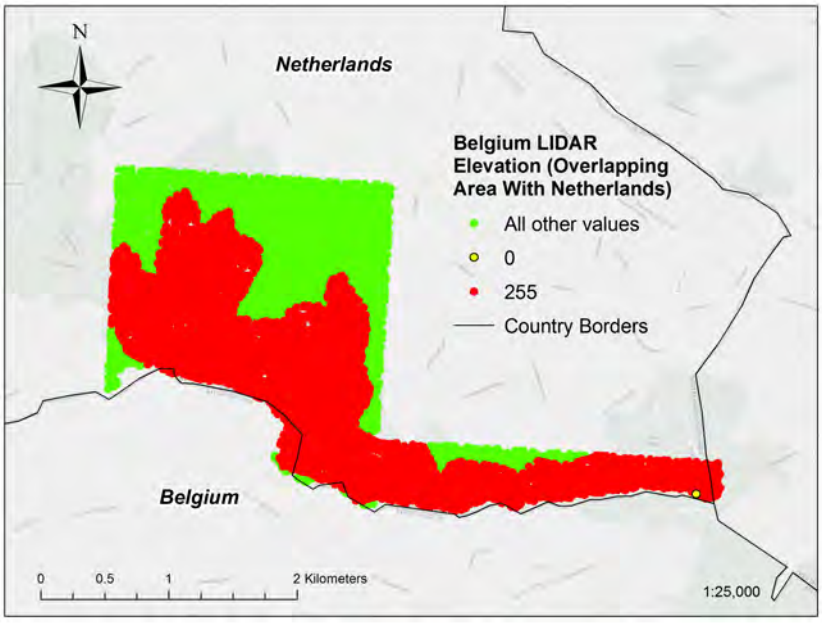


Figure 3.11: Extensive amount of ambiguous elevation values of 255 meters in the overlapping area between Belgium and the Netherlands. Units are meters.

certain places inside the extent of Belgium. Outside of Belgium and in the overlapping area within the Netherlands, the dataset significantly deteriorates in height validity. In contrast, the AHN2 with its extremely high point density contains exceptionally valid height values. The AHN2 does not extend much into the interiors of Belgium and Germany, unlike the other two datasets that extend considerably more into their neighboring countries.

3.1.3 Germany and Belgium

Z-Deviation

As can be seen in Figures 3.4 and 3.7, there are excessive differences in height values from (-50)-(-30) meters to (31-50) meters. More than half of the overlapping area contains deviations of (-50) to (-30) meters. The greatest positive height differences are mainly distributed along a small part of the German and the Belgian boundary. As can be seen from Figure 3.7 there are not as many positive deviations as negative deviations.

As explained in Section 3.1.1, the height values of Germany have not been correctly converted by taking into consideration the quasigeoidal height, i.e. the vertical distance between the ellipsoid and the quasigeoid. Thus, a value of approximately 46 meters is used as an approximation of the quasigeoidal height values (in the overlapping area) to be added to the source heights of the point cloud of Germany (in the *DHHN92* system). This is needed to recalculate the height differences between Germany and Belgium, since both of them will have by then heights in WGS84 (for Germany it is an approximation). As mentioned in Section 3.1.1, the value 46 is used firstly because the correction grid for the quasigeoid is not freely provided. The

e-tool used to calculate the quasigeoidal height in Germany with the online service (Anonymous, 2017) gives values of around 46 meters for this region. Lastly, because as it can be seen in Figure 3.8 the values of the correction grid for the quasigeoid in the three countries point area, range between 46m and 48m.

Apart from an investigation of quasigeoidal heights, an attempt was made to assess the quality of the source LIDAR point cloud of Germany to check if there is any relation with the observed height differences. The aim is to use height GCP's for the city of Aachen (Fachbereich, 2017). The drawback of this approach is that there is not any information about the distances of those points above the ground see (Atkis, 2017). Therefore, these GCP's are of no use since the examined point clouds represent information for the ground-surface (the terrain).

Thus, is proved that from the side of Germany, the initial founding regarding the differences between Germany and Belgium in the point clouds overlapping area are caused only by the ignorance of the quasigeoidal heights (separation of ellipsoid and quasigeoid). It is not stated though that this is the only reason (from the side of Germany), but this is was is proved.

With respect to the data of Belgium, some height reference stations (of the *Second General Levelling - Ostend datum, EPSG 5710*) were examined in Section 3.1.2 , for whom also their height above the ground surface is provided and their accuracy related to the point cloud dataset is considered sufficient. However, the fact that there are again an excessive amount of height values equal to 255 meters (in EPSG 5710) (see Figure 3.12) prompts for further investigation. By observing Figure 3.10, it can be seen that there is a big area within the boundary of Belgium and just on the south of the TCP, in which the height value according to the Belgian dataset is consistently 255 meters. Using the NGI online free service it can be seen that the examined area corresponds to a mountain 3.13. If that Figure is compared with Figure 3.10, it can be seen that the area within the territory of Belgium for which most of the 255 meter values are distributed, depicts the mountain 3.13. Nevertheless, there are no height benchmarks on the mountain to validate the ambiguous height values of 255 meters. However, it is possible to observe the contour lines in Figure 3.13, and see that there are many other height values on the mountain higher than 255 meters (e.g. 340, 300, and 338 meters).

Therefore, it can be concluded as to why there are such massive amounts of differences in height between Germany and Belgium, from the side of Belgium. The areas where the Belgian dataset has values of 255 meters are almost identical to the areas of maximum deviations between the two datasets (as well as with the Netherlands dataset).

Besides, according to (Wikipedia, 2017) the most amount of difference between the DHHN92 and Ostend Datum can be 2 meters and 30 centimeters at the most. Therefore, this makes it *impossible* to have an Ostend height of 255 meter because it results in much higher height differences (in WGS84) than discussed above.

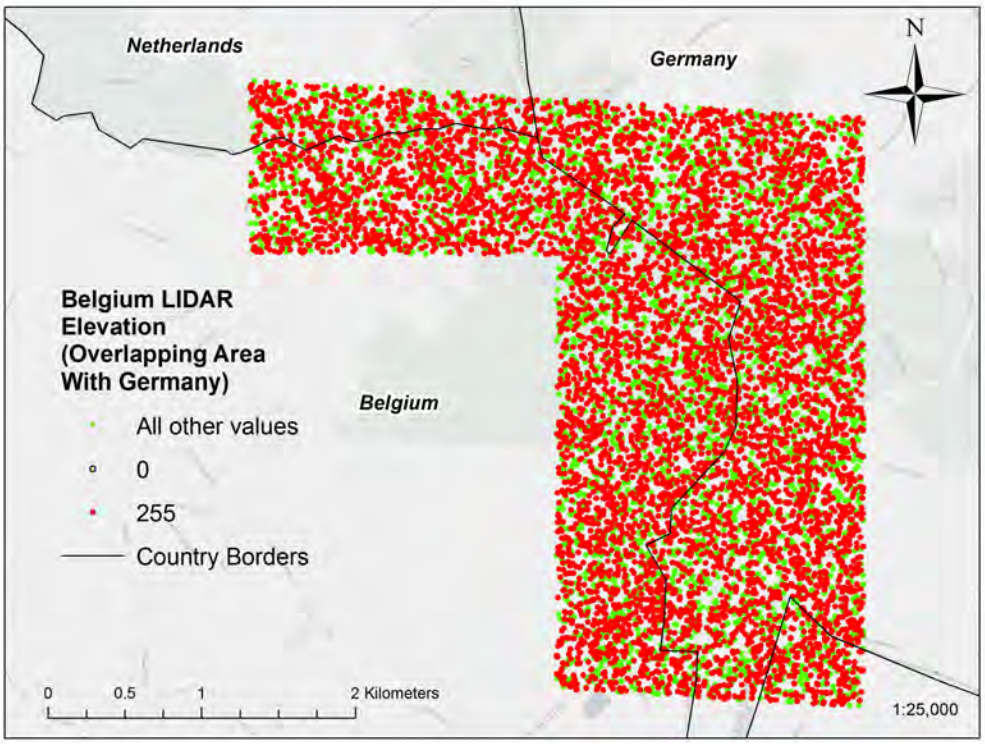


Figure 3.12: An extensive amount of ambiguous height values of 255 meter and one height value of zero meter in the overlapping area between Belgium and Germany.

