




UPDATING THE DANISH ELEVATION MODEL WITH UAV DATA

**HENRIK BRÆNDSKOV LARSEN
& NINA STAHL MADSEN**

**SPECIALISATION IN SURVEYING AND MAPPING, 4TH SEMESTER
MSC IN SURVEYING, PLANNING AND LAND MANAGEMENT
DEPARTMENT OF PLANNING
AALBORG UNIVERSITY
D. 02-06-2017**





AALBORG UNIVERSITY
STUDENT REPORT

The Technical Faculty of IT and Design

Department of Planning

Vestre Havnepromenade 5

9000 Aalborg

<http://www.plan.aau.dk/>

Title:

Updating the Danish Elevation
Model with UAV data

Theme:

Master's Thesis

Project period:

4th semester, spring 2017

Group members:

Henrik Brændskov Larsen

Nina Stahl Madsen

Supervisors:

Jens Peter Cederholm

Jens Juhl

Number of pages: 114 + appendix

Date of hand in: 02-06-2017

Synopsis:

In the present project it is investigated how UAV data can be used for making local updates to the Danish Elevation Model (DK-DEM). It was found that a change detection and categorisation of the UAV pointcloud is essential parts of an algorithm for updating the DK-DEM. It was experienced that the UAV data have several quality issues and for this reason a quality assessment of the data was required. It was chosen to prioritise the development of an algorithm for quality assessment and change detection. The quality assessment removes outliers in the UAV data by performing a robust adjustment of a plane on the UAV data and the data quality is evaluated by statistical measures from the adjustment. The change detection evaluates elevation differences by a standard deviation of the difference. The standard deviation of the difference is based on the quality of both the DK-DEM and the UAV data.

The quality assessment successfully removed outliers in the UAV data and it was seen that the dataset in general had a higher precision after it was performed. The change detection was able to detect changes of terrain and from inspection of the change detection it was seen that the algorithm achieve a satisfying result. As a categorisation have not been performed several issues regarding non-terrain objects is present. For this reason the developed algorithms is only seen as part of a final algorithm.

Preface

The present Master's Thesis was made by Henrik Brændskov Larsen and Nina Stahl Madsen on the 4th semester of the Master's Program in Surveying, Planning and Land Management with specialisation in Surveying and Mapping.

The time frame of the Master's Thesis runs from the 1st of February 2017 to the day of the examination, which is the 15th of June 2017. The project report was digitally handed in on the 2th of June 2017.

The project group would like to thank Andrew Flatman and SDFE for their support and help during the initial processes and data supply.

The image on the front-page is from [The Danish Road Directorate, 2016].

Reading guide

Sources of literature used through the project are referred to through the report and presented in the bibliography at the end of the report. Source references are listed according to the Harvard citation style.

A number of appendices are to be found at the end of the report, where Appendix B, *Digital appendix* lists the contents of the digital appendix.

Contents

1	Introduction	1
1.1	Basic data in Denmark	1
1.2	Sources of elevation data	3
1.3	UAVs in Denmark	6
1.4	Case area - Odense south-east	7
1.5	Issues regarding photogrammetric based DTMs	8
1.6	Project focus and initial problem statement	10
2	Problem analysis method	15
2.1	Elevation models	16
2.2	Comparison of elevation models	16
2.3	Methods and supporting datasets	17
3	The Danish Elevation Model	19
3.1	Quality control of the DK-DEM/PointCloud	21
4	GeoFyn data	23
4.1	Data collection	23
4.2	Data presentation	25
5	DEM comparisons	29
5.1	DK-DEM/Surface – DK-DEM/Terrain	29
5.2	GeoFyn DSM – GeoFyn DTM	30
5.3	GeoFyn DSM – DK-DEM/Surface	33
5.4	Summary	34
6	Methods for the updating process	35
6.1	Quality assessment	36
6.2	Categorisation	39
6.3	Change detection	48
7	Problem statement	53
7.1	Summary	53
7.2	Choice of data and methods	55

8	Method	57
8.1	Quality assessment	58
8.2	Change detection	59
8.3	Changes around Odense SE	60
8.4	Conclusion and discussion	60
9	Quality assessment	61
9.1	Dividing point cloud into cells	62
9.2	Robust adjustment of plane	63
9.3	Remove outliers	71
9.4	Summary	73
10	Change detection	75
10.1	Adjustment of planes with remaining data	77
10.2	Subtraction of DEMs	83
10.3	Standard deviation of the difference	84
10.4	Detect changes	87
10.5	Filter change detected cells	96
10.6	Summary	98
11	Changes in Odense SE	101
12	Conclusion	105
13	Discussion	107
	Bibliography	109
A	GeoFyn specification	
B	Digital appendix	
C	select_data.m	
D	data_to_cells.m	
E	planfit.m	
F	remove_outliers.m	
G	plan_fit_data.m	
H	change_detection.m	
I	median_filter.m	
J	test_sigma_0_vs_s.m	

Introduction 1

This project rapport concerns the problem of making local updates to the most recent version (2015) of the Danish Elevation Model (DK-DEM). This problem was presented by the Danish Agency for Data Supply and Efficiency (SDFE). More specifically they are interested in locally updating the DK-DEM by alternative means than airborne Light Detection And Ranging (LiDAR) scanning which was used to create the DK-DEM. Based on a discussion with SDFE it is understood that only preliminary considerations regarding other means of making local updates has been made, but ideas concerning the use of unmanned aerial vehicle (UAV) data has been suggested. UAV data can be several types of data, in this case it is defined as being a raw point cloud derived from true color (RGB) images captured from an UAV weighing less than 25 kg.

Several other datasets in the Danish *Basic Data* program may play a role in the updating process as well. As the DK-DEM also is part of the Danish Basic Data program, an investigation of the policies regarding basic data and the maintenance of it is performed, see Section 1.1, *Basic data in Denmark*. Other elevation data sources could also be used in the updating process and for this reason they will be investigated, see Section 1.2, *Sources of elevation data*. This section leads to the conclusion that the use of UAV data is a good solution, which is why an investigation into the use of UAVs and availability of UAV data in Denmark will be investigated, see Section 1.3, *UAVs in Denmark*. As UAV data will be used for the updating process a dataset for a case area, appointed by SDFE, will be used and is described in Section 1.4, *Case area - Odense south-east*. A photogrammetric Digital Terrain Model (DTM) based on UAV data could be used in the updating process. For this reason a paper, which investigates the accuracy of photogrammetric DTMs, will be summarised and discussed, see Section 1.5, *Issues regarding photogrammetric based DTMs*. In the end of the introduction, several problems concerning the updating process will be discussed, which leads to an initial problem statement, see Section 1.6, *Project focus and initial problem statement*.

1.1 Basic data in Denmark

The world is dynamic due to both natural and man-made changes. This entails that geospatial data in a similar way have to be dynamic, as the data must represent the current state of the world to be most useful. In Denmark free, current and homogeneous geospatial data is

secured through the political agreed strategy “Gode grunddata til alle” (Basic data for everyone). In addition to geospatial data this strategy encompass a wide array of other datasets such as individuals, businesses, addresses, property, etc. Amongst others the strategy is initially focused on geospatial data, but in general the goal of the strategy is:

- That basic data should be as accurate, adequate and updated to a degree which is deemed practicable
- That all authorities must apply the public basic data
- That the basic data as far as possible must be freely available to businesses and citizens - excluding personal sensitive information
- That basic data are to be distributed in an efficient way that meets the users’ needs. [Finansministeriet, 2012, p. 6]

For geospatial data these goals are implemented by SDFE and GeoDanmark which is an association where the 99 members represents all the municipalities and SDFE. [GeoDanmark, 2017] The geospatial data collected in Denmark covers a broad selection of data layers and types, but generally the data is split into two categories; GeoDanmark data and the DK-DEM. GeoDanmark data is comprised of vector data such as buildings, roads, technical installations, forest, etc. Additionally to vector data, multispectral orthorectified image mosaics are included in GeoDanmark data.

The DK-DEM consists of several different products such as:

- DK-DEM/PointCloud
- DK-DEM/Terrain
- DK-DEM/Surface

The DK-DEM/PointCloud can be seen as the “raw” LiDAR data whereas the other products are derived from it. [SDFE, 2015]

1.1.1 Maintenance of basic data

The production and maintenance of GeoDanmark data is done as a collaboration between GeoDanmark and SDFE. The roles are divided such that GeoDanmark and the represented municipalities provide information about changes and SDFE is the data-producer. The change registration is governed through a process called “løbende sagsorienteret ajourføring” (ongoing case-oriented updating). This imply that each municipality is obligated to register changes during casework dealing with for example building permits, road work, etc. The registration is then continuously reported to SDFE, which in turn update the corresponding vector data through a photogrammetric process using the annual nationwide image coverage. SDFE’s role is moreover to handle tasks regarding public procurements, formation of contracts and quality control. The expenses for maintenance and production of new GeoDanmark data is equally split between the municipalities and SDFE. [GeoDanmark, 2014b, p. 3-8]

Only SDFE is responsible for maintaining and updating the DK-DEM. The first airborne LiDAR based DK-DEM was produced during the period 2005-2007. At the time of the models release it was not freely available to the public, but with the enacted basic data strategy it was made available for commercial and private use the 1st of January 2013. Another consequence of the basic data strategy was that the DK-DEM underwent a total update during the period 2014-2015, using newly produced airborne LiDAR data. Not only was the elevation model updated but the horizontal accuracy was improved by a factor of more than four. Additionally the point density was increased by a factor of about nine. [SDFE, 2009, p. 6] [SDFE, 2015, p. 6] Currently there are no official policy for how often a total update of the DK-DEM is to be performed. It is however expected by SDFE, that the new 2015 DK-DEM will be a dynamic entity receiving continuous improvements and updates when needed. This is in contradiction with how the 2007 elevation model was viewed, as this was seen as a static dataset which never was supposed to receive any form of improvement or update. Presently a set of rules describing how the DK-DEM will be maintained is non existent and SDFE has at the time of writing only performed improvements, such as removing points on power lines and smoothing of water surfaces on lakes and the sea, etc. [SDFE, 2015, p. 7] It is expected that SDFE's intention is to make local updates whenever changes, reported by the municipalities, to some extent affects the terrain or surface of the earth. SDFE want to update all products of the DK-DEM, but their main focus is the DK-DEM/Terrain. This is because a DTM is often the most requested model when it comes to project design. For these reasons updating the DK-DEM/Terrain will primarily be focused on.

When wanting to update the DK-DEM it is essential to consider which primary data source that should be used in the updating process. SDFE have contemplated the idea of using UAV data, but additional elevation data sources exist which may or may not be usable. For this reason the potential data sources are investigated, which will make it possible to determine which elevation data sources that could be used in the process of making local updates to the DK-DEM.

1.2 Sources of elevation data

There are many different sources of elevation data, but the following five are seen as the ones which have the potential of being used to update the DK-DEM:

- Terrestrial surveying
- Design plans
- UAV surveying
- Aerial photography
- Airborne LiDAR scanning

Terrestrial surveying methods can potentially be very precise, within millimetres depending on the method and used equipment. With a total station the points can be precisely measured, and a dense mesh of triangles can be generated. The use of total stations for this type of

survey requires a lot of planning and work. To minimise both planning and time consumption a GNSS receiver can alternatively be used, but it comes with the disadvantage of being less precise and depends on having a mostly open view to the sky. Terrestrial laser scanning is also an option for collecting elevation data, but the time consumption for data collection is also high for larger areas, caused by many set-ups depending on the range and measurements of control points. The advantages of using a total station and a GNSS receiver is that the intensity of measurements can be varied in accordance with the terrain. With terrestrial laser scanning a full point cloud is measured including non-terrain points, which therefore require filtering. Using a total station or laser scanner makes it possible to survey forested and vegetated areas without too much extra effort. When the elevation data is collected, generation of a DEM does not require much processing, as it only is a matter of using triangulation methods on the measured points. However data from laser scanning need some kind of processing before a DEM can be generated. The primary disadvantages of terrestrial surveying is that the time required to survey many square meters is high and the work is very labour intensive, even with a GNSS receiver. [Nelson et al., 2009, p. 65-66] If the data has to be collected for the purpose of making nationwide local updates, the method is thought to be impractical to use for updating the DK-DEM. However, if data is available for some reason, maybe from as-built surveying or situation plans, the data sources are suitable for the purpose.

Design plans are used during construction projects and act as a communication link between project planner/engineer and entrepreneur. Design plans are an essential tool in most if not any construction process, as it prevents any misconceptions and keeps construction on the right track. The plans can consist of a 3D model specifying the project which is to be constructed. Construction plans can be fed to advanced machine control systems, that incorporates precise GNSS technology. This makes it possible for the operators of excavators and graders etc. to perform construction with an accuracy which matches that of a GNSS receiver. [Leica, 2017] If state of the art technology is used, it is plausible that design plans are an accurate source of elevation data when construction ends. The disadvantage of using design plans is that they only represents constructed areas, which means that relaying only on design plans for new elevation data may cause other terrain changes to be neglected. To make sure the plans represent the actual elevations, it would be necessary to measure check points, with either a totalstation or a GNSS receiver. As some terrain changes may be overlooked, the use of design plans may not be sufficient for updating the DK-DEM.

UAV surveying is used in a wide array of professions and is quite diverse when it comes to types of equipment, sensors and processing software. In this case the focus is on UAVs weighing less than 25 kg which carries a true color (RGB) digital camera. The raw product of an UAV with an integrated camera is images captured of the earth at an usual altitude of 120 m or less. [Danish Transport, Construction and Housing Authority, 2017] From the images a geo-referenced point cloud can be generated by using photogrammetric techniques and ground control points (GCPs). The dense point cloud can be transformed into a DEM by the use of triangulation methods. Surveying an area with an UAV can be done quite rapidly and effectively, but the time needed also depend on the area that is to be surveyed, as several

flights may be needed which may extend the need for GCPs. Surveying with an UAV is quite flexible as the flight mostly depends on weather conditions. Typically UAVs come with flight planning software which makes planning of the flight effortless. Because of the diversity of UAVs the accuracy can vary depending on the type of used equipment, but also the used flight parameters (flight altitude, overlap, etc.) and texture of the area. Even though the accuracy may vary, different investigations have shown that by using UAV data to generate a DEM, a vertical accuracy close to the specification (5 cm) of the 2015 DK-DEM is achievable for open areas (terrain). [SDFE, 2015, p. 6] [Larsen et al., 2016] [Kršák et al., 2015] [Küng, 2016]

Aerial photography is similar to UAV surveying as the raw and derived products are very alike. Aerial photography is usually done at a much larger scale compared to UAV surveying and the images cover much larger areas and are captured by a calibrated large format camera from a manned aircraft. The images are normally captured at an altitude of 1500 m. [Nelson et al., 2009, p. 71-73]. As mentioned annual aerial photography is performed in Denmark, but this is not for the purpose of creating elevation data but to create orthophotos. SDFE do create DEMs based on the annual captured photographs, but these are not publicly available. A downside of aerial photography is also the fact that the photographs are annual, meaning that they might not represent the current state of an area.