Figure 3.13: *Preusswald*, also known by the name *Vaalserberg*, has an Ostend Height elevation value of more than 340 meters, whereas the Belgium LIDAR dataset at this location has a height of 255 meters. This goes to show that the dataset is not entirely accurate even inside the extent of Belgium. Figure obtained from the GDOC viewer from NGI Belgium(http://www.ngi.be/FR/FR0.shtm).

3.2 FINAL RESULTS

Maps and histograms for the final results of the point cloud harmonization are provided below. The Germany point cloud has been corrected, ambiguous LIDAR elevation values of 255 meter have been removed from the Belgium dataset, and the maps have been regenerated from the previous section. As can be seen, most of the deviations are very small; for instance, most of the height difference values in between the Netherlands and Germany fall within the -1 to 0 meter range; for Germany and Belgium, fall within the -0.5 to 0.5 meter range; and for the Netherlands and Belgium, fall within the -1.5 to 1.5 meter range. This can be deemed as an acceptable level of deviation.

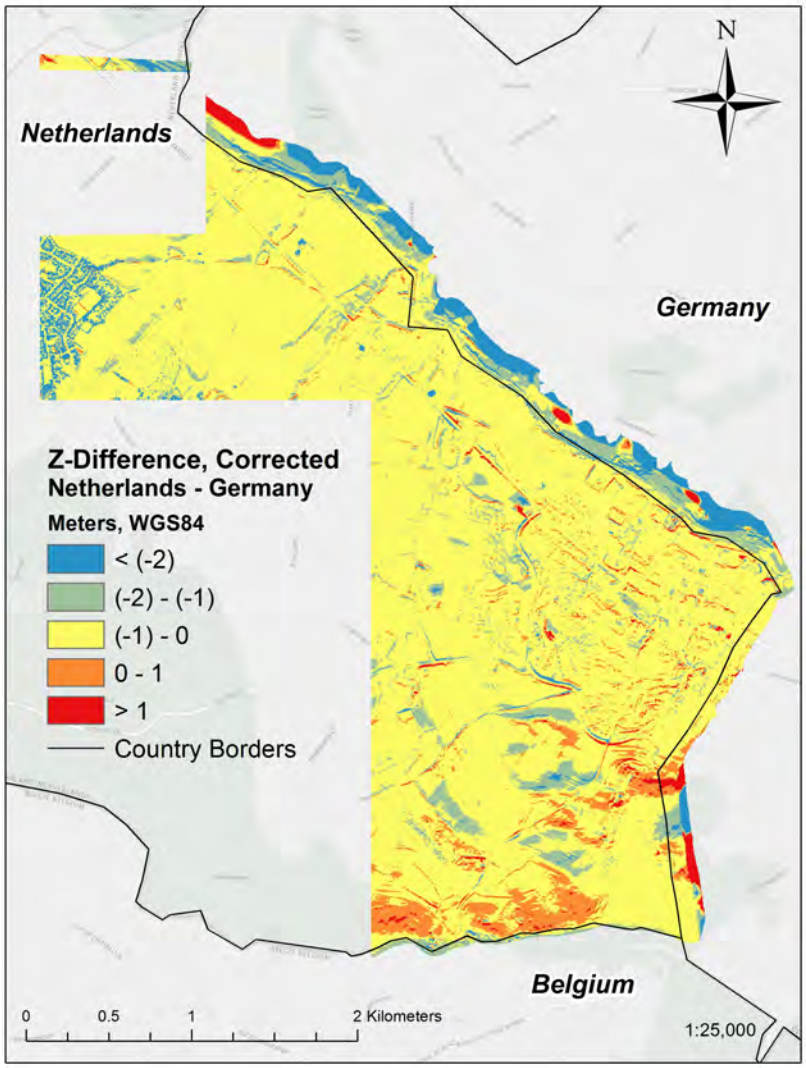


Figure 3.14: The Germany DEM subtracted from the Netherlands DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. This area falls almost entirely within the extent of the Netherlands.

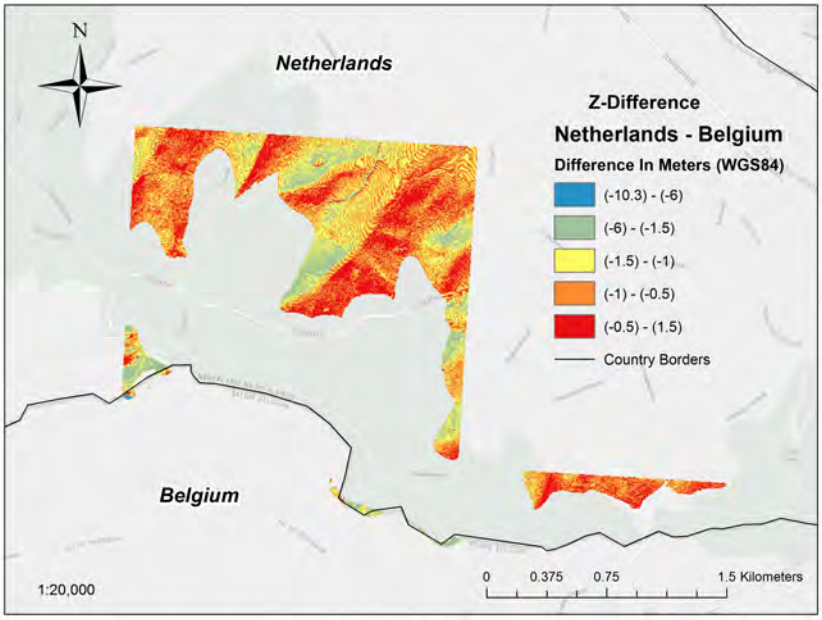


Figure 3.15: The Belgium DEM subtracted from the Netherlands DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. This area falls almost entirely within the extent of the Netherlands. The ambiguous Belgium elevation values of 255 meter were removed prior to the creation of this raster.