Airborne LiDAR is a successful way of collecting elevation data and even though SDFE wish to update the DK-DEM by other means, the method is not to be neglected. The raw elevation data from airborne LiDAR is a point cloud which a DSM and a DTM can be derived from by the use of triangulation and classification methods. Airborne LiDAR scanning is performed from a manned aircraft with an active sensor at an altitude between 200 to 4,000 meters depending on the purpose of the dataset. An important feature of LiDAR is that several reflections can occur for each laser beam, because the radius of the laser beam, when it hits the ground, is around 20-30 cm for a typical set-up. This makes it possible to collect data below vegetation, which is very useful when dealing with DEMs. [Balstrøm et al., 2010, pp. 102-104] [Vosselman, 2008, pp. 609-612]

It is thought that doing airborne LiDAR for small local areas is impractical because of the needed planning, resources and expenses. Another possibility lies in the use of LiDAR on an UAV, which already exists and is an attractive technique of collecting elevation data of a bounded area. The existing set-ups are expensive and the technology is not widespread. With development of the technique it is very likely that UAV LiDAR will be able to create high quality elevation data, but at this time the technique is not available. [Rising, 2016]

A DEM based on the annual aerial photography could potentially be used as part of the updating process, but using UAV data for the updating process is found to be most intriguing. SDFE also suggests this solution and the other sources of elevation data, besides the annual aerial photography, have several disadvantages. These disadvantages make the data unsuitable for a more general approach when locally updating the DK-DEM. As UAV data will be focused on, the extent of the use of UAVs in Denmark and the availability of UAV data will be investigated in the following section.

1.3 UAVs in Denmark

The proposal of using UAVs is well in line with the ambition of the current government. The ambition is that authorities should rely on UAV technology if it is deemed applicable. This aspiration is mentioned in the government's strategy "Danmarks Dronestrategi" (Denmarks Drone Strategy) which concern the national development and use of UAV technology. In the strategy it is stated that:

There is considerable potential in increasing the use of drones in the public sector as the technology advances and knowledge about application possibilities disseminates. [The Danish Government, 2016, p. 30]

One of the reasons why the government sees this potential is that an increase in companies using UAV technology has been observed. This is based on the number of professional UAV operators, which presently is above 340 compared to just 10 in 2014, and a report made by The Danish Technological Institute for The Danish Agency for Science and Higher Education. [The Danish Government, 2016, p. 10] The report maps the global and national market for UAVs and the companies using them. In march 2016 there were about 294 Danish companies which in some form utilises UAV technology. Most of these companies are based in media and film production (45%), but 25 companies are architectural and engineering firms (9%). The architectural and engineering firms are typically surveying and engineering consultancies, which uses UAV technology to collect geospatial data, inspections and monitoring. [Teknologisk Institut, 2016, p. 17] Based on this information it can be said that the use and selling of UAV based services is established in Denmark. Because of this it is not a questions of whether or not these services can be provided, but more a question of how the public sector can find applications which utilises these services in a way that will make processes more efficient.

UAV technology is already being adopted by Danish authorities and has been for several years. Some municipalities are even investing in their own cheap UAVs for inspections and mapping tasks. [Bækthøj, 2016] [Raasthøj, 2015] Another Danish authority which have found excellent uses of UAVs is The Danish Road Directorate. They are using UAVs for different tasks such as counting traffic, investigating traffic patterns and driver behavior. [Vejdirektoratet, 2015] For major road projects they additionally use UAVs to monitor the construction, so they can make sure it is kept within the land that is expropriated for the purpose. The latter task is put out to tender and typically performed by an engineering or surveying company. [Vejdirektoratet, 2016]

As described in this section there generally is a big potential for the use of UAVs in the private and public sector. This includes SDFE as they could get UAV data from both municipalities and other authorities such as the Danish Road Directorate. Municipalities give regular information about changes because of the ongoing case-oriented updates which is implemented through the policies regarding the maintenance of GeoDanmark data. This information can be used

as a tool for deciding whether or not a change is substantial enough to make a local update to the DK-DEM.

As mentioned it is likely that UAV data and thereby derived DEMs is available, at least for major road construction, which entail that the acquisition of data for the updating process should not be an issue. The project group has been informed that SDFE have access to an UAV dataset flown in an area that has changed since data was acquired for the 2015 DK-DEM. The following section will describe this area and its changes, which will act as a case area in this project.

1.4 Case area - Odense south-east

On the project groups behalf SDFE requested a UAV dataset from GeoFyn A/S. GeoFyn A/S is a joint-stock company owned by ten municipalities on Fyn and support the municipalities in the acquisition, maintenance, use and display of spatial data. [GeoFyn A/S, 2017]

The UAV data is based on images captured on the 28th of November 2016 from a fixed wing UAV and it covers the area around Odense south-east (SE). In the area a new highway exit/entrance has been constructed, see Figure 1.1. SDFE find the area to be a good representation of a typical change in the terrain and the UAV dataset is what can be expected to be delivered from other data suppliers. The dataset from GeoFyn consist of a dense point cloud, a raster DSM, a raster DTM and an orthophoto.



Figure 1.1: Aerial photo of Odense SE [The Danish Road Directorate, 2016]

The new construction project involves the new highway exit/entrance number 50, along with new roads connecting the city and the future hospital to the highway. A map of the construction project is shown in Figure 1.2. The highway construction is carried out by The Danish Road Directorate, while Odense municipality is responsible for the construction of the connecting roads.

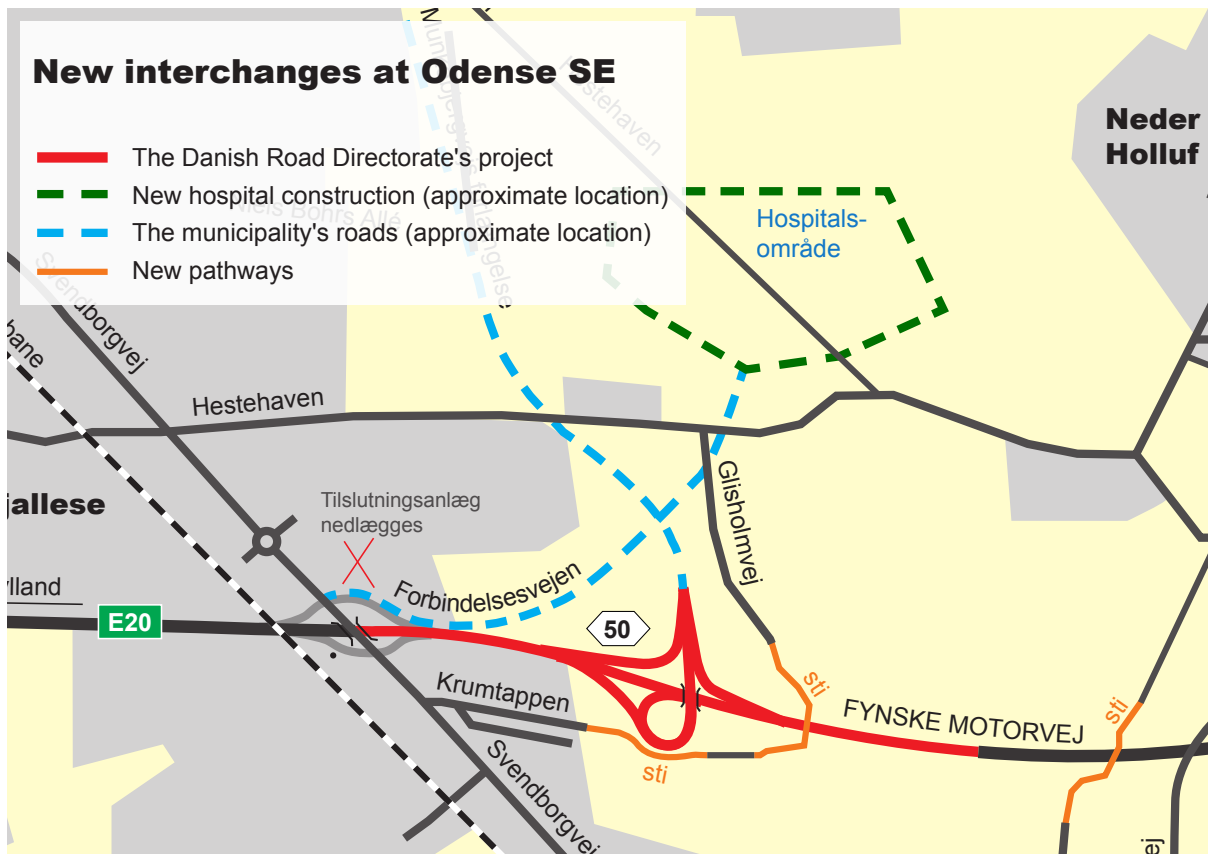


Figure 1.2: Map showing the construction project around Odense SE [Vejdirektoratet, 2014]

The updating process is inevitably dependent on the area that is to be updated, hence a developed process will not necessarily show good results for all UAV datasets. This means that the updating process is not applicable for all areas and it might have to be adjusted according to it.

The raster DTM in the UAV dataset could be used to update the DK-DEM/Terrain, but the project group does not see this as an option. The project group holds this view, because it is experienced that the estimated parts of such a model are inaccurate, especially regarding vegetated areas. To support this view the following section will investigate several problems regarding photogrammetric based DTMs.

1.5 Issues regarding photogrammetric based DTMs

A project made by Professor Joachim Höhle at Aalborg University, concerning photogrammetric based DTMs, was commissioned by SDFE and will be summarised in this section. The project investigated how well a DTM based on aerial photography could be produced and the motivation was that the generated DTM could be used to locally update the 2007 DK-DEM/Terrain. The 2007 DK-DEM/Terrain did however never get updated with photogrammetric based DTMs.

Seven different tests (A to G) were performed, but only test D will be focused on, as the produced DTM is similar to the 2007 DK-DEM/Terrain (1.6 m grid size). Test D did furthermore not involve any manual editing or external data, which was the case for some of the other tests. Several professional photogrammetric software packages and filtering techniques from Trimble Inpho were used to produce the DTM.

Checkpoints in three categories, open land, built-up and forested area where used to check the accuracy of the generated DTMs . [Höhle, 2009, p. 31] Several accuracy measures were derived from the vertical error ($\Delta h = h_{\text{DTM}} - h_{\text{checkpoint}}$) and compared to those of the LiDAR DTM. Besides traditional accuracy measures (mean, standard deviation and root mean square error) additional robust statistical measures were used because:

“DEMs derived by digital photogrammetry and laser scanning very often have outliers and a non-normal distribution of errors” [Höhle, 2009, p. 17]

The robust measures are the median and the Normalized Median Absolute Deviation (NMAD). The median is the middle value of the elevation errors and the NMAD value is a robust estimation of the standard deviation of the normal distribution. [Höhle, 2009, pp. 17-18] In Figure 1.3 the comparison between the accuracy measures can be seen.

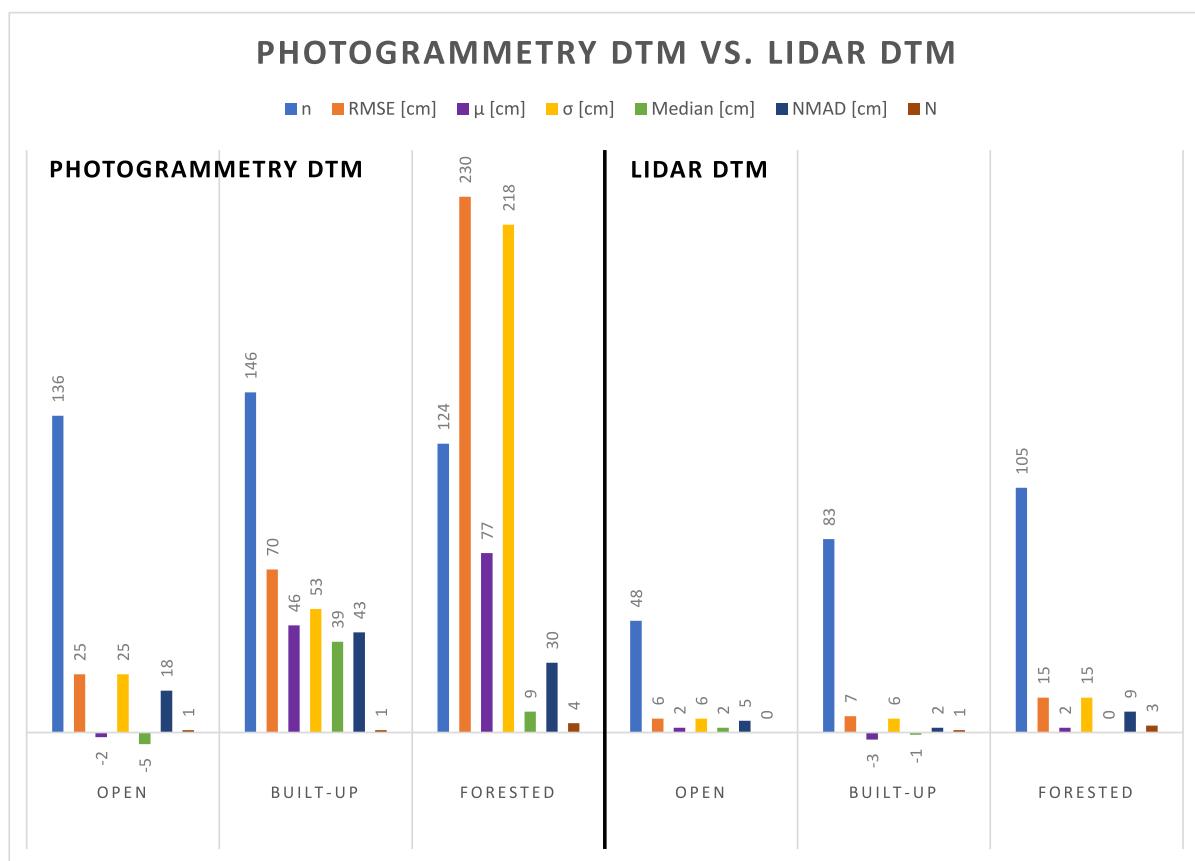


Figure 1.3: The left side of the figure shows the accuracy measures of the photogrammetric 1.6 m grid DTM. The right side shows the accuracy measures of the 1.6 m grid DK-DEM/Terrain. The lower case “n” shows the number of check points and the upper case “N” is the number of outliers ($3 \times \text{RMSE}$). The graph is based on [Höhle, 2009, Table 7. p. 31](Photogrammetry DTM) and [Höhle, 2009, Table 7. p. 47](LiDAR DTM)

From Figure 1.3 it can be seen that the two DTMs performs quite differently in the three different checkpoint categories. Generally the LiDAR DTM has a higher accuracy in all categories but both DTMs show that especially forested areas can be challenging as both DTMs have the lowest accuracy in this category. For the photogrammetric DTM it clearly can be seen that the robust measures are needed. The NMAD is considerably lower than the standard deviation, which implies that the estimated surface of the photogrammetric DTM is poor in forested areas. In open areas both DTMs has the best accuracy, but the LiDAR DTM is still three times better compared to the photogrammetric DTM, when looking at the NMAD values.

The reason for the variation between the DTMs is that there is a fundamental difference between how the elevation data is obtained. As mentioned several reflections can occur with airborne LiDAR, which makes it possible to generate points on both the terrain and treetops, which is very useful when generating DTMs. [Chen et al., 01-14-2017, pp. 3-4] When it comes to photogrammetric based DTMs, measuring terrain points can be difficult if not impossible for a vegetated area. This is because the generation of a photogrammetric DTM is pixel based and requires that each matched pixel must represent the ground when doing the image point matching, else the terrain point can not be extracted. As a consequence it is more likely to get points on the surface of vegetation rather than on the terrain below. This makes the estimation of terrain, at vegetated areas, during DTM generation very challenging. [El-Ashmawy, 01-14-2017, p. 162]

It is quite clear based on Höhle's project that a photogrammetric based DTM may not be very accurate, which is in line with the project group's experiences. The project uses aerial photography, which has a bigger ground sampling distance (GSD) compared to images captured by an UAV. The smaller GSD of the UAV captured images could potentially make it possible to get more ground points for vegetated areas as the images would be more detailed. However it is still expected that a DTM based on UAV data would be affected by similar issues as the one generated from aerial photography.

Based on the discussion above the directly use of a photogrammetric based DTM for updating the DK-DEM/Terrain is not seen as a good approach. For this reason another approach is needed, which will be suggested in the following section. This leads to an initial problem statement.