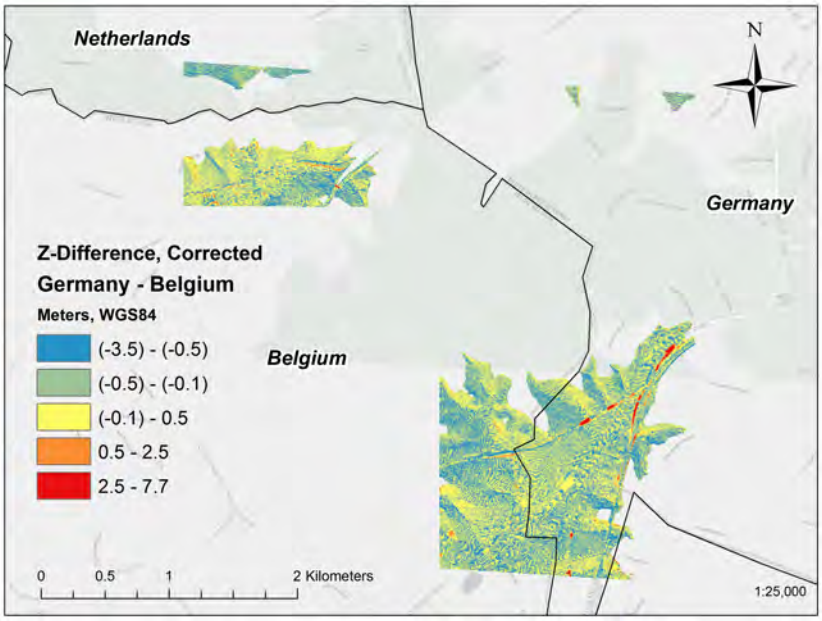


Figure 3.16: The Belgium DEM subtracted from the corrected Germany DEM. Both DEM's were generated using an interpolation of the Z-values in the point cloud datasets. The ambiguous Belgium elevation values of 255 meter were removed prior to the creation of this raster.

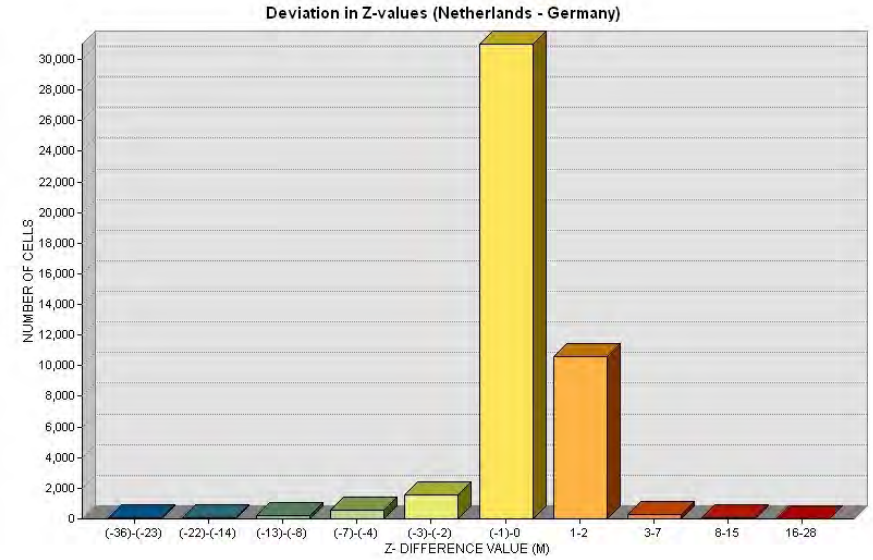


Figure 3.17

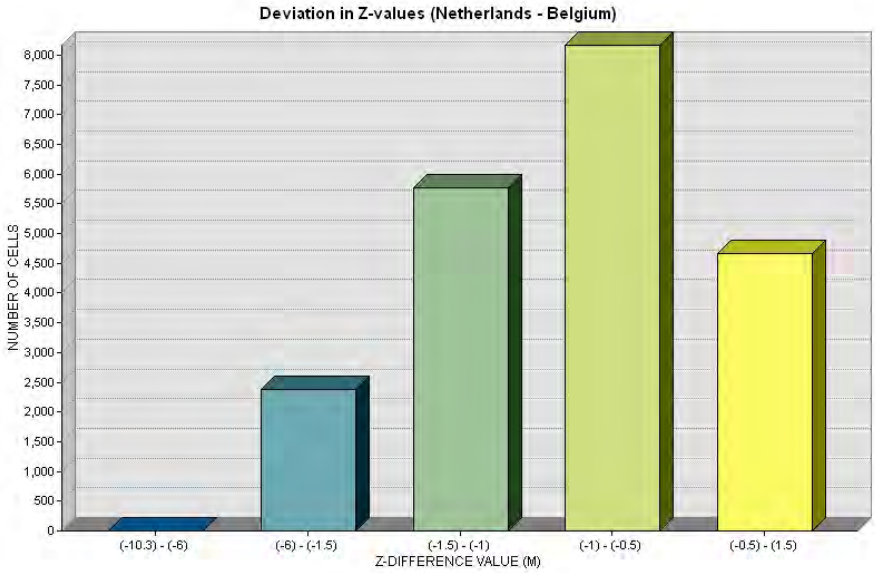


Figure 3.18

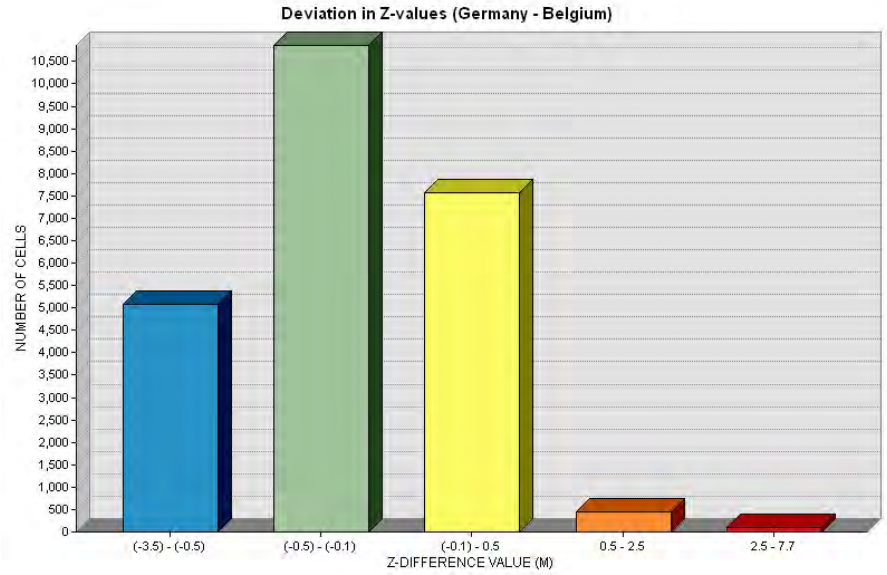

Deviation in Z-values (Germany - Belgium)
NUMBER OF CELLS
10,500
10,000
9,500
9,000
8,500
8,000
7,500
7,000
6,500
6,000
5,500
5,000
4,500
4,000
3,500
3,000
2,500
2,000
1,500
1,000
500
0
(-3.5) - (-0.5)
(-0.5) - (-0.1)
(-0.1) - 0.5
0.5 - 2.5
2.5 - 7.7
Z-DIFFERENCE VALUE (M)

Figure 3.19

4 DATA VISUALIZATION

The general objective of the Geomatics Synthesis Project 2017 is to research whether it possible to establish an open-source platform for the accessibility of point clouds i.e. something similar to the "Open Street Map of point clouds". One of the key deliverables of this research is an (open) point cloud web viewer containing harmonized datasets of the three countries bordering the TCP.