1.6 Project focus and initial problem statement

In this section the primary focus of the project will be described based on the investigated subjects through the introduction. When the focus of the project has been detailed a initial problem statement will be formed.

From the investigation of basic data in Denmark it was found that the maintenance of the DK-DEM is performed by SDFE and that currently no plans for updating the 2015 DK-DEM has been established. Even though no plans exist, it is SDFE's intention to make local updates

to the DK-DEM. Based on a discussion with SDFE, it was found that they are mostly interested in updating the DK-DEM/Terrain and for this reason the project will focus on making updates to this model.

Different elevation data sources were investigated, but UAV data were found to be most intriguing. A UAV dataset was delivered by GeoFyn A/S and will be the primary data source for updating the DK-DEM/Terrain.

After discussing different approaches with SDFE it was found that they are interested in having an algorithm which identifies an actual change of the terrain between the DK-DEM and UAV data. Such an algorithm can depend on several methods, variables and data. Currently it is uncertain how the algorithm will work, but the fundamental workflow is showed in Figure 1.4. The figure shows the full workflow, but the present project will not necessary focus on all parts of the workflow. A choice regarding which parts of the workflow that is to be focused on, will be made during the problem analysis.

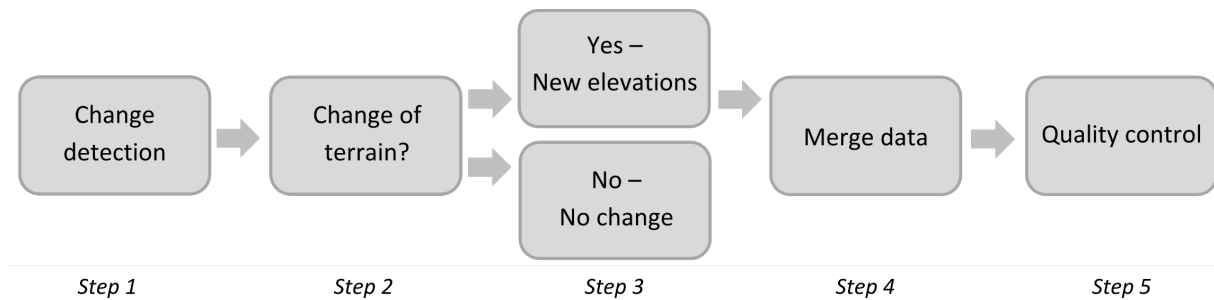


Figure 1.4: Fundamental workflow of the proposed algorithm

The first step concerns *Change detection* and may encompass several processes and data sources. The step will determine changes of the case area based on the available data sources. The second step *Change of terrain?* will determine to which degree the detected changes in the UAV data can be seen as terrain. As UAV data represents the surface of the earth it can be challenging to determine if the detected change concern the terrain or if it is caused by other factors such as grown vegetation, new or removed features or inaccuracies in the model. SDFE do not see a black and white solution to this issue, as situations where “close to terrain”, for example at areas with low vegetation, might represent the new terrain better than the existing model. Therefore this step can be seen as a categorisation where the UAV data get ranked on a scale from “good” to “bad” terrain data. Figure 1.5 illustrates this scale with four steps going from “good” to “bad” terrain elevation data, additionally Figure 1.6 shows an example of how the different features on the ground could be ranked.



Figure 1.5: Scale with four steps going from “good” terrain data to “bad” terrain data

As UAV data basically represent the surface of an area, Figure 1.6 illustrates data as a DSM. Six features are shown in the figure, where each feature have been ranked according to the scale. As seen the ground or road is ranked with 1, hence it is seen as terrain data. Grass and low vegetation is ranked with respectively 2 and 3, hence it is not terrain data, but in some cases it might be of interest. If a terrain change of a couple of meters have been detected, it might be of interest to update with the new data even though the area is grass or low vegetation. Finally high vegetation, buildings, cars and other object is ranked with 4, as there is no interest in these areas when it comes to finding areas that represent terrain.

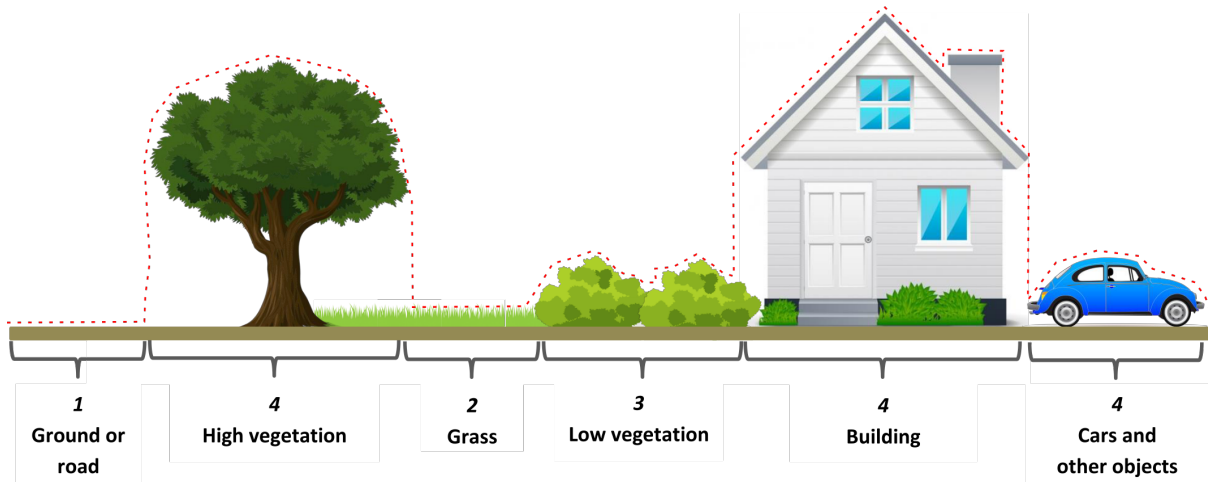


Figure 1.6: Features ranked according to the scale

Figure 1.6 illustrates different features, but quality of the data is not taken into account. SDFE do not want to update the DK-DEM/Terrain at all costs, so if there is doubt about the quality of the UAV data, they will rather keep existing data over new inaccurate data. The quality of the data will be considered during the development of the algorithm.

As the third step illustrates in Figure 1.4, a change of the terrain will result in new elevations and hence an update will be performed for the changed areas. On the other hand, no updates will be made if the change is caused by non terrain elements. Step four *Merge data* is the process of replacing the old terrain elevation data with the new. Lastly a *Quality control* is carried out which can correct errors and edge effects.

The proposed workflow of the algorithm will act as a starting point and the order and nature of the steps may be altered as the development of the algorithm progresses. To develop the algorithm a deeper understanding of the methods which can be used and the available data is needed. This leads to the following initial problem statement:

Which data and methods can be used to update the Danish Elevation Model?

To answer the initial problem statement a problem analysis will be made. The problem analysis will investigate and describe the available data and potential methods which could be incorporated in the algorithm during the different steps of the workflow. As the steps are

dependent on the used data, such as the UAV data and the DK-DEM, an understanding of the data is needed. Each steps in the workflow require one or several methods in order to achieve the desired result. For this reason an investigation of which methods that can be used during the steps will be carried out. As the introduction has outlined the main concerns are to identify changes and terrain, hence step one and two. The methods that will be investigated will therefore mostly concerns these steps.

The obtained knowledge in the problem analysis will result in a choice regarding which data and methods that will act as the foundation of the algorithm. This will lead to a problem statement. In the following chapter the method to answering the initial problem statement will be described.

Problem analysis method 2

The present chapter will introduce the method used to answer the initial problem statement. The structure of and approach to the problem analyses will be described and the method of the individual analyses will be explained. Figure 2.1 shows a diagram of the structure that the problem analysis will follow.

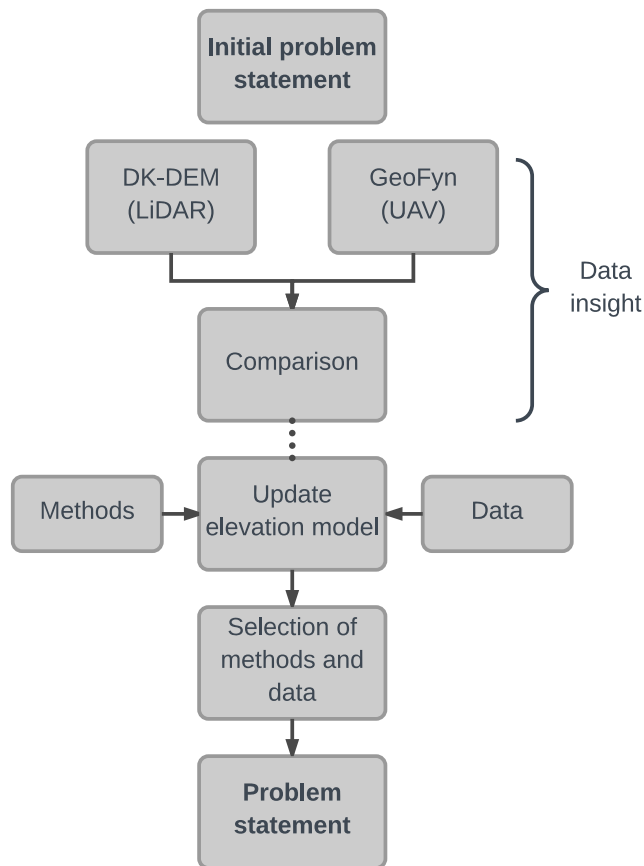


Figure 2.1: Structure diagram of problem analysis

The problem analysis will answer the initial problem statement, which is the starting point of the diagram. Generally seen the problem analysis consist of three steps. The first two steps deal with data insight, which creates familiarity with the data sources. First the products of the DK-DEM and the data from GeoFyn will be described and secondly the two datasets will be compared. Thirdly a selection of other datasets and methods, which possibly can be

incorporated in the algorithm for updating the elevation model, will be discussed. Through the three steps a number of selections and rejections regarding data and methods will be made and a final chapter will sum up the findings and lead up to the problem statement.

The method of the three steps will be described in the following sections, which imply what the analysis is about and why it is relevant. The sections will shortly describe how the analysis is carried out, but a further description is placed in the relevant chapter.

2.1 Elevation models

The first step of the problem analysis concern the primary elevation models and is split into an analysis of the DK-DEM data and one of GeoFyn data. The analyses can be found in Chapter 3, *The Danish Elevation Model* and Chapter 4, *GeoFyn data*. The study of the DEMs will clarify the methods used when collection the data, the precision and the differences between the datasets. The DEMs used in the present project is the DK-DEM from 2015, which contain a point cloud (DK-DEM/PointCloud), a DSM (DK-DEM/Surface) and a DTM (DK-DEM/Terrain), and the data from GeoFyn, which also contain a point cloud, a DSM and a DTM. The datasets used in the project is collected by other parties, so the description of the data is based on documentation from the data collector.

The analyses of the two data sources are carried out to get familiar with the data and to clarify what can be expected by the data. The analyses will give a basic knowledge before the data sources are compared.

The documentation of the DK-DEM consist of product specifications for all three products and a quality assessment of the point cloud.

The documentation of the data from GeoFyn is based on the quality report that is generated during data processing in Pix4D. Furthermore GeoFyn have assisted with additional information about flight lines and GCPs. The information on the data processing in Pix4D is limited, especially when it comes to generation of the DSM and DTM.

2.2 Comparison of elevation models

Step two of the problem analysis encompass comparisons of the primary elevation data. The four elevation models DK-DEM/Surface, DK-DEM/Terrain, GeoFyn DSM and GeoFyn DTM will be compared where it is relevant. The comparisons are found in Chapter 5, *DEM comparisons*. When two models are compared the difference between two raster model are calculated and then illustrated and commented on. The subtraction of the raster models are carried out in ESRI ArcMap and the differences are showed on a map. Furthermore the individual models are shown as hillshade models which makes the variation of the surface clearer.

Table 2.1 shows a matrix where three comparisons are marked. The first comparison is between the DK-DEM/Surface and the DK-DEM/Terrain and the result will illustrate how

successful the filtering for the DTM is. The comparison will also show what can be expected by a LiDAR based model.

The second comparison is between the GeoFyn DSM and the GeoFyn DTM, which has the same purpose as the first comparison, but it will also enhance some of the difference between a LiDAR and a UAV based model.

Lastly a comparison between the DK-DEM/Surface and the GeoFyn DSM is carried out, where the difference between the two models will illustrate where the area has changed between the time of data collection.

Three comparisons have been omitted as seen in the matrix. It is assessed that these three comparisons will not contribute to a better understanding of the data.

Table 2.1: Comparisons between elevation models

	DK-DEM/ Terrain	DK-DEM/ Surface	GeoFyn DTM	GeoFyn DSM
DK-DEM/Terrain				
DK-DEM/Surface	X			
GeoFyn DTM				
GeoFyn DSM		X	X	

2.3 Methods and supporting datasets

As the two primary datasets have been presented and compared, step three of the problem analysis can be carried out. This step comprise an investigation of a number of methods and data sources for updating the DK-DEM. As described in the Chapter 1, *Introduction* the main concern of the updating process is detecting change and identifying terrain, why the investigated methods will focus on these subject. This part of the problem analysis have the purpose of investigating and selecting some relevant methods, as numerous exists. Through the introduction a number of sources of elevation data was listed, but the problem analysis will also involve other types of data which can be used and support the methods.

Step three is spilt into three sections. Firstly methods to assess the quality of the UAV data will be investigated. The aim is to find a method to quantify the quality of the data, as only assumptions has been made about the quality. Different methods and their applicability for this project will be described.

Section two concern methods of categorising the UAV data to identify terrain. Several methods exists, so a selection of three methods will be investigated. The methods are vector analysis, point cloud filtering and image classification and for each of the methods several approaches exist. These methods will be described and their suitability for the use in this project will be evaluated.

The third section concern the methods used for change detection. Change detection cover the comparison of the two datasets to see if and where changes has happened. The issues lies in how to decide whether a detected change is real or caused by inaccuracies in the two datasets. Different approaches for evaluation the change will be discussed.

For some of the analysed methods additional data is needed and while the methods are described the need of supporting data sources will be mentioned.

The Danish Elevation Model 3

The Danish Elevation Model (DK-DEM) is central in the present project, as it is the data source that needs updating. This chapter will introduce the data and shortly describe the accuracy of the products. The description is based on product specifications and a quality assessment of the point cloud, which both are written by SDFE who is responsible for the production and maintenance of the DK-DEM.

The newest DK-DEM was released in December 2015 and the raw data is collected with airborne LiDAR. [Flatman et al., 2016, p. 3] The primary products of the DK-DEM are; DK-DEM/PointCloud, DK-DEM/Surface and DK-DEM/Terrain. The DK-DEM/PointCloud is the foundation of the other two products. The products of the DK-DEM can be seen in Figure 3.1.

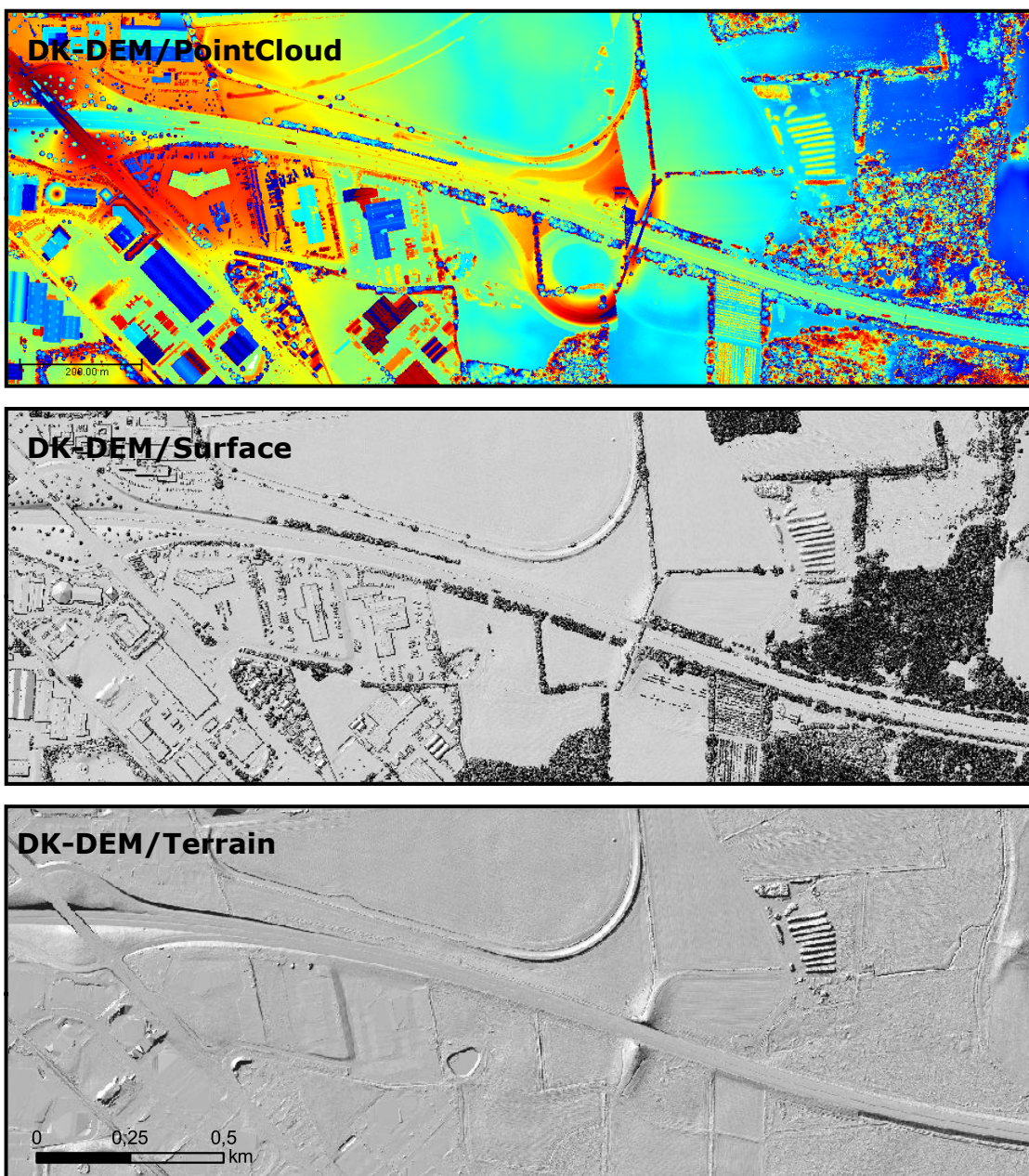
The DK-DEM/PointCloud have a point density of 4-5 points per square meter (ppm^2) and the specification promises a horizontal accuracy of 0.15 m and vertical accuracy of 0.05 m. The DK-DEM/PointCloud is stored as compressed .LAZ files in blocks of 1×1 km. [SDFE, 2015, p. 5-6] The .LAZ format follows the standard of The American Society for Photogrammetry & Remote Sensing (ASPRS) laser (.LAS) format. The .LAS format is non-proprietary and ensures that the exchange of data between users happen in a common format, which different LiDAR hardware and software can read and output. [ASPRS, 2011, p. 2] Each stored point have information about: [SDFE, 2015, p. 4]

- Elevation
- Class
- Intensity
- Scanline ID
- Position in the reflection signal (first, ..., last reflection)
- RGB-value, for the points collected during daylight

In addition to this information each point is classified according to the classes seen in Table 3.1

Table 3.1: LiDAR Point Classes [SDFE, 2015, p. 4]

Value	Meaning	Note
0	Created, never classified	
1	Surface	
2	Terrain	
3	Low vegetation	0-0.3 m
4	Medium vegetation	0.3-2 m
5	High vegetation	2 m >
6	Buildings	
7	Outliers	Noise
8	Model key points	Used in the production process
17	Bridge	

**Figure 3.1:** DK-DEM/PointCloud and hillshade models of DK-DEM/Surface and DK-DEM/Terrain

In Figure 3.1 a discrepancy between the DK-DEM/PointCloud and the DK-DEM/Surface and DK-DEM/Terrain is seen. It seems as if DK-DEM/PointCloud have received a partial update, as some of the new highway exit is seen. The partial update looks peculiar and ends abruptly which indicates some form of error is present in the DK-DEM/PointCloud. After discussing the error with SDFE, it was found that a partial update was made with data obtained at a later time. The new data lies together with the data first collected, and the new data can be removed by filtering the block. The filtering is done by using the LAStools software suite, which have a tool for filtering points in the .LAS point cloud according to the stored point information. Every point in the updated data have the Scanline ID: 11152, and the points can thereby be removed from the block by using LAStools and dropping the points with this Scanline ID.

The grid size of both models is 0.4 m and they keep the same accuracy as the point cloud. Both models are homogeneous with the exception of areas covered by water or other non-reflective surfaces. At these areas the point density of the DK-DEM/PointCloud is typically less and as a consequence the triangulations are accordingly bigger. For the DK-DEM/Terrain a lower point density is also the case for areas covered with vegetation. The horizontal reference system for the DK-DEM products is ETRS89, UTM zone 32N and the vertical datum is DVR90. [SDFE, 2015]

A quality control quality control of the DK-DEM/PointCloud has been performed by SDFE, which the specified accuracies are based on. In the following section a description of the quality control performed on the DK-DEM/PointCloud will be made.

3.1 Quality control of the DK-DEM/PointCloud

Only the DK-DEM/PointCloud is quality checked as all other products of the DK-DEM are derived from it. This section will make it more clear how the specified accuracies of the DK-DEM were found and which methods that were used in the QC. Several quality controls were performed but only the vertical accuracy and classification control will be described as they seem most relevant.

3.1.1 Vertical accuracy control

The vertical accuracy is controlled by measuring check points on plane horizontal surfaces. It is not mentioned in the quality assessment report how the check points are measured or which accuracy that can be expected from them.

The technique for comparing the checkpoint and the DK-DEM/PointCloud is not detailed, but it is expected that a type of nearest neighbour search is performed, such that the nearest LiDAR point is subtracted from the check point. For each block between 40 and 75 uniformly distributed check points were used. In the quality assessment report an example of the vertical accuracy control of block C10 is given. The statistics are as following: [Flatman et al., 2016, p. 11]

Block C10 - Vertical accuracy control

Check points : 75

Minimum value : -0.065 m

Maximum value : 0.062 m

Mean value : 0.002 m

Median value : 0.003 m

Standard deviation : 0.033 m

As it can be seen the vertical accuracy control of block C10 is within the specification of 0.05 m. The calculation of statistics for block C10 is the only example shown, but it is expected that all blocks are within the specification.

3.1.2 Classification control

Several classification controls were performed, but only the control of vegetation or terrain within buildings and the manual control will be described as this is deemed most relevant.

By using closed building polygons from the GeoDanmark data it is possible to determine if building points are wrongly classified. The points within the checked building polygons should be classified as building and if this is not the case they are reclassified. [Flatman et al., 2016, pp. 23-24]

If building polygons does not have any building points within them, the classification of the points are also checked. In many of the cases the buildings did not exist any more, and for this reason the classification is not incorrect. [Flatman et al., 2016, p. 19]

From the description above it can be seen that the GeoDanmark data can be useful and play a part in the updating process of the DK-DEM/Terrain. As seen from the last example it should be noticed that the GeoDanmark data can be outdated as this was the case with the last control.

In addition to the building polygon control a manual control was performed. The manual control was done as a visual inspection of hillshaded DEMs. The hillshade raster cells are colored according to the predominance of classified points which are within the cell. The visual inspection was done systematically and in grid of 1 × 1 km hillshaded tiles. [Flatman et al., 2016, p. 20]

GeoFyn data 4

A dataset collected with an UAV has been provided by Geo Fyn A/S. This dataset is the primary source of data for updating the DK-DEM and therefore the data collection will be described and the data will be presented. The data provided is based on aerial images taken from an UAV and the images have been processed with the drone-based mapping software Pix4D. The data consist of a point cloud made from photogrammetric methods, a DSM, a DTM and an orthophoto. As the data is collected by GeoFyn the description is based on the quality report generated in Pix4D, see Appendix A, *GeoFyn specification*, and information given by GeoFyn. Furthermore the received data has been investigated in ESRI ArcMap.

GeoFyn A/S have several purposes for the collected data. Large changes of the terrain has happened caused by the new Odense SE highway entrance/exit, which is why they want to generate a DTM covering the area. The purpose of the updated DTM is to calculate the impact of different climate scenarios. Secondly they want to use the UAV data to update vector data for GeoDanmark, where they primarily use the orthophoto. Thirdly they want the data for visualisations and communication. [Communication with Sten Frandsen, GeoFyn, see Appendix A, *GeoFyn specification*]

4.1 Data collection

The UAV used to collect the data was a fixed wing senseFly eBee, see Figure 4.1. As payload the UAV carried a Sony DSC-WX220 18.2 MP camera. The internal camera parameters are seen in Table 4.1.

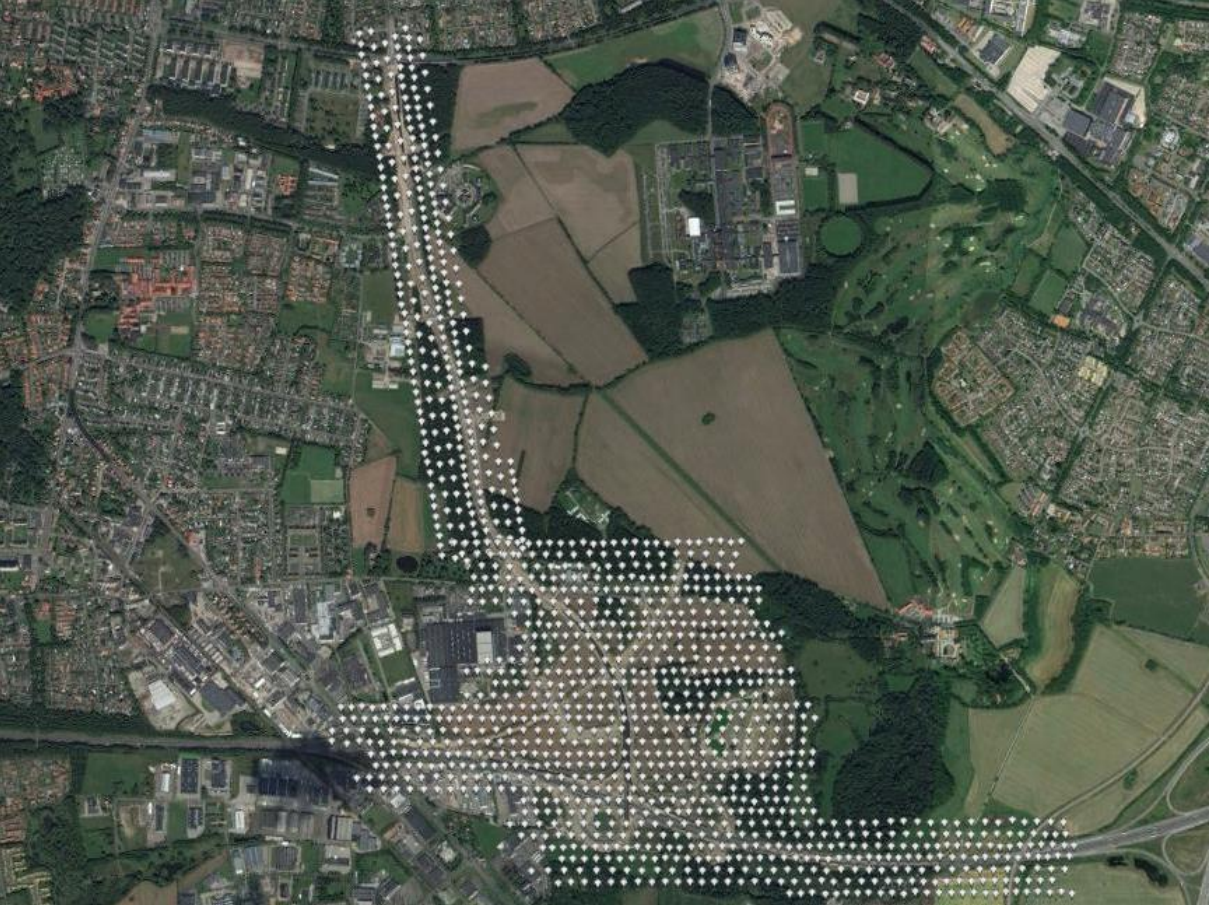


Figure 4.1: SenseFly eBee and Sony DSC-WX220 [GeoNetworking, 2016] [SONY, 2016]

Table 4.1: Internal camera parameters

Resolution width	4896 pixels
Resolution height	3672 pixels
Principal point x'_0	2458.664 pixels
Principal point y'_0	1824.479 pixels
Focal length c	4.546 mm
Pixel size pel'	0.00126 mm

To plan the flight the senseFly eMotion 2 planning software was used. The data collection was split into four flights caused by the size of the area, and the individual flights were carried out in parallel lines. The overlap is unknown, but it is assumed that Pix4D's recommendation of forward overlap of 75% and side overlap of 60% is used. [Pix4D, 2017b] The average altitude of the flight was 100 m which leads to an average GSD of 2.75 cm. In Figure 4.2 the position of the captured images are shown.

**Figure 4.2:** UAV position when the images were captured [Google Earth, 2017]

A total of 1271 images were captured covering an area of 2.85 km². In Pix4D the images are processed through a bundle block adjustment. In a bundle block adjustment the images are tied together using tie points and a 3D model of the surface is created. In the adjustment the object coordinates (model of the surface) and internal orientation parameters are calculated, based on the observations, which is comprised of image coordinates and coordinates of GCPs. The data flow of a bundle block adjustment is shown in Figure 4.3, where the input and output is illustrated.

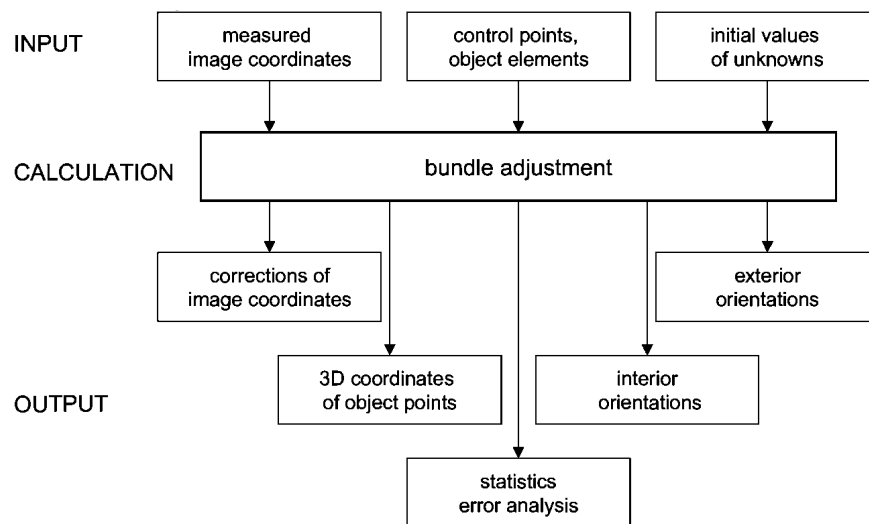


Figure 4.3: Data flow of bundle block adjustment [Luhmann et al., 2006, p 233]

In Pix4D 6,947,371 automatically selected tie points have been identified in the images. 32 GCPs was used and the points was surveyed with RTK GNSS. The GCPs was a mix of white plastic lids and white painted circles, all between 22-29 cm.