There are several options for visualizing our model in 3D, such as Potree, Cesium and Three.js. Potree is based on Three.js, and is used to visualize big point clouds, by clustering and indexing the data. Since our point cloud is relatively small (about 10.000/50.0000 points per hexagon), this is of no use. Cesium is a web viewer in which we cannot transform our data as easily as we can in Three.js. We conclude that the most appropriate tool for our case is Three.js, because it creates a simple 3D environment using a JavaScript library, which can be personalized. It is built around the Web Graphics Library (WebGL) APIs and can also handle many files at once, which means that the processing time is fast. WebGL is a web technology that brings hardware-accelerated 3D graphics to the browser without the need of installing plug-ins and downloading additional software (Agar, 2016). As a result it gives accessibility to many people as also it can be supported from different browsers such as Chrome, Firefox, IE, Opera, and Safari. It works on both mobile and desktop browsers. Three.js is an open source library that simplifies WebGL tools and environment. Using Three.js has several advantages as it includes a wide range of lower level programming in developing GPU-accelerated 3D animations (Agar, 2016).

To make a 3D visualization using Three.js, firstly, a scene has to be constructed. It contains all the objects, in our case the points of the point clouds, that are going to be rendered. The next step is to define the camera. In our case a perspective camera is being set. In order to be able to capture the mouse events on the canvas and to get our position back around the scene an external module is used named *Orbit controls*. Moreover, another external library is used named *dat.GUI* in order to create our interface where some basic information is given like the meaning of the colors that are assigned to the points and the coordinates in the WGS84 system of the position of the mouse after the user clicks on the screen.

After defining the above things we are ready to start our main script. As an input we have all the hexagons including the points covering the TCP region, in local coordinates and also the transformations matrices for each hexagon. So, first we have to define the geometry of our objects, in our case the 3D points inside the hexagons. Then we have to define the material of our objects. The parameters defining the material of the 3D points are the size and the color of the points. About the transformations, actually we are working with hexagons and each of them have different ID and transformation matrices that belong to one, two or to all of the three countries

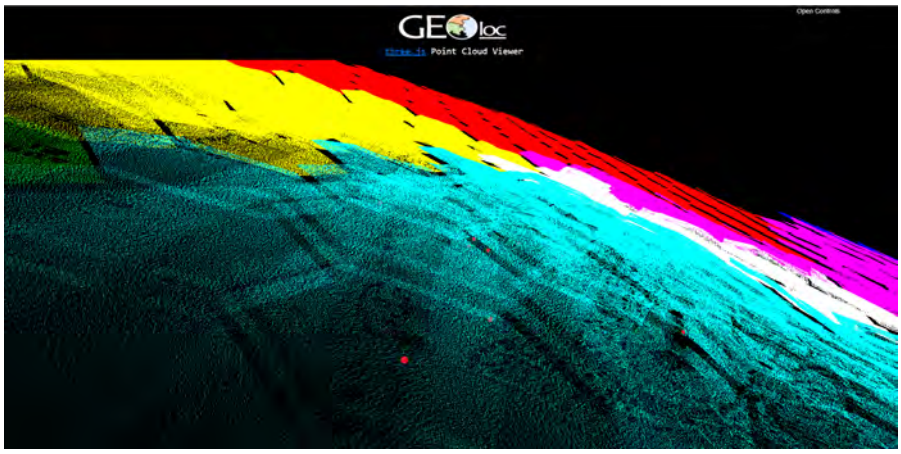


Figure 4.1: Transforming each hexagon from local coordinates to ECEF and visualizing the hexagons on the right place in the viewer

(Netherlands, Belgium, and Germany). Applying these matrices we are able to present the hexagons at their actual positions and display and visualize them in ECEF Coordinates, as these are Cartesian coordinates.

Figure 4.2: Colouring the overlapping areas

The next step is to present the countries we want to visualize with different colors so the user is able to understand where the boundaries for each country lie and the overlapping areas between the countries. For each country one of the primary colors is used (*Belgium: Red, Netherlands: Green, Germany: Blue*). For the overlapping areas between two countries the combination of their primary colors is used (*Belgium-Netherlands: Yellow, Belgium-Germany: Magenta, Germany-Netherlands: Cyan*). For the overlapping area between all three countries, the area is represented in white.

The local coordinates that are stored in the hexagon files have no meaning in a geographical sense. The meaning is added when the points are transformed back to ECEF and then to WGS84. To make the viewer user-friendly, an option to click on a point and display its on-the-fly converted coordinates in WGS84 has been added. The original idea was to use *proj4js*, a library for JavaScript. Unfortunately, conversion from ECEF coordinates to WGS84 is not implemented in this library, so the transformation was calcu-

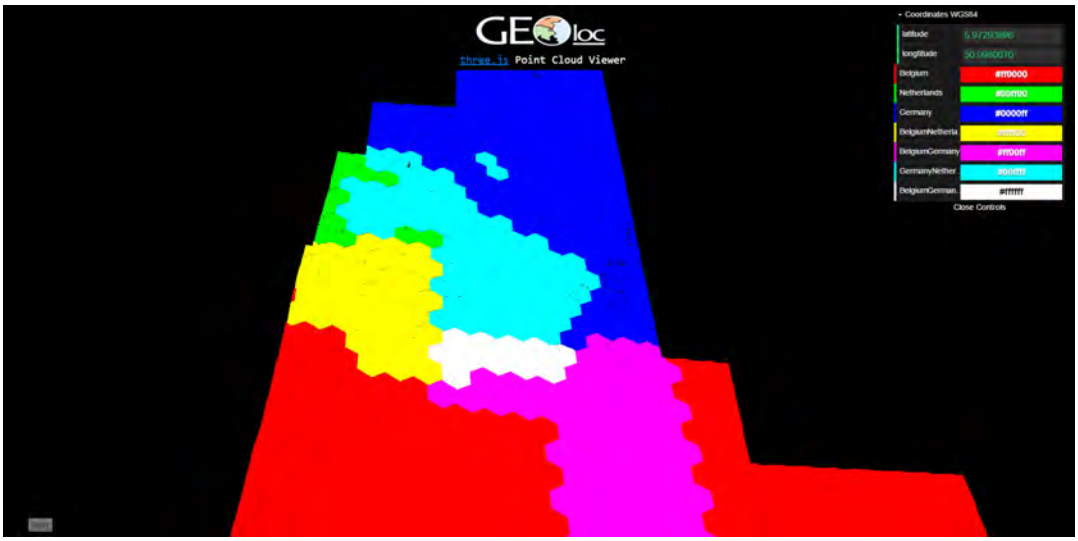


Figure 4.3: Final interface of the web viewer

lated manually. The pseudo code of the JavaScript code that has been used to implement the web viewer is provided in the Appendix.

As can be seen in Figure 4.2, there are some gaps between the hexagons. This suggests that the transformation to or from local coordinates was not completely accurate. It could not be figured out what the precise problem is, but our guess is that this has to do with the digital precision that is stored within the transformation parameters.

5 CONCLUSION

The general objective of this project is to research the harmonization of different point cloud datasets that connect and overlap at a common geographic area and to visualize the integrated data in a web viewer which allows for analysis. The TCP where Germany, Belgium, and the Netherlands share a border is chosen as a case study. Based on multiple GIS techniques and a literature review, the report seeks to answer the research question:

How can differences between point cloud datasets of various origins that connect and overlap at a common geographic area be harmonized to allow for data interoperability between these varying datasets?

As the scope of the research question is quite extensive, the paper focused on the challenges associated with the data collection, defining an ideal CRS for the datasets, analysis and harmonization of the differences in the data, and data storage.