4.2 Data presentation

As mention the data received from GeoFyn includes a point cloud, a DSM, a DTM and an orthophoto. The point cloud is the raw dataset from Pix4D and the other three products are also generated in Pix4D.

The point cloud consist of 150 million points and all points have RGB values. The point cloud is shown in Figure 4.4. As seen on the figure the point cloud is incoherent. The areas where Pix4D has not managed to match points in three, which is the minimum in Pix4D, or more images is mainly areas with vegetation.



Figure 4.4: Point cloud from GeoFyn

The DSM is generated automatically in Pix4D where noise filtering and surface smoothing was selected. For surface smoothing the method 'Sharp' was chosen, where edges on for example buildings are preserved and only areas that almost are planar are flattened. The DSM is created as a raster and the cell size is 0.027 m, corresponding to the GSD of the project. The DSM is shown as a hillshade model in Figure 4.5. [Pix4D, 2016]



Figure 4.5: DSM from GeoFyn

The DTM is also generated automatically. The used method is unknown, but it most likely creates an automatic classification and then removes non-terrain data followed by interpolation. It is recommended by Pix4D that the cell size of the DTM is five times the GSD, because the method uses smoothing and it has shown a better result with a lower resolution. This was applied for the DTM and gives a cell size of 0.138 m. [Pix4D, 2017a]

The orthophoto is generated based on the DSM and the images are stitched together and the overlaps are blended. The orthophoto is shown in Figure 4.6.



Figure 4.6: Orthophoto from GeoFyn

DEM comparisons 5

In the present chapter a number of comparisons between the available DEMs will be performed. The comparisons serve the purpose of getting familiar with the data, but also to see how derived DEMs from LiDAR and UAV data differentiate. The comparisons will also give the first picture of where the changes has happened in the case area. The comparisons are carried out so it is possible to see how well objects are removed in the DTMs based on both LiDAR and UAV data. More specifically the comparison between the DSM and DTM from the same data source will show how well vegetation and man-made objects has been removed.

In the following the DEMs in raster format will be compared by showing and commenting on some examples which point out distinct differences. The examples will show two DEMs which has been subtracted from each other and hillshade models have been generated and included for all datasets. The hillshade model enhances the visualisation of a surface and makes the differences between the two models stand out. To make the comparison models cover the same area, they are cropped according to the extent of the UAV point cloud. This means that places where there is no data in the point cloud from GeoFyn, the subtraction will not be performed, hence the area will just be white. The models are generated by tools which are part of the ESRI ArcMap toolbox.

5.1 DK-DEM/Surface – DK-DEM/Terrain

This first comparison between the DK-DEM/Surface and the DK-DEM/Terrain illustrates the two models and the difference between them. The DK-DEM products acts as the reference frame in the project and the goal is to make updates that live up to the standard of these products. The comparison illustrates what is removed in the DTM and gives a general idea of what can be expected by models based on LiDAR data.

Figure 5.1 shows the difference when the DK-DEM/Terrain is subtracted from the DK-DEM/Surface. The segment showed in the figure is chosen as there are areas of vegetation that must be removed in the DTM. When looking at Figure 5.1 it is easy to see the difference between the DSM and DTM based on LiDAR data; the vegetation on the DSM is quite successfully removed in the DTM, leading to a surface of what appears to be terrain. The terrain below vegetation still contain a lot of detail, which can be seen on the square field where small rises in the ground are visible. Another feature to point out is the road which is a smooth surface in both models.

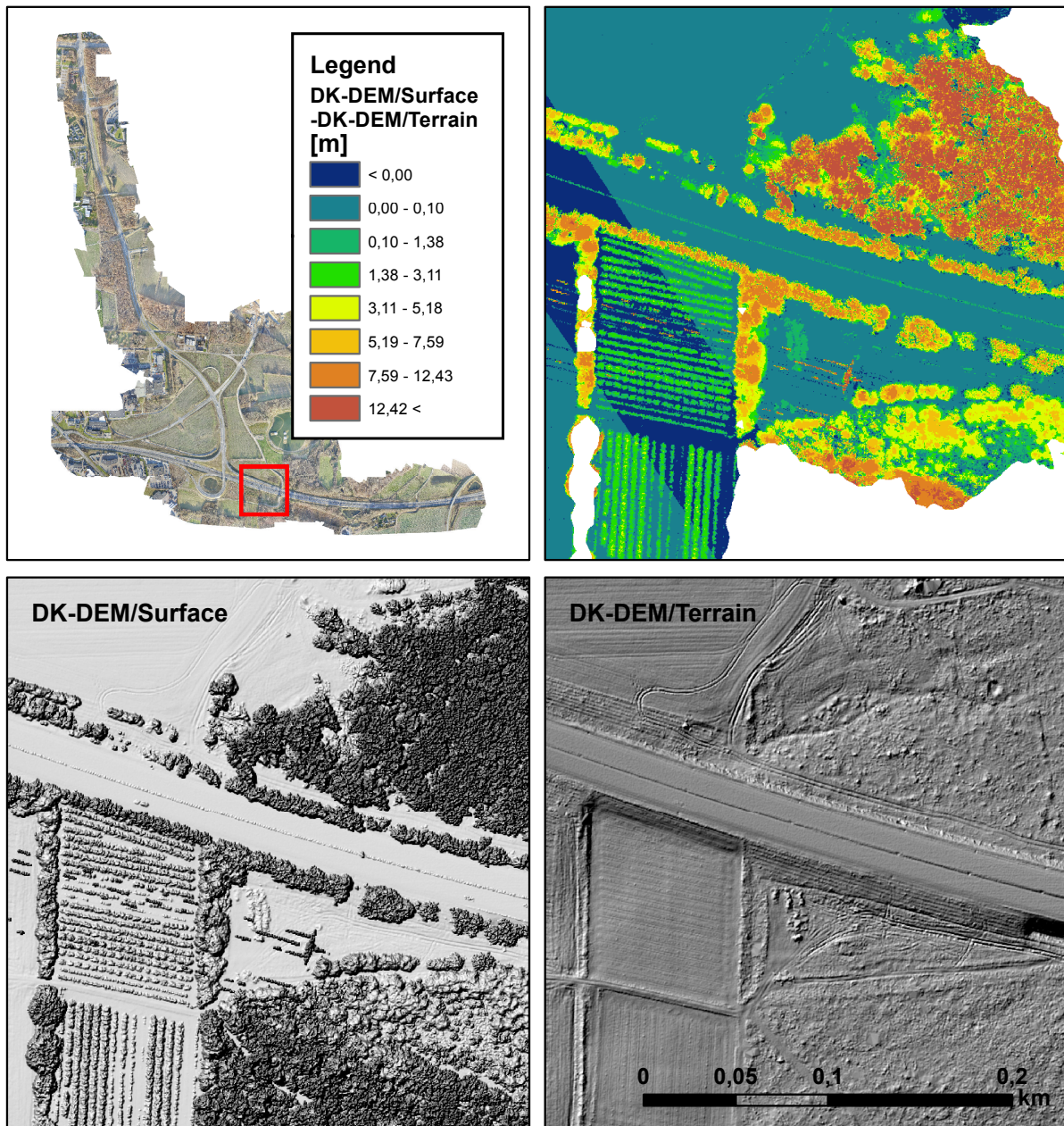


Figure 5.1: Difference between the DK-DEM/Surface and the DK-DEM/Terrain. The red box on the upper left figure shows which area that is depicted on the other figures. The upper right figure shows the two models subtracted from each other. The bottom left and right figures show a hillshade model of the DK-DEM/Surface and the DK-DEM/Terrain respectively

5.2 GeoFyn DSM – GeoFyn DTM

The GeoFyn DSM and the GeoFyn DTM are compared to see how well non terrain objects have been removed and to illustrate some of the issues regarding the data source. The comparison will try to clarify which quality a DTM based on a photogrammetric product has and possibly support the earlier mentioned issues on this topic. The comparison of the GeoFyn DEMs will be held up against the DK-DEM.

Figure 5.2 shows the difference when the GeoFyn DTM is subtracted from the GeoFyn DSM. The segment for this example is chosen as a new walkway bridge is present, which has shown to be challenging for Pix4D, when the GeoFyn DTM was generated.

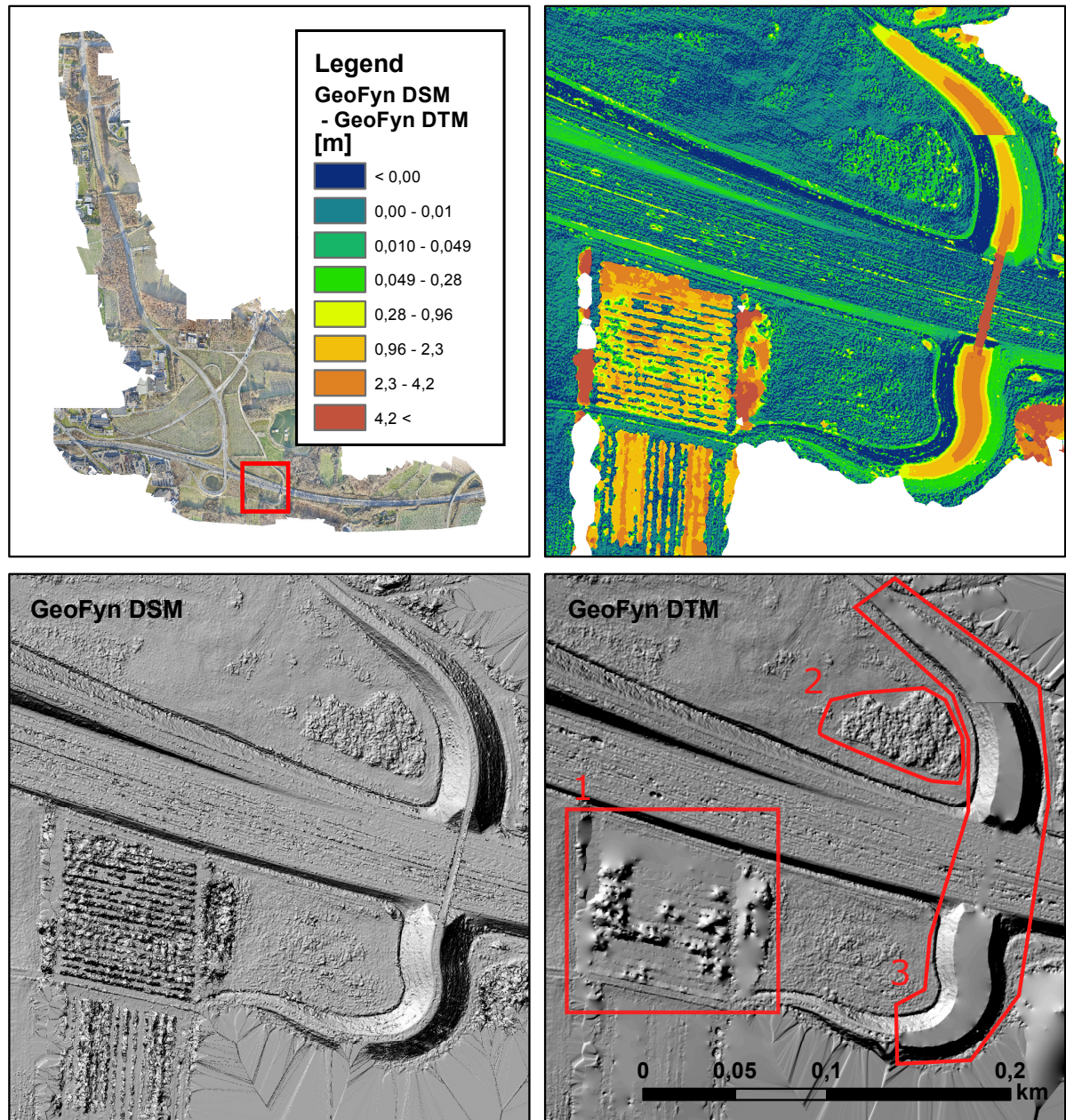


Figure 5.2: Difference between the GeoFyn DSM and the GeoFyn DTM. The red box on the upper left figure shows which area that is depicted on the other figures. The upper right figure shows the two models subtracted from each other. The bottom left and right figures show a hillshade model of the DSM and DTM respectively. The areas marked in red on the lower right figure are places where issues regarding the DTM exist.

The two models are generated by the Pix4D software and as mentioned it is uncertain which parameters affects them as the processing works as a black box. Either way it is possible to compare them and see how they differentiate from each other. As seen in Figure 5.2 filtering of the vegetation, at the lower left field (area 1) and at the upper left side of the walkway bridge (area 2), has been attempted in the DTM. The filtering is clearly not as successful as the one performed on the DK-DEM, as some of the vegetated areas appear to only have been smoothed. Generally some kind of smoothing of the terrain has been performed, as it can be seen that the subtracted terrain surfaces do not equal zero. On the contrary the areas with terrain in the DSM lies above or below the DTM, which shows as the alternating blue and green colours on the terrain. This is not necessary a big issue, but a DTM algorithm which leave out actual terrain when smoothing would be preferred.

It can be seen in Figure 5.2 that the top part of the walkway bridge (area 3) has been removed. This is an error, as the bridge and the surrounding banks clearly are part of the terrain. It is uncertain what has gone wrong, but it has happened on several other embankments in the case area, see Figure 5.3. Common for the embankments and the walkway bridge is the “man-made shape”. If the terrain filtering is too aggressive, the embankments may have been misinterpreted as buildings and thus removed.



Figure 5.3: Example of embankment which has been removed

The generation of the GeoFyn DTM is clearly not very successful as terrain that could resemble buildings are incorrectly removed and some of the vegetation is still present. For this reason the GeoFyn DTM will not be used onwards in the project.

The issues regarding vegetation is in accordance with Höhle's investigation. Even though the present comparison does not quantify the error, it clearly illustrates that removing vegetation is difficult. In the attempt of removing it, it also results in an estimated and smoothed terrain. Not all errors can be blamed on the fact that data is based on photogrammetry. The DTM generation in Pix4D is fully automated, so it is likely that another algorithm, maybe with the user being able to adjust more variables, would perform better.

5.3 GeoFyn DSM – DK-DEM/Surface

This comparison between the GeoFyn DSM and the DK-DEM/Surface will give an understanding of the changes that has happened since the airborne LiDAR scanning for the DK-DEM products was performed in 2015.

The difference between the two models are shown in Figure 5.4. The segment for this example is chosen because it shows where some of the most substantial changes has occurred since the airborne LiDAR scanning was performed.