Point cloud data of the three countries was provided as both a DSM and a DTM, out of which the DTM is more suitable for point cloud harmonization, as this model does not include above-ground objects like vegetation and buildings. During the data collection process, several differences in the acquired point cloud data had to be dealt with, such as the data accuracy, file formats, and different CRSs. Due to these differences, the datasets had to be processed to enable the CRS transformation in a later step. First, all the datasets were clipped to the study area of the regions close to the TCP. In case of the German and Belgian point cloud data, the format of the datasets was changed to (compressed) LAZ files; the Dutch point cloud data were already in LAZ format.

The main challenge of the scientific research in this project was the determination of an ideal common CRS for the point cloud data of different origins. Using a standard map projection for achieving the goal gives high distortions that differentiate at the equator and the poles. Furthermore, it displays the surface of the Earth as flat, while it actually is curved. There are mainly two kinds of CRS: the Cartesian Coordinate System and the Geodetic Coordinate System. The Cartesian Coordinate System measures positions in X,Y,Z ECEF coordinates resulting in large numbers that take up a lot of storage space. The Geodetic Coordinate System represents the Earth's surface using an ellipsoid, using geocentric latitude, longitude, and height above the ellipsoid. These are the simplest rotational shapes that match the true Earth. Although this is the most common representation for points on the surface, it is not very suitable for spatial analyses as well as for visualization of the latitude and longitude values in web viewers that are based on a Cartesian CRS.

In order to store the point cloud data efficiently and to be able to visualize the datasets in a web viewer, a DGGS was used as a spatial reference for dividing the Earth in a multitude of small regions, which makes spatial analyses easier to perform. The DGGS can be constructed by defining the following design parameters: polyhedron, cell type, polyhedron orientation, and inverse map projection. The icosahedron was chosen as the most suitable polyhedron due to the lower size of faces, and thus, less distortions. Hexagons are selected as cell refinement, as these quantize the plane with the smallest error and provide the greatest angular resolution. The best polyhedron orientation that serves the use case is the Sahr projection that has all but one vertex in sea. The Snyder Equal Area projection is used as an inverse map projection due to its property to preserve equal area cells. The aforementioned design parameters are specified in a metafile which is used in the DGGRID software to construct the ideal DGGS. This software was chosen, as it has the capabilities to create the icosahedral grid with the planar hexagonal topology and project the hexagons inversely on the WGS84 datum. While constructing the DGGS, no hierarchical indexing was implemented, which is why no multi resolution grids could be created.

After having defined an ideal CRS for the point cloud data, the CRS of the distinct datasets was converted first into the WGS84 system and then hexagons in the chosen DGGS were generated. The point clouds were clipped with the hexagons made using the DGGS and these were used to construct a local CRS for each hexagon in order to reduce data storage.

In order to analyze the height discrepancies of the point cloud data, the datasets were converted to raster DEMs using interpolation of the points in the datasets. The WGS84 heights were classified into classes using both an equal interval and natural breaks (Jenks) classification and the overlapping areas were analyzed for height discrepancies by subtracting one DEM from another DEM. The difference between the Dutch and German datasets is approximately 45-46 meters almost everywhere, since the software used for the CRS transformation (Proj.4) did not take the conversion of the German vertical CRS (quasigeoid) into account. An online service is provided for computing the quasigeoid height for one point at a time. However, the complete correction grid file is not provided free of charge, and therefore, a sample of 40 points was used for the harmonization. The difference between the Dutch and Belgian datasets ranges from -10 to +68 meters and the datasets deviate mainly in the southern parts of the overlapping area. To find the reasons for the visible difference, the datasets were compared to their benchmarks that were provided only within the countries' borders. It can be concluded that the Belgian dataset deteriorates significantly in accuracy in the overlapping area, as this area is almost entirely within the territorial extents of the Netherlands. As to the German and Belgian datasets, these do not align at all, i.e. about half of the German data are more than 42 meters below the Belgian. The height differences were corrected in the same way as described in the part related to the differences between the Dutch and German datasets.

As a result of the point cloud harmonization, a web viewer was created according to the requirements of the client to allow for visualization and/or analysis of the integrated data. For that purpose, Three.js was used due to the variety of options it offers for visualizing in 3D. In order to be able to show the point cloud data, the latitude and longitude and the centroid of

the hexagons along with the X,Y,Z ECEF coordinates of the same point were used to construct transformation and rotation matrices to get to the local CRS. This allowed the storage of the points in Cartesian coordinates with the hexagon centroid as origin and the hexagon plane as ground plane. In that case, the translation and rotation matrix was applied to each hexagon so that it can be shown at the correct place. In addition, the WGS84 coordinates were also displayed after a user click event and the points were colored according to country.

Despite some limitations of the research, the solutions found proved to be a good approach to integrate point cloud data of different origins, due to the meaningful results obtained. Therefore, the method has great potential to work as a common model for harmonizing point cloud data.

6 FURTHER RECOMMENDATIONS

As expected, several essential issues should be considered for advancement in terms of the DGGS and web viewer. The ideal DGGS for the purpose of this project was created in the DGGRID by using several tools that the software offers. In order to make the research applicable in a global scale, the following recommendations for the DGGS generation are provided:

- **Hexagonal grids**. Further research is necessary to find out how the icosahedron grids are actually created in terms of the equations used, computing limitations, accuracy, etc. According to Sahr (2015), "while we believe it performs as described in this documentation, we do not guarantee or warranty DGGRID for any specific use or purpose." The recommendation for the extra research is based on this statement and the inconsistent results generated while testing the method *Superfund*.

- **Inverse map projection**. ISEA was used, but it has the disadvantage that it produces changes in the directions (Furuti, 2015), which are visible as breaks in the lines connecting vertex to face centers. Therefore, further research should be done to find other possible equal area projections that do not give these errors.

- **Indexing**. As the study area was very small, single resolution grids were created, which is not sufficient to constitute a DGGS (Strobl et al., 2016). Thus, a hierarchical indexing method for the hexagonal grids should be developed, otherwise a hierarchy of grids with successively finer resolution cannot be efficiently used.

The web viewer was developed with Three.js that uses a JavaScript library and offers a wide range of functionalities to visualize 3D objects and process point cloud data. However, there are several recommendations to improve the performance of the web viewer in the future:

- **Interactive**. The web viewer could be made more user-friendly by adding various options like turning on/off layers according to the area or country that the user wants to examine.

- **Analysis tools**. It would also be useful to add some tools on the interface that allow the user to calculate the distance and give some statistics about the input data, such as the quality and the density of the point cloud.

- **Open data**. In order to make the web viewer truly open, download and upload functions could be added so that the users are able to work with the data.

BIBLIOGRAPHY

AdV (2017). Hoetra2016.

Agar, M. (2016). Animating scenes with webgl and three.js.

Amiri, A. M., Samavati, F., and Peterson, P. (2015). Categorization and conversions for indexing methods of discrete global grid systems. *ISPRS International Journal of Geo-Information*, 4(1):320–336.

Anonymous (2017). Onlineberechnung von Quasigeoidhöhen mit dem GCG2016.

Atkis, A. A. (2017). Working Committee of the Surveying Authorities of the States of the Federal Republic of Germany - AdV-Online.

Budge, S. and Niederhausern, K. (2011). Automatic merging of lidar pointclouds using data from low-cost GPS/IMU systems. *Laser Radar Technology and Applications XVI*.