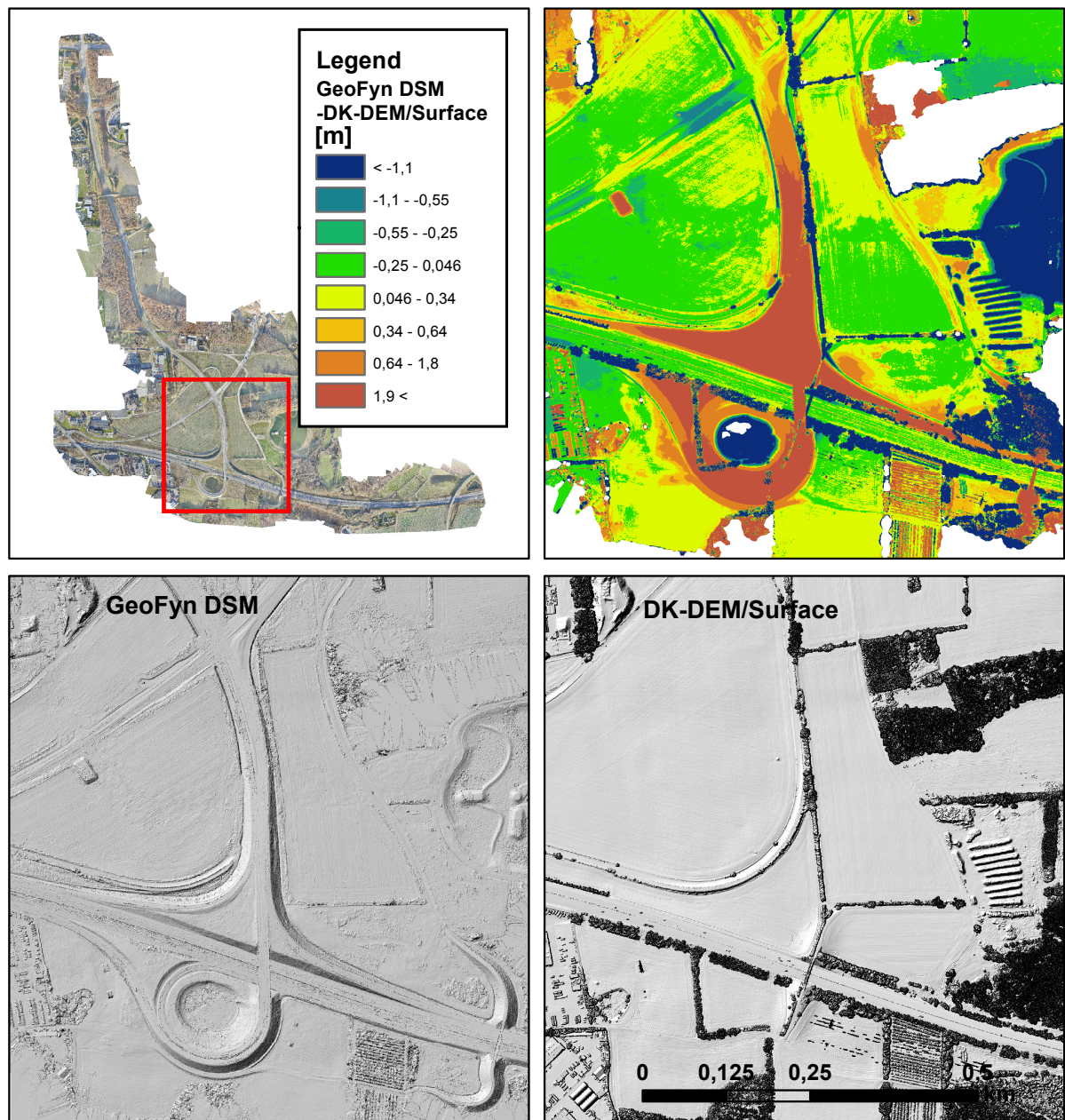


Figure 5.4: Difference between the GeoFyn DSM and the DK-DEM/Surface. The red box on the upper left figure shows which area that is depicted on the other figures. The upper right figure shows the two models subtracted from each other. The bottom left and right figures show a hillshade model of the GeoFyn DSM and the DK-DEM/Surface respectively

As it can be seen in Figure 5.4 the new highway exit and walkway bridge are some of the most drastic changes that has occurred. Additionally many trees and vegetated areas has been cut down and in the right side of the area, just north of the new walkway bridge, a new pond has been built. Subtracting an old DSM from a new DSM is a quick way to get an overview of changes. Whether the changes are of the actual terrain is uncertain, so this method can not alone be used as a way of detecting terrain changes. However subtracting a new and old DEM may possibly play a role during change detection, as areas of interest can be found.

5.4 Summary

Comparisons of the primary data sources have been carried out in the previous sections and some problems regarding the data sources, especially GeoFyn data, were found. For example it can be seen that compared to the DK-DEMs the road is uneven in the GeoFyn products, see road on hillshade model in Figure 5.1 and 5.2. When the UAV data is generated the quality of point matching is depended on structure on the given surface. On an asphalt road there is a lack of structure and colour deviation, which causes a bad point matching, hence an uneven surface.

Based on the comparisons between the GeoFyn DSM and DTM it was made clear that the DTM from GeoFyn does not live up to the standard of the DK-DEM/Terrain, which is why it has been deselected for further use.

Issue regarding unevenness of the surface have only been visually identified and more issues may be present in the UAV data. Because of this it is thought that a quality assessment of the UAV data is needed. The last comparison illustrates the changes that have happened in the area between the collection of the DK-DEM and GeoFyn data. As expected the major difference is along the new highway exit.

In the following chapter the methods for the primary steps in the algorithm will be investigated. Furthermore it was experienced in this section that the UAV data has several issues regarding quality that have to be addressed. Because of this the following chapter will begin with investigating quality control methods, which identify and quantify issues in the UAV data. Afterwards methods for the primary steps, terrain detection and change detection, will be investigated and discussed.

Methods for the updating process 6

Ideas on how to solve the problem with the updating process has been discussed in Chapter 1, *Introduction* and the primary data sources has been presented and compared. This leads to which specific methods that can be used when updating the elevation model. This chapter will look into a selection of methods for the updating process.

As specified previously the project group will focus on the first steps of the updating process, which concern change detection and detection of terrain. In the introduction a workflow for the proposed algorithm was presented in Figure 1.4 and the present chapter will focus on step 1 and 2. The two steps of the workflow has been unfolded, which is shown in Figure 6.1. The relationship between 'Change detection' and 'Change of terrain?' has been changed in the figure, as the arrow points in both directions. This is added to illustrate that the order of the steps is still unknown and it might not necessary be one before the other, but an interaction between the two steps.

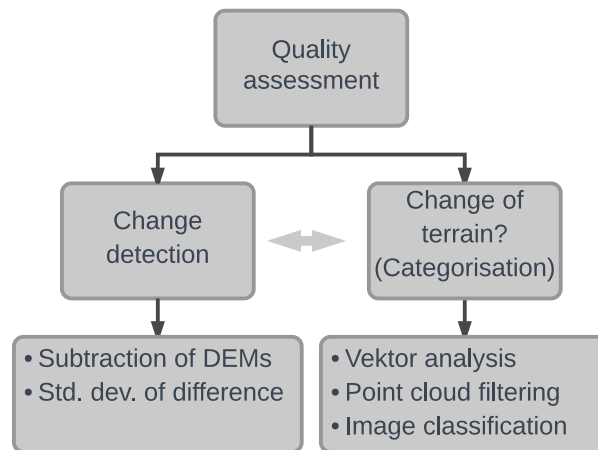


Figure 6.1: Step 1 and Step 2 from the workflow presented in Chapter 1, *Introduction* is unfolded

Before the two steps can be carried out the quality of the UAV dataset will be investigated. In this chapter methods for a quality assessment will also be introduced first. The word quality is in this context understood as an indication of the data's influence from noise, hence lower noise means higher quality. A quality assessment of the UAV data is essential as inaccuracies already have been identified. Two methods are described with focus on how the assessment is carried out and the used statistical measures.

Secondly methods regarding categorisation is studied, which focus on the step 'Change of terrain?'. It is decided that a categorisation is needed to categorise terrain and non-terrain, but several approaches exist for this purpose. The section will focus on three approaches; vector analysis, point cloud filtering and image classification. The approaches will be described and the applicability will be assessed.

Thirdly methods for change detection will be described. The change detection comprise a subtraction of two DEMs and an evaluation of the obtained difference. It will focus on which parameters that should be considered when evaluating if a detected change is an actual change or caused by inaccuracies.

6.1 Quality assessment

The purpose of the quality assessment is to quantify the quality of the UAV data. Quality can be assessed on many parameters, but the following will focus on if the data is effected by noise and thereby how well it represents the surface.

The quality of the UAV data from GeoFyn is unknown, as no quality check has been carried out by GeoFyn or others. By visual inspection in Chapter 5, *DEM comparisons* it was found that there are several inaccuracies in the dataset, but it has not yet been quantified.

A quality assessment will be carried out, because the precision of the data influences the detection of changes. If both datasets are accurate it is possible to detect small changes and vice versa if the data is very inaccurate.

There are different ways to carry out a quality assessment of a dataset and the following will present two ways. Firstly a control of the accuracy based on a collection of surveyed check points and secondly by statistical means, where the precision is assessed.

6.1.1 Check points

It is possible to check the accuracy of the UAV data by a set of check points. The check points will be surveyed by terrestrial means and the accuracy of the check points has to be superior, which can be obtained by a GNSS receiver or total station. As the quality of the UAV data most likely vary across the area depending on the characteristics, the check points should be divided into sub-areas, but also be evaluated generally for the whole area. [Höhle, 2009]

When a sufficiently large dataset of check point has been collected, the UAV data can be evaluated by subtracting the elevation of the check point and the elevation of the UAV data. To be able to compare the elevations at the position of the check point it is necessary to either interpolate between the points or create a DEM based on the UAV data. The elevations are compared with the following equation.

$$\Delta H = H_{DEM} - H_{check\ point} \quad (6.1)$$

Where

ΔH : Difference in elevation

H_{DEM} : Elevation of DEM

$H_{check\ point}$: Elevation of check point

To evaluate the differences of the area a number of accuracy measures can be calculated. To start these calculations imply that the errors follow a normal distribution. A RMSE value and standard deviation is calculated with: [Höhle, 2009, p. 17]

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n \Delta H_i^2} \quad (6.2)$$

$$\hat{\sigma} = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (\Delta H_i - \hat{\mu})^2} \quad (6.3)$$

Where

$$\hat{\mu} : \text{Mean error} \quad \hat{\mu} = \frac{1}{n} \sum_{i=1}^n \Delta H_i$$

If the dataset is heavily affected by outliers, the errors may not follow a normal distribution and need to be evaluated with robust accuracy measures, as discussed in Section 1.5, *Issues regarding photogrammetric based DTMs*. The mentioned method of assessing the accuracy is reliable, but the dataset can not be assessed for every meter with this method, as it would require a very large set of check points. A disadvantage of the method is also that the check points have to be collected, which is not a part of a regular UAV data collection. For this reason the following method is introduced, as no additional data has to be collected to assess the quality of the UAV data.

6.1.2 Adjustment of best fit plane

The second method uses the UAV data itself for assessing the quality, resulting in a relative measure of quality. By dividing the dataset into a grid, data inside each cell can be evaluated by statistical measures.

A simple approach is to calculate the standard deviation based on the mean value of the elevations inside the cell.

$$\hat{\sigma} = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^n (H_i - \hat{\mu})^2} \quad (6.4)$$

As this approach uses the mean value it will result in large standard deviations if the terrain inside the cell is sloped, and then incorrectly indicate that the precision of the data is low. Alternatively a best fit plane of the points can be calculated based on least squares adjustment, which is illustrated in Figure 6.2. In this case the plane will take the sloped terrain into consideration.

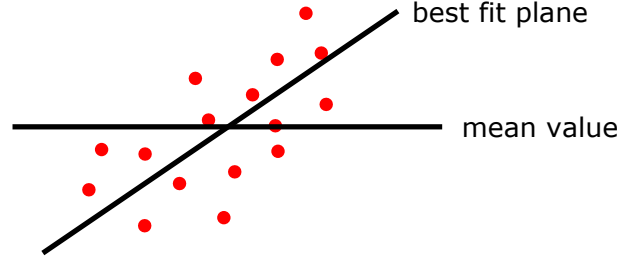


Figure 6.2: Points on a slope where the mean value and best fit plane is illustrated

The plane is defined by the following equation and a minimum of three points is needed to define the plane.

$$H = aE + bN + c \quad (6.5)$$

The plane is calculated based on least squares adjustment and if the system of equation is overdetermined, residuals can be calculated. Based on the residuals the a posteriori standard deviation of unit weight can be derived with the following equation. [Cederholm, 2000, p. 39]

$$\hat{\sigma}_0 = \sqrt{\frac{\hat{r}^T C \hat{r}}{m - n}} \quad (6.6)$$

The precision inside a cell can be assessed based on the standard deviation of unit weight. It is important to note that the assumption for least square adjustment is that the observations are independent. [Cederholm, 2000, p. 5] This might not be the case for a photogrammetric based dataset, why the statistical measures might be biased.

If it is expected that the dataset is affected by outliers, it is advisable to perform a robust adjustment. There are several robust methods, for example minimising the residuals according to the L_p -norm. To do this Iteratively Reweighted Least Squares (IRLS) can be used, which adjusts the weights through iterations to minimise the residuals according to the L_p -norm.

To evaluate the precision of the data inside a cell where the plane is calculated with a robust adjustment eq. (6.6) can not be used as it is not robust. Another equation for the standard deviation of unit weight is used, where the median of the residuals is utilised, as it is robust. [Cederholm, 2016b]

$$s = \frac{\text{median}(|\hat{r} - \text{median}(\hat{r})|)}{0.6745} \quad (6.7)$$

When a plane is calculated the robust standard deviation of unit weight can be used as a measure to remove outliers.

Another method, that has a more practical approach but still is robust, is based on an unweighted least squares adjustment. The method does not adjust the weight of the observations, but deselect a number of observations through iterations, based on the residual being outside an interval. This interval is based on the standard deviation of unit weight and observations with residuals that exceed for example $\pm 2.5 \cdot \hat{\sigma}_0^2$ can be removed. By performing

a new adjustment and calculating the standard deviation of unit weight, the change of the standard deviation can indicate if outliers have been removed or not.

The approach with a robust method using adjustment of a plane will be used to assess the quality of the UAV data, as it is expected that the dataset is affected by outliers. The exact method will be chosen and described in Section 9.2, *Robust adjustment of plane*.

6.2 Categorisation

A categorisation of different features in the case area will make it possible to remove parts of the UAV data which are not seen as terrain. As mentioned in the introduction the features which definitely are not seen as terrain are cars, buildings, high vegetation and other man-made objects and will be removed when categorised. Some features, such as grass and low vegetation, are only close to terrain and may or may not be removed as it depends on how much the terrain has changed. This underlines the need for an interaction between the change detection and the categorisation step. It is uncertain in which order the steps will be performed, as some experimentation has to be done in order to find the best approach.

Before figuring out the order of the steps an understanding of how to categorise features in UAV data is needed. Many different methods exist and it is uncertain which will be best to utilise. For this reason different methods will be reviewed and described, which will lead to a choice regarding which methods that can be used to categorise the features in the case area.

Three different methods of terrain categorisation will be described, as these are seen as being the most relevant methods to use. For some of the methods many approaches exist, in this case only a few will be described. The described methods are all seen as a possible way to categorise UAV data, but the final choice of method will primarily be subjective as time prohibits the possibility to test and compare the methods.

6.2.1 Vector analysis

A simple approach for categorising the UAV data is to use the public available GeoDanmark vector data. The idea is to determine which points falls within a feature polygon and thereby classify the points according to the polygon. The features in the GeoDanmark vector data which are thought to be used can be seen in Table 6.1.

Table 6.1: GeoDanmark data which can be used to categorise UAV data [GeoDanmark, 2014a]

Feature	Geometry
Building	Polygon
Road centre	Line
Forest	Polygon
Lake	Polygon

A line feature, such as the road centre, requires that a buffer polygon is created around the line, which may require manual inputs as the size of the buffer have to be varied depending on the road. It should be considered how wide the buffer polygon is created as points on road equipment, such as guardrails and light poles, may fall within it. The points which falls within the road buffer polygon can be seen as terrain points.

For buildings and forest polygons a small buffer extending the polygon may be necessary to generate in order to make sure points get correctly classified. The points which falls within the polygons of the building and forest are certain to be non-terrain points. It should be considered if GeoDanmark data is up to date, as some buildings may have been removed or added. Removed buildings is not a big issue as the DK-DEM/Terrain have approximated the terrain under the building. Even though a house has been demolished the terrain may not have changed considerably if nothing else has been built. If a new building has been built, the points in the UAV data can get incorrectly categorised if the building polygon has not been added to the GeoDanmark vector data.

The closed lake polygon is included as photogrammetric used techniques to generate points on lakes or other water surfaces can not be successfully performed. This is because a water surface either is in motion or still, which creates a homogeneous surface, making point matching very difficult. If the lake is very shallow and clear and the bottom can be seen in the images, points can potentially be generated on the bottom of the lake. This is also not wanted as the water surface is seen as “terrain”. For this reason the lake polygon, maybe with a small buffer, can be used to classify points on lakes as non-terrain.

The GeoDanmark vector data can quickly be used to make a coarse classification of the UAV data. But some features may be neglected and not classified, for example different types of vegetation, which could be useful information during the change detection step. Additionally the dataset may not be up to date as it only gets updated once a year, which can introduce an error in the classification of terrain points.

6.2.2 Point cloud filtering

Before a point cloud is classified a quality assessment, like the one mentioned in this chapter, can be made in order to remove outliers. This is done as outliers can affect the effectiveness of the filtering methods. Many different methods for filtering a point cloud exist, but in the following only two methods will be described. The first methods is the block-minimum filter, which is a very simple method and used for generating a coarse DTM. [Pfeifer, 2008, p. 315] Following this a surface-based filtering method will be described as this method is used in a DTM filtering program developed at Aalborg University as part of a master thesis. [Matthesen and Schmidt, 2014]

Block-minimum filter

A block-minimum filter operates by dividing a point cloud into blocks of a certain size and afterwards finding the point with the lowest elevation. The size of the block is chosen such that it is certain that for each block at least one ground point is present. To be certain that a ground point is present the block must be of a size which matches the size of the biggest building and vegetated area in the point cloud. If a block size is chosen this way, it is most likely that a ground point will be found, as it is probable that the block will cover some of the ground and the large building or vegetated area. [Matthesen and Schmidt, 2014, p. 64] When the points with the lowest elevations are found a DTM can be generated. The generated DTM is only a coarse representation of the terrain as the filter assumes the terrain is flat. If the terrain is sloped the filter will in most cases find a point which is too low and thus not represent the terrain very precisely. [Pfeifer, 2008, p. 315] For this reason the block-minimum filter is typically used as a starting point for other filtering methods, such as the surface-based filtering method. [Briese et al., 2002]

Surface-based filtering

Surface-based filtering works by using a least squares adjustment which approximates a surface, for example a polynomial, to the point cloud. The filter works by first using a block-minimum filter to find local minima points. In this case the block size do not have to be chosen according to buildings or vegetated areas, as the filter gradually will remove any non-terrain points. The filter will do this by assigning weights to the points found by the block-minimum filter after an initial adjustment of a surface for each block. The weights are distributed such that points above the adjusted surface will have a smaller weight than the points which lies under the surface. The adjustment iterates until a stop criteria is met. This will create a coarse DTM where most non-terrain points are removed. To further refine the DTM a vertical buffer around the coarse DTM is created, the points which fall within the buffer will be used in a similar adjustment as the previous. This process keeps iterating until the refinement is deemed satisfactory and a new smaller buffer can then be used to select the final terrain points. [Briese et al., 2002]

An example on the refinement process can be seen in Figure 6.3. The example is based on how the DTM filtering program developed at Aalborg university works. In a) a block minimum filter is used to find the lowest points in a 5×5 m grid. In b) a polynomial is approximated based on points found in a) within a 100×100 m grid. The red polynomial is the first iteration and the black is the last. Notice that the point on the roof is eliminated by giving it a low weight. In c) points for the refinement is chosen based on a vertical buffer which is formed around the polynomial from b). In d) the chosen points from c) is used to approximate a new green polynomial which a smaller buffer can be formed around and used to make the final selection of terrain points. [Matthesen and Schmidt, 2014, pp. 106-107]

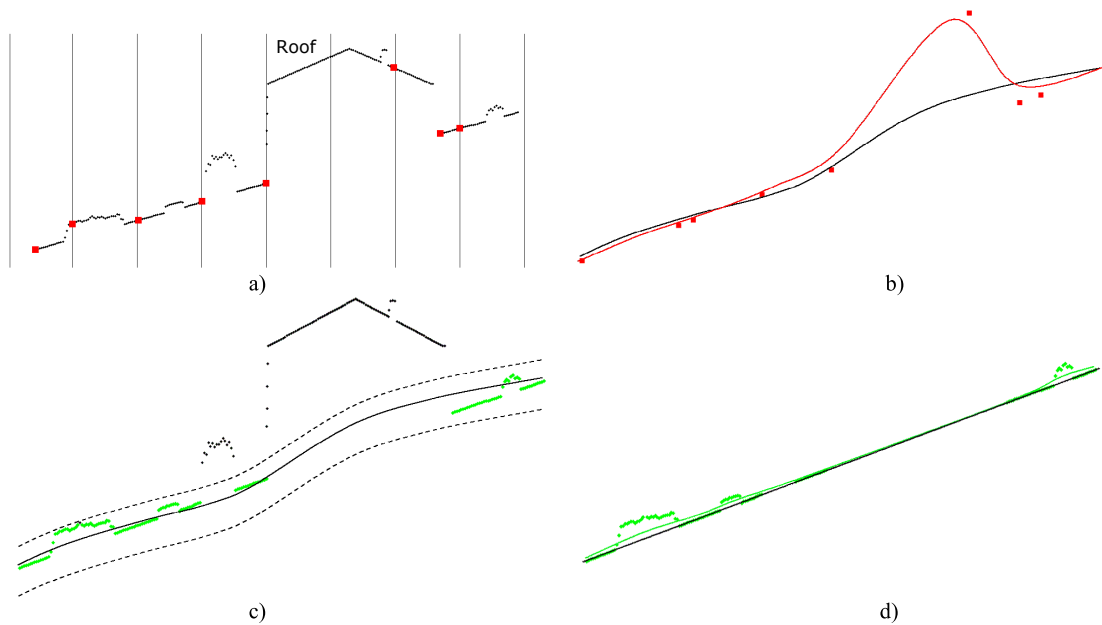


Figure 6.3: Surface-based filtering refinement process. [Briese et al., 2002]

The surface-based filtering categorise points as either terrain or non-terrain. To classify non-terrain points into subcategories such as vegetation, buildings, etc. additional filters can be used. A vegetation filter can for example extract points in a certain elevation range to categorise low, medium and high vegetation. [Terrasolid, 2016, p. 491] A building filter can classify points on roofs of buildings by searching for planar surfaces in the classified non-terrain points. [Terrasolid, 2016, p. 510]

If the surface-based filtering are used on UAV data, the same issues as described in Section 1.5, *Issues regarding photogrammetric based DTMs* persists. This means that for areas with almost no terrain points, for example if large buildings and forests are present, the filter can potentially classify non-terrain points as terrain. [Matthesen and Schmidt, 2014, pp. 140-141] For this reason the filter can not alone be used to classify terrain and non-terrain points, which makes the use of the filter for this project questionable.

6.2.3 Image classification

The data source for image classification is digital raster images, for example orthorectified images captured from an UAV, aeroplane or satellite. When an image is captured it can contain several bands, which represents frequencies across the electromagnetic spectrum. The orthophoto from GeoFyn only have three bands; the red, green and blue bands. These three bands cover the electromagnetic spectrum between 380 and 740 nm. The GeoDanmark orthophoto has a fourth band which covers the spectrum between 750 to 1400 nm, this is called the near-infrared (NIR) spectrum. For both the GeoFyn and GeoDanmark orthophotos the bands are digitally stored as 8 bit gray values, which means that each pixel in each band can have a discrete value between 0 and 255, representing the brightness of each bands color spectrum. [Geodatastyrelsen, 2015, p. 1]

To make an image look how a human perceives the world the red, green and blue bands can be merged, this is called a “true color composite”. The NIR band is typically used to create a “false color composite” where the NIR, red, and green bands are used. The false color composite highlights different types of vegetation and its health, which makes it useful for monitoring biodiversity and ecosystems etc. As the orthophoto from GeoFyn do not have the NIR band it is expected that it will be difficult for the image classification methods to distinguishing between some types of vegetation and objects. A possible option is to use the yearly updated GeoDanmark orthophoto for the classification, but as with the other GeoDanmark data, it needs to represent the current state of the area else the classification may end up being inaccurate. [Potůčová, 2016] An example of the merging of the red, green and blue bands to form a true color composite is seen in Figure 6.4

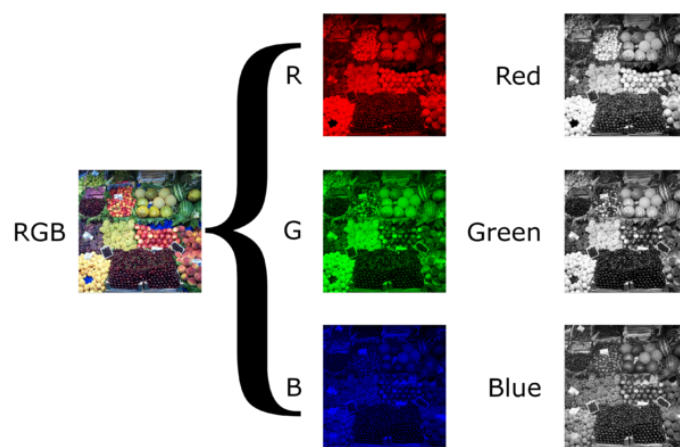


Figure 6.4: True color composite from red, green and blue gray scale image bands [Dilmen, 2012]

Based on digital images a classification representing different features in the orthophotos can be generated by utilising pixel grouping techniques. After the image has been classified a thematic map is generated, which is comprised of pixels with a value representing a feature classes. To classify the UAV data an approach similar to the one mentioned in Section 6.2.1, *Vector analysis* can be used. The feature pixels can be converted to polygons and based on this a selection of the points in the UAV data can be performed. An example of this is shown in Figure 6.5, where the blue points are categorised according to the blue polygon which they fall within.

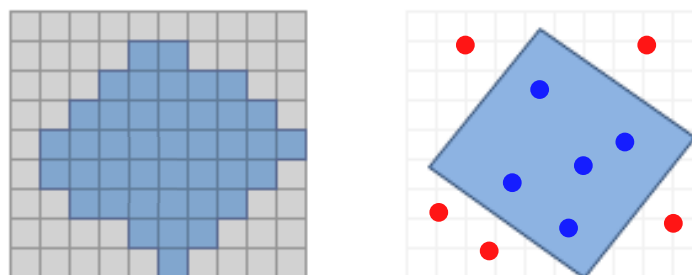


Figure 6.5: Example of conversion from raster to polygon and subsequent categorisation of UAV data. The figure is based on [Esri, 2017]

The three main approaches for making the image classification are: [Potůčková, 2016]

- Unsupervised image classification
- Supervised image classification
- Object-based image analysis

Only unsupervised and supervised image classification will be described. For each of these approaches different methods exist but only one method for each approach will be described. Common for the unsupervised image classification methods is that the user picks the number of feature classes for the image. For some unsupervised image classification methods additional variables can be picked which will enhance the classification process. [Potůčková, 2016] The method for unsupervised image classification which will be described is the Iterative Self-Organizing Data Analysis Technique (ISODATA).

Common for the supervised image classification methods are that the user creates samples of pixels or “training sites” in the images of the feature classes which are to be categorised. These samples are then used to classify the image. [Potůčková, 2016] The method for supervised image classification which will be described is the Maximum likelihood classifier technique.

ISODATA - unsupervised

The first step of the ISODATA methods is to pick “ j ” random pixels in the multispectral space. j is the maximum possible number of feature classes in the image. The multispectral space consists for each pixel of a x and y position in the image and three color values for red, green and blue and additional bands, such as the NIR. This means that a pixel in the ISODATA algorithm is seen as at least a 5D point and not a 2D point in a grid.

The randomly picked pixels are seen as the first centroid of the multispectral pixel clusters. The second step is to calculate the distance to each centroid from each pixel. The pixels which are closest to a centroid will be seen as part of a new pixel cluster. The third step is to calculate a new cluster centroid, this is found by taking the mean of the pixels in the cluster. The second and third step iterates until the change of the cluster centroids are less than a given threshold or the number of iterations exceeds a given number. [Potůčková, 2016]

Additional to the variable j several parameters are set, these are: [Potůčková, 2016]

- Minimum number of pixels in a cluster
 - If the number of pixels in the cluster is too low it will be deleted and the pixels will go to other clusters
- Maximal standard deviation within a cluster
 - If the standard deviation in a cluster is too big the cluster will be split into two new clusters.
- Minimum distance between cluster centres
 - If cluster centres are too close to each other the clusters will be merged

As a consequence of the additional parameters fewer clusters than the chosen j value is possible. This is because clusters can be merged and deleted. If more than j clusters are present, the closest clusters will be merged until only j clusters exist. [Potůčková, 2016] As it is impossible to illustrate the ISODATA method with 5D pixel clusters, an example with 2D points can be seen in Figure 6.6.

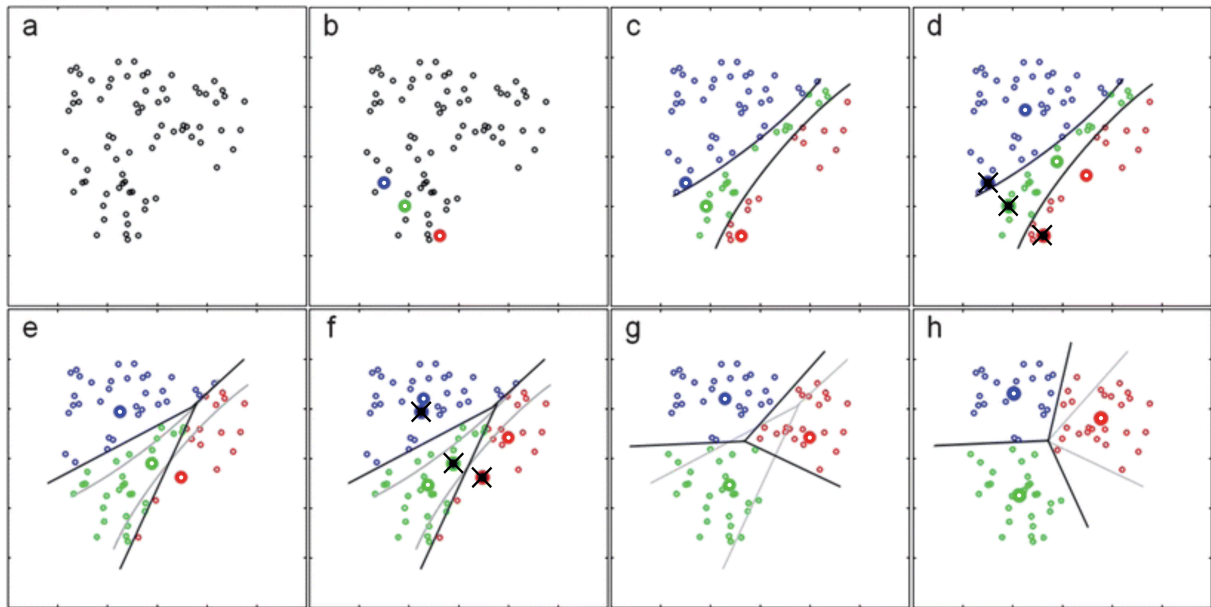


Figure 6.6: Example of clustering 2D points with the ISODATA method [Schuermann, 2017]

In Figure 6.6 a) is the initial data points, an initial guess on the number of clusters is three, so $j = 3$. In b) three random (red, green and blue) points are chosen as the first centroids. The colors do not represent any value, it is only for the sake of visualisation. In c) the closest points are found for each centroid and the first clusters are formed. In d) the new cluster centroids are found by calculating the mean of the points in the clusters, afterwards the old ones are cancelled. In e)-g) new centroids and clusters are formed and will keep iterating until the centroids positions converge. In h) the centroids positions have converged and the final clusters are found.