Bundesamt für Kartographie und Geodäsie (2016). Quasigeoid der Bundesrepublik Deutschland. Die Höhenreferenzfläche der Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder.

Chapman, C. and Ward, S. (2003). *Project Risk Management: Processes, Techniques and Insights.* John Wiley & Sons, Ltd, Chichester, 2nd edition.

Defensie, M. v. (2017). PCTrans - Hydrography - Defensie.nl.

Dutton, G. (1996). Encoding and handling geospatial data with hierarchical triangular meshes. In *Proceeding of 7th International symposium on spatial data handling*, volume 43. Netherlands: Talor & Francis.

ESRI (2017). Lidar point classification.

Fachbereich, B. (2017). Höhenfestpunkte (HFP) - Open Data Portal.

Featherstone, W. and Vanicek, P. (1998). The Role of Coordinate Systems, Coordinates and Heights in Horizontal Datum Transformations.

Frisch, U., Hasslacher, B., and Pomeau, Y. (1986). Lattice-gas automata for the navier-stokes equation. *Physical review letters*, 56(14):1505.

Fuller, R. B. (1975). *Synergetics: explorations in the geometry of thinking*. Estate of R. Buckminster Fuller.

Furuti, C. A. (2015). Map Projections: Polyhedral Maps - part 1.

Geonovum (2015). Handreiking ruimtelijke referentie systemen.

Geonovum (n.d.). Advies overstap RD naar ETRS89.

Gibb, R. G. (2016). The rHEALPix Discrete Global Grid System. 34, page 9, New Zealand. Earth and Environmental Science.

Gutierrez, A. (2007). Platonic Solids, Theaetetus's Theorem, Euler's Polyhedron Theorem. HTML5 Animation for iPad.

Heiskanen, W. and Moritz, H. (1967). Physical Geodesy.

INSPIRE (2008). Specifications On Coordinate Reference Systems – Draft Guidelines. Technical Report D2.8.I.1, INSPIRE Reference Systems Thematic Working Group.

INSPIRE (n.d.). INSPIRE Principles.

Isenburg, M. (2017a). Nrw open lidar: Download, compression, viewing.

Isenburg, M. (2017b). Nrw open lidar: Merging points into proper las files.

Killet, F. (2012). Are Reference Systems WGS84 and ETRS89 despite continental drift really same?

Lounge, G. (2011). Gis data: A look at accuracy, precision, and types of errors.

Marel, v. d. (2016). *Reference Systems for Surveying and Mapping*. Delft University of Technology.

Miranda, E. (2011). Time Boxing Planning: Buffered Moscow Rules | StickyMinds.

Mochal, T. (2007). Define project scope to include deliverables, boundaries, and requirements - TechRepublic.

NOAA (2017). What is a datum?

NRW, G. (2017). Digitales Oberflächenmodell mittlerer Punktabstand 1m.

Ottoson, P. (2001). Retrieval of geographic data using ellipsoidal quadtrees. In *ScanGIS*, pages 89–98. Citeseer.

PDOK (2012). Actueel hoogtebestand Nederland (gefilterd) (AHN2).

PDOK (2017). PDOK - publieke dienstverlening op de kaart.

PMI (2000). *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*. Project Management Institute, Inc.

Product & Process Innovation, I. (2017). Project Management Guru Initiation.

Purrs, M. (2015). OGC Discrete Global Grid System (DGGS) Core Standard.

PYXIS (2006). Discrete Global Grid System.

Sadiq, M. and Ahmad, Z. (n.d.). Comparative study of geoid-quasigeoid separation at two different locations with different topographic distributions.

Sahr, K. (2015). dggridManualV62.pdf.

Sahr, K., Carr, D., Cressie, N., Gregory, M., Huntley, F., Kahn, R., Kiester, A. R., Kimerling, A. J., Knighton, J., Olsen, A., Rosenbaum, B., Song, L., and White, D. (2015). DGGRID Software.

Sahr, K., White, D., and Kimerling, A. J. (2003). Geodesic discrete global grid systems. *Cartography and Geographic Information Science*, 30(2):121–134.

Snyder, J. P. (1992). An equal-area map projection for polyhedral globes. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 29(1):10–21.

Song, L., Kimerling, A. J., Sahr, K., and others (2002). Developing an equal area global grid by small circle subdivision. *National Center for Geographic Information & Analysis, Santa Barbara, CA, USA*.

Strobl, P., Purrs, M., Peterson, P., and Samavati, F. (2016). Discrete Global Grid Systems for handling big data from space.

Vanicek, P., Kingdon, R., and Santos, M. (2012). Geoid versus quasigeoid: a case of physics versus geometry. *Contributions to Geophysics and Geodesy*, 42(1):101–118.

Wikipedia (2017). Digital elevation model.

Wikipedia (2017). Höhe über dem meeresspiegel. Page Version ID: 166215902.

Wikipedia (2017). Höhe über dem Meeresspiegel. Page Version ID: 166215902.

Zon, N. v. d. (2013). Kwaliteitsdocument AHN2. Technical Report 1.3, AHN, Amersfoort.

A SOFTWARE COMPARISON: CHARACTERISTICS

	Characteristics	Advantages	Disadvantages
Proj4			
	A standard UNIX filter function Data storage in a database Converts geographic (long,lat) coordinates (and vice versa) C API for software developers to include coordinate transformation in their own software Is maintaned on Github	Support las/laz files Supports many transformations with the correction grids Global use Used by most of the existing QGIS package Open source	Complicated installation procedure Grid shift (datum transformation) files not available for all countries
PC Trans			
	PCTrans network version. This version is intended for network computers at companies and government institutions. (Defensie.nl, Hydrography, 2017)	Accurate transformations for NL data (correction grid) Open source	Does not support laz files Time for preprocessing data No global use
FME			
	Interoperability tool that can harmonize data from hundreds of data formats Can handle data from different coordinate systems or combine datasets with those in other coordinate systems	This specification can handle 2D and 3D coordinates, as well as 4D,5D etc. Works with las/laz files directrly.	For computing height, correction grids are needed for the applicable countries
C-Convert			
	Transforms coordinates between the following systems used in Belgium and the associated projections	Contains the correction grid for Belgium Open source	It is only for Belgium Does not support las/laz files directly.

Table A.1: Comparing software for coordinate conversion

B SOFTWARE COMPARISON: RESULTS

				ETRS89/UTMZone32N		ETRS89/Latitude Longitude		
NL	Original X Value	Original Y value	Original Z Value	New X Value	New Y Value	New Longitude	New Latitude	New Height
Proj.4	200078.68	309148.47	200.83	290319.8741	5628521.0329	6.02618832	50.7705010	246.289
PC Trans	200078.68	309148.47	200.83	290319.8739	5628521.0304	6.02618709	50.7704999	247.112
FME	200078.68	309148.47	200.83	290319.8737	5628521.0305	6.02618832	50.7705009	Need geoid height grid
Lidar Convert	200078.68	309148.47	200.83	*REMAINS TO BE TESTED*				

Table B.1: Comparison of software test for the Netherlands

				ETRS89/UTMZone32N		ETRS89/Latitude Longitude		
Germany	**Original X Value**	**Original Y value**	**Original Z Value**	**New X Value**	**New Y Value**	**New Longitude**	**New Latitude**	**New Height**
Proj.4	296528	5618076	273.07	296528	5618076	6.119876637	50.67889911	273.07
PC Trans	296528	5618076	273.07	296528	5618076	6.11987764	50.67889911	----
FME	296528	5618076	273.07	296528	5618076	6.119876637	50.67889911	Need geoid Height grid
Lidar Convert	296528	5618076	273.07	*REMAINS TO BE TESTED*				

Table B.2: Comparison of software test for Germany

				ETRS89/UTMZone32N		ETRS89/Latitude Longitude		
Belgium	**Original X Value**	**Original Y value**	**Or.V. Z**	**New X Value**	**New Y Value**	**New Long.**	**New Latitude**	**New Height**
Proj.4	266000.5	161999.5	255	289320.4444	5627050.706	6.012875928	50.7569344	299.105
PC Trans	266000.5	161999.5	255	289320.4446	5627050.705	6.01287762	50.75693425	298.555
FME	266000.5	161999.5	255	289320.4669	5627050.77	6.012876209	50.7569350	geoid height grid
Lidar Convert	266000.5	161999.5	255	*REMAINS TO BE TESTED*				
C-Convert	266000.5	161999.5	255	289320.562	5627050.673	6.012877623	50.75693425	298.955

Table B.3: Comparison of software test for Belgium

C DGGS COMPARISON

Table C.1: Possible parameters and their advantages

Design parameters comparison to define the optimal DGGS for point cloud datasets					
POLYHEDRON	**Icosahedron**	**Cube**	**Tetrahedron**	**Octahedrons**	**Dodecahedrons**
Advantages	Smallest face size - less areal and angular distortions	Square quadtree subdivision-Efficiently handled	Simplicity	Better approximation of the earth	Adjusted: Smallest face - Even less distortions
Disadvantages		Higher distortions than others	Higher distortions than others	Higher distortions than Icosahedron and Dodecahedron	Multiresolution not possible Adjusted: Cells not congruent-edges inconsistencies & tree-based algorithms cannot be directly used
CELL TYPE	**Square**	**Triangule**	**Hexagon**	**Diamond**	
Advantages	Unsuitable for triagulated polyhedrons		Smallest error-better angular resolution Uniform adjacency-Neighboors Share an Edge-Discrete events' simulations	Can be used in triangulated poluhedrons - Use quad-tree based algorithms	
Disadvantages				Centroids not in the same distance-Not uniform adjacency-Complicate data analysis	
ORIENTATION (Icosahedron)	**Orientation 1**	**Orientation 2 (Dymaxion)**	**Orientation 3 (Sahr)**		
Description	Vertices on poles and one edge aligned with the prime meridian	Vertices in oceans	Triangles oriented in a way that there is: Symmetry on the equator Minimum vertices in land		
Advantages		No ruptures at the land			
Disadvantages	No symmetry on the equator				
TRANSFORMATION	**Direct Spherical Subdivision (DSS)**	**Inverse Map Projection (IMP)**			
Advantages	Grid Cell of equal area	Higher efficiency than DSS	**1. Gnomonic**	High distirtions areas, shapes	
			2. Snyder projections	Equal area Irregular spherican cell ages Changes in directions Efficient statistical analysis Uniform coverage of data	
			3. Fuller's Dymaxion	Only for icosahedrons Less area, shape distrotions than gnomonic Less shape sistortions than Snyder Not equal area spherical cells	

D PSEUDOCODES

Algorithm 1 Construct local coordinates

Require:

Centroid (lat/lon) hexagons = C_{hex}^{latlon}

Array of points per hexagon = P

rotationMatrix = $\begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix}$

Ensure:

function PROJ(pt)

EPSG:4326 $\leftarrow Pr_{in}$

EPSG:4978 $\leftarrow Pr_{out}$

pt2 = Project(pt,Pr_{in} to Pr_{out})

return pt2

for i in C_{hex}^{latlon}) **do**

Hexagon $\leftarrow$ i.id

Hexagon.rMatrix $\leftarrow$ rotationMatrix(i)

$i_{ECEF} \leftarrow$ PROJ(i)

Hexagon.tMatrix $\leftarrow i_{ECEF}$

for hexArray in P **do**

Hexagon $\leftarrow$ hexArray.id

rotation $\leftarrow$ Hexagon.rMatrix

translation $\leftarrow$ Hexagon.tMatrix

localHexArray $\leftarrow \emptyset$

for Pt in hexArray **do**

ECEFPt $\leftarrow$ PROJ(pt)

LocalPt $\leftarrow$ (ECEFPt-translation)*rotation^{-1}

localHexArray $\cup$ LocalPt

END

Algorithm 2 Projection from Belgium Reference system to WGS'84

Require:
List of Belgium points (x,y,z) ← points
C ← [ ] ▷ empty list to get the new transformation points from Belgium

Ensure:
function PROJ(pt)
EPSG:6190 ← Pr_{in}
EPSG:4326 ← Pr_{out}
pt2 = Project(pt,Pr_{in} to Pr_{out})
return pt2

for pt in points **do**
C ← $PROJ(pt)$

END

Algorithm 3 Create a point cloud web viewer

Require:
List of hexagons with points inside = H
Dictionary with the translation matrix of each hexagon = Tmatrix
Dictionary with the rotation matrix of each hexagon = Rmatrix

Ensure:
function INIT
origin ← 3D central point
camera ← perspective camera with x, y, z position
gui ← displaying the coordinates in the interface

for hexagon in H **do**
for point in hexagon **do**
geometry ← defining the geometry
material ← defining size and color
particles ← adding geometry and material to the points

if hexagon.country = Germany **then**
hexagon.id ← specifying the id of the hexagons
matrix ← applying Tmatrix and Rmatrix for the given hexagons
color ← adding purple color to the given hexagons
if hexagon.country = Belgium **then**
hexagon.id ← specifying the id of the hexagons
matrix ← applying Tmatrix and Rmatrix for the given hexagons
color ← adding blue color to the given hexagons
if hexagon.country = Netherlands **then**
hexagon.id ← specifying the id of the hexagons
matrix ← applying Tmatrix and Rmatrix for the given hexagons
color ← adding green color to the given hexagons

function ONDOCUMENTMOUSEDOWN(event)
mouse ← 2D vector ▷ specifying the position on the screen
raycaster ← mouse picking
intersect ← getting the position of the closest point
ptlanlon ← TRANSFORMXYZ(pt)
sphere ← creating a new object around the selected point
gui.latitude ← displaying ϕ from the coordinates transformation
gui.longitude ← displaying λ from the coordinates transformation

END

E MEDIA OUTREACH STRATEGY

The purpose of this appendix is to inform society on the outcome of the Geomatics Synthesis Project 'Towards Point Cloud Harmonization'.

• Audiences identification

The project refers to people that are interested in point cloud datasets, big data handling, storing, integration, access and visualization in 3d. But not only! To achieve the goals, many data transformations are applied, including common ones, such as converting from local CRS to global CRS (WGS84), but also custom ones. The transformations are strongly related to the method used for accessing and visualizing the point clouds. That specific method, the Discrete Global Grid System, aims for a virtual representation of the planet, with emphasis on point cloud data management. These could easily attract earth scientists, cartographers and people who cope with spatial data visualization. Besides this, since the project deals with LIDAR datasets from and for neighbouring countries that form an overlapping area, it comprises critical steps regarding the data integration. This is an issue that every person who works with geospatial data has to face almost in a daily basis. What is interesting about that is that the data interoperability involves subjects such as global and local coordinate reference systems, geoids, quasigeoids, height benchmarks and special representations to indicate 'no data'. Furthermore, prior to the data comparison in the overlapping areas, the LAS files are processed and analysed with tools such as FME, ArcGIS and LAStools.
Overall, the project approaches a variety of spatial procedures and knowledge. Thus, the project could catch the attention of conferences such us 'The International Conference on Advanced Geographic Information Systems, Applications, and Services', the 'Big Data Week Conference', the 'ISPRS International Joint Conference which for 2017 has a subject: Acquisition, storage, modelling and analysis of spatial data and information' and the '3D Geoinfo Conference'. In terms of societies, the project could attract the 'International Society for Digital Earth' and the 'Point Cloud Library'. Regarding companies, at least in the Netherlands, the project interests Fugro (partner company) and could be interesting for Kadaster, Rijkswaterstaat, Cyclomedia, ESRI etc. Ultimately, the project also refers to people that are working specifically with spatial data from Belgium, Germany and Netherlands.

• What society should know

There is now a method developed that allows for harmonization and visualization of point cloud datasets. This method is not only suitable for small scale applications, but also for datasets that cover extents of countries. By saying harmonization of point cloud datasets from different countries, 2 things are implied:

a) A method is proposed that allows for integration, easiest and quickest (than the existing solutions, see section 1) access, processing and visualization of massive point clouds that have different sources.
b) Analysis of the point clouds is applied on the overlapping areas of three countries with the aim to explain the possible differences and if feasible, to harmonize (minimize) them.

• Relationships with Community Partners

The main partner of the project was the company Fugro. To establish relationships, weekly meetings were settled with the mentors from the company. During those, the team's purpose was to inform the partners about the progress made, but also to acquire feedback. Apart from that, two presentations were scheduled, one for the mid-term achievements and one for the final research and outcomes. In addition, contacts and meetings were made with some other employees of the company for more technical support, i.e. regarding the visualization using the Javascript 3D library (Three.js).

• Use of existing resources to inform society

To create news material for this project existing geospatial magazines could be approached, but also attempts can be made to contact organizers of conferences with related focus. The executive summary (i.e. a scientific article) could be used to explain what were the objectives and the achievements. This could be accompanied with the poster made, and if someone is eager to know more, then the report which clarifies everything, could be provided.

• Propose a scenario for outreach

To actually bring this project to a next level, the outputs of the project can be sent to people that were contacted during the project or people that could be interested. Precisely, these could be the contributors to the DGGrid Software such as Kevin Sahr, the geodetic specialist from the National Geographic Institute of Belgium Pierre Voet, employees at the Dutch Cadastre e.g. Jochem Lesparre and Lennard Huisman who gave lectures about geoids for the Geomatics MSc at TU Delft the last years, the Professor Thomas H. Kolbe from TU München (guest at TU Delft the academic year 2016 – 2017) and employees at the Cadastre of Germany (Bundesamt für Kartographie und Geodäsie). In addition, the conferences mentioned above, the magazines GIM International and Geo-Informatie Nederland (GIN), and the organization FIG Young Surveyors Network through GIN (which is an associate member) could be approached for a possible publication.

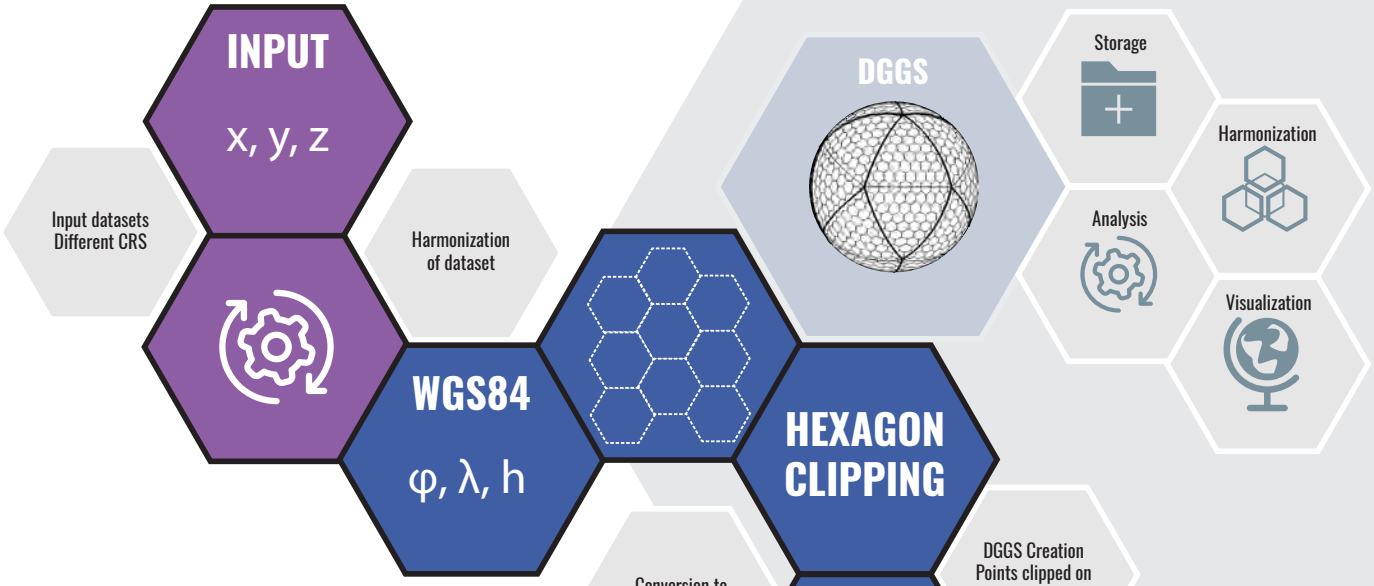
F POSTER

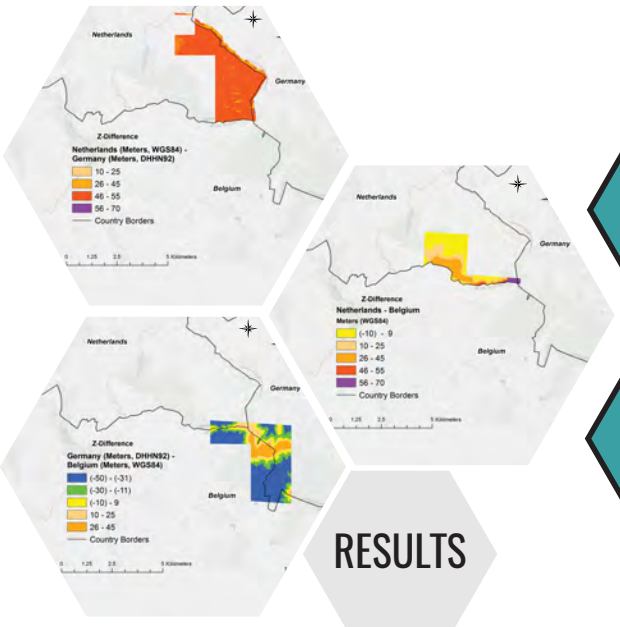
POINT CLOUD HARMONIZATION

Geomatics Synthesis Project

GOAL

Harmonizing and **visualizing** point cloud datasets from different countries using a common Coordinate Reference System.





RESULTS

GEOloc

FUGRO

TU Delft

Neeraj Sirdeshmukh
Andria Christodoulou
Lydia Kotoula
Manuela Manolova
Laurens Oostwegel