By using ISODATA a quick classification can be made and no prior knowledge of the area is needed. The downside of unsupervised image classification is that it works best with images that have clusters with a similar variance. If this is not the case, the classification may have a “salt and pepper” effect. This means that clusters which should be one, are split into several smaller clusters. [Nath et al., 2014, p. 556] Because of this, using ISODATA may not be suitable, as the case area has many fields and forested areas which can have a large pixel variability. [Wu, 2017]

Maximum likelihood classifier - supervised

The first step of the maximum likelihood classifier is, as with any supervised classifications methods, to pick training sites. The training sites must represent the classes which are seen in

the area and several training sites can be picked for one class. It is expected that the pixels in the image follow a normal distribution. The training sites are chosen by creating a closed polygon and the pixels which fall within the polygon are the samples of the training site. For each pixel only the color values of its band are used in the algorithm. A vector, $\mathbf{x}_{ts,i}$, with values of a i 'th training site pixel look like:

$$\mathbf{x}_{ts,i} = [BV_1, \dots, BV_k]^T$$

Where

BV_{1-k} : Brightness value (0-255) of band 1 to k for the training site pixel

When the training sites have been picked, the mean vector and covariance matrix of the pixels in the class has to be calculated. The mean vector has the following content:

$$\mu_{ts} = [\mu_{BV_1}, \dots, \mu_{BV_k}]^T$$

Where

$\mu_{BV_{1-k}}$: Mean brightness value of band 1 to k for all pixels in the training site

When the mean of the training site class have been found the variance and covariance of the class can be calculated and ordered into a covariance matrix. The covariance matrix has the following content:

$$\Sigma_{ts} = \begin{bmatrix} \sigma_{BV_1}^2 & cov(BV_1, BV_2) & cov(BV_1, BV_3) & \dots & cov(BV_1, BV_k) \\ cov(BV_2, BV_1) & \sigma_{BV_2}^2 & cov(BV_2, BV_3) & \dots & cov(BV_2, BV_k) \\ cov(BV_3, BV_1) & cov(BV_3, BV_2) & \sigma_{BV_3}^2 & \dots & cov(BV_3, BV_k) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ cov(BV_k, BV_1) & cov(BV_k, BV_2) & cov(BV_k, BV_3) & \dots & \sigma_{BV_k}^2 \end{bmatrix}$$

Where

$\sigma_{BV_{1-k}}^2$: Variance of the brightness values of the bands from 1 to k in the training site

$cov(BV_1, BV_2)$: Covariance of the brightness values of the bands from 1 to k in the training site

When the statistics of the class have been calculated the probability of a single pixel belonging to a class can be calculated by the following discriminant function: [Richards and Jia, 2006, p. 196]

$$g_i(\mathbf{x}) = \ln |\Sigma_{ts}| - (\mathbf{x} - \mu_{ts})^T \Sigma_{ts}^{-1} (\mathbf{x} - \mu_{ts})$$

Where

$g_i(\mathbf{x})$: Probability that the evaluated pixel belongs to the i 'th class

$|\Sigma_{ts}|$: determinant of the covariance matrix of the class

\mathbf{x} : Vector with brightness values of an evaluated pixel

The probability is calculated for each class and the pixel is assigned to the class which is the most probable. If the pixel has a low probability of belonging to all classes it is not classified. An example of a pixel evaluated with the maximum likelihood method for two classes that has two bands can be seen in Figure 6.7

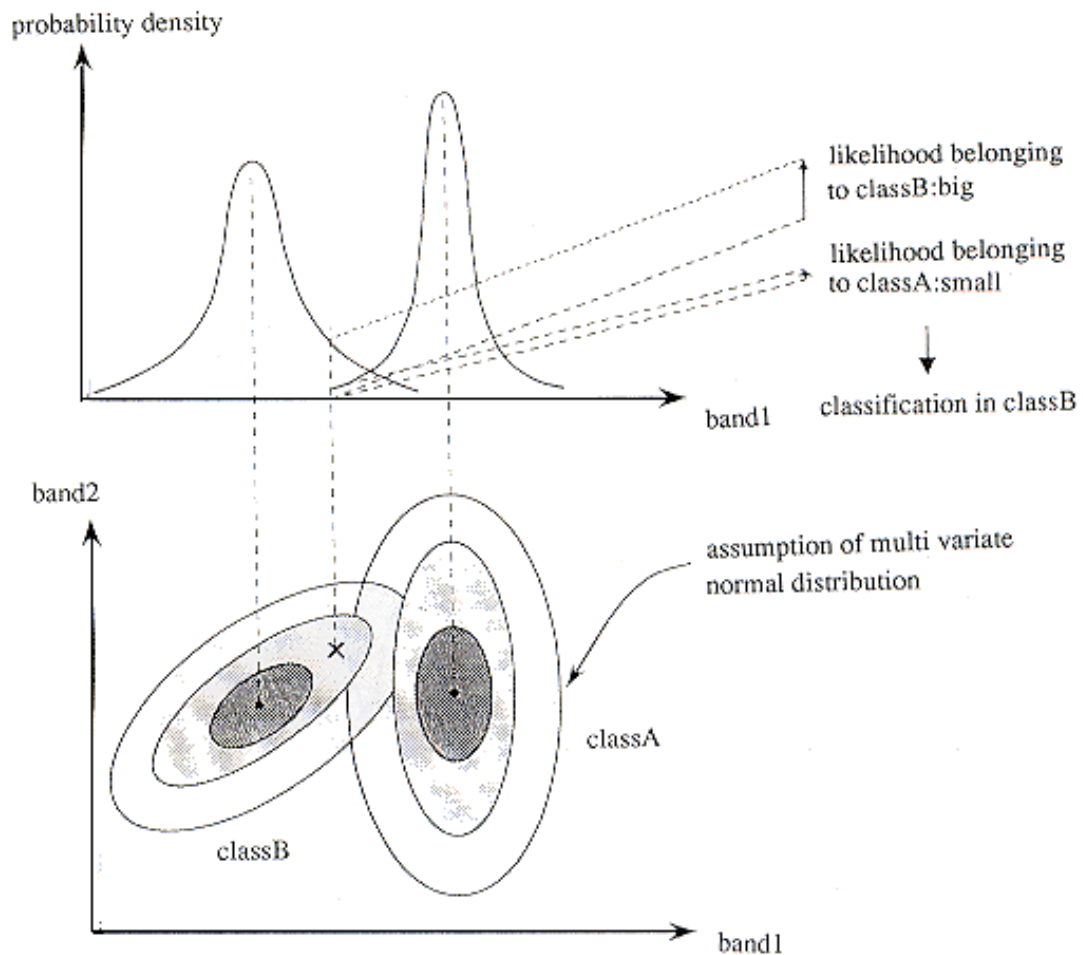


Figure 6.7: Evaluation of a pixel according to the multivariate distribution of two classes with two bands [JARS, 1996]

In Figure 6.7 two classes, A and B, with two bands and their 2D multivariate probability density functions are seen. The cross in the bottom figure is the pixel which is evaluated if it belongs to class A or B. As seen on the figure the pixel is more probable to be in class B.

The advantage of using a supervised classification method is that the user can train the classifier to detect the classes, which can give good results if done correctly. The drawback is that the process of making training sites can be quite time consuming and the user can possibly neglect certain classes in the image, which can introduce errors to the classification. [Nath et al., 2014, p. 556] It is thought that the maximum likelihood classifier will perform better than the ISODATA method when the GeoFyn orthophoto is used, but some experimentation have to be done in order to determine this.

6.3 Change detection

The primary aim of this project is to detect changes in terrain and this section will focus on the methods and consideration of change detection between two DEMs. When a change has to be detected based on two DEMs it is unavoidable to subtract the two models from each other. The question is which parameters should effect the decision if an actual change has happened or if it is caused by something else. The subtraction results in a direct difference between two elevations and a measure is needed to evaluate this difference. This measure will often be a result of the standard deviation of the two models, leading to the standard deviation of the difference.

When the standard deviation of the difference is calculated it is important to consider if the standard deviation should be uniform for the whole area, be uniform for sub-areas or differ for every cell, as illustrated in Figure 6.8. In a PhD thesis from 2008 concerning change detection of rivers the same consideration was made. It was stated that the most common take on the issue is to calculate a global standard deviation, but the option of making a standard deviation of the difference on a cell-by-cell basis is mentioned. It is noted that this is an elegant solution, but the thesis does not propose a solution for the task. The thesis uses an example of treating the standard deviation of the difference for sub-areas, but it rises the concern of what factors play a role when deciding the extent of the sub-areas. [Wheaton, 2008, pp. 91-93]

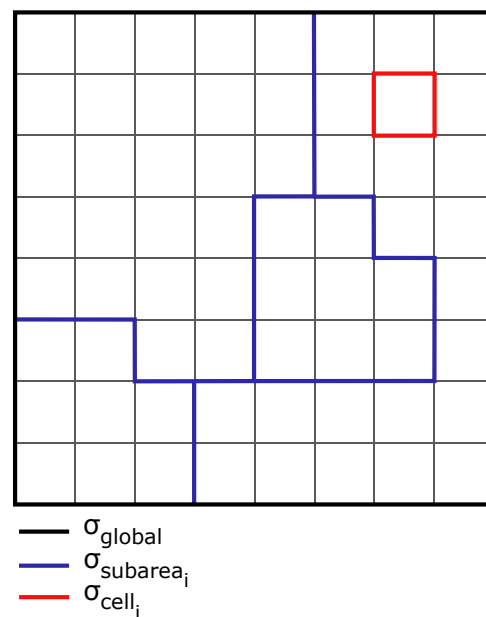


Figure 6.8: Standard deviation of the difference specified globally, for subareas or on a cell basis

The following will look into the concerns regarding subtraction of two DEMs. The different options regarding representation of the models will be discussed along with how the two models need to match to be able to compare them. Afterwards the different options of finding the standard deviation of the difference will be listed and one will be chosen.

6.3.1 Subtraction of DEMs

When the difference between two DEMs need to be calculated the two models are subtracted. This require that the models are continuous, so an elevation can be found for every position. A continues model can both be a raster model and a TIN-model, but the following is based on raster models as the DK-DEM/Surface and the DK-DEM/Terrain already are raster models.

When creating a raster model the cell size is important, but also the relationship between the cell size of the two DEMs is of significance. If the cell size is used for adjustment of a plane, as introduced in Section 6.1, *Quality assessment*, it is necessary that the cells on average contain a reasonable amount of points for an adjustment.

If the cell size of the two models is not the same, it is necessary that the larger cell size is dividable by the smaller cell size, as Figure 6.9 illustrates. If this is not the case, the subtraction of the centre points is not possible. If the cell size of the two models are different, it is still desired to calculate a difference for every cell in the smaller grid. As the subtraction is made between centre points of the cells, an equal number of centre points is needed in the two models. As Figure 6.9 shows, this can be done by letting the cells of the larger grid overlap. In the figure the smaller grid is gray. The first cell of the larger grid is marked with red along with the corresponding centre point. The second cell, which is blue, is only shifted a third of the cell size (corresponding to the smaller cell size), hence overlapping the red cell, creating a centre point in the next smaller cell. This pattern continues resulting in an equal number of cells and centre points in the two models. When performing this a gab is created around the edge, which can be taken care of by extending the model with the larger grid by a third pixel size on all edges.

A last condition to be able to carry out the subtraction is that the corner of the cells line up, which has to be considered when dividing the data into a grid.

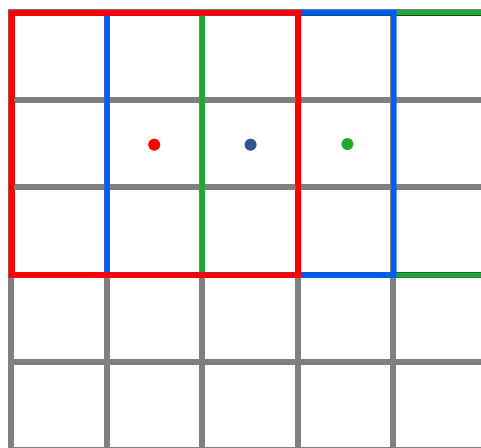


Figure 6.9: Example of two grids with different cell size. One large cell contain nine small cells. The red, blue and green cell shows that the larger cells overlap to be able to create a centre point for every smaller cell

The UAV data is a point cloud, which is not a continuous dataset, so data processing is necessary. Through the adjustment of a plane inside a given cell, which was introduced in Section 6.1, *Quality assessment*, the elevation of the centre point of the cell can be found. By converting the data to a raster with the elevation as the cell value, a continuous dataset can be created, which means that an elevation can be found for every position within the model. This method reduce the level of information, but it is impossible to keep all information when a raster is made.

The DK-DEM consist of both a point cloud and raster models (DK-DEM/Terrain and DK-DEM/Surface) and considerations on which product that should be used in the subtraction has not yet been discussed.

If DK-DEM/PointCloud is used it is possible to treat it like the UAV data as described above, though it need to be considered that the point density is lower, which effects the choice of cell size. The advantage of this is the statistical output of the adjustment, which can be used to evaluate the precision of each cell, rather than using a general measure for the whole dataset. If one of the raster models are used, no processing of the dataset is needed prior to the subtraction, as long as the cell size of 0.4×0.4 m is maintained. This is effortless regarding data processing, but the disadvantage is that the precision of the dataset only is given as an overall value for well-defined surfaces in the specification. To a start the raster models from the DK-DEM will be used for the change detection. Later on it will be discussed if the DK-DEM/Terrain or the DK-DEM/Surface should be used for the subtraction or maybe both.

6.3.2 Standard deviation of the difference

When the subtraction of the two models is carried out the next concern is to evaluate the difference, to be able to decide if the change is caused by inaccuracies in the models or if it is an actual change. In accordance to the *special law of propagation of variances*, the standard deviation of the difference is found by the partial derivatives of the function and standard deviation of the two products as shown in eq. (6.9). The partial derivatives of eq. (6.8) is 1 for both terms, which leads to eq. (6.10). [pp. 81-86][Wolf and Ghilani, 1997]

$$H_{diff} = H_{DK-DEM} - H_{UAV} \quad (6.8)$$

$$\sigma_{diff} = \sqrt{\left(\frac{\partial H_{diff}}{\partial H_{DK-DEM}}\right)^2 \sigma_{DK-DEM}^2 + \left(\frac{\partial H_{diff}}{\partial H_{UAV}}\right)^2 \sigma_{UAV}^2} \quad (6.9)$$

$$\Downarrow$$

$$\sigma_{diff} = \sqrt{\sigma_{DK-DEM}^2 + \sigma_{UAV}^2} \quad (6.10)$$

Where

σ_{diff} : Standard deviation of difference

σ_{DK-DEM} : Standard deviation DK-DEM

σ_{UAV} : Standard deviation UAV data

In the present project several solutions regarding the standard deviation of the difference is possible. When the DK-DEM raster model is used as one of the models the standard deviation will be an overall value based on the specification. So even though the precision of the data is not uniform, which it most likely is not, it will be treated so, as a cell-by-cell standard deviation does not exist. As mentioned before the option of using the DK-DEM/PointCloud exist, which could allow for the heterogeneous precision across the area.

Based on the adjustment of a plane for the UAV data a standard deviation is found for each cell, but a global standard deviation could also be found by other means. As the UAV data is generated by photogrammetric methods it is expected that the precision vary between different areas such as roads, fields, vegetation and buildings. Therefore it seems beneficial to use a cell-by-cell standard deviation, which can be calculated based on the plane.

Based on this the standard deviation of the difference between the DK-DEM raster model and UAV data will be a product of a global standard deviation and a cell-by-cell standard deviation, resulting in a cell-by-cell σ_{diff} . The difference between the two models will be calculated on a cell-by-cell basis and the evaluation will be individual for every cell. If the UAV data inside a cell has a large standard deviation, it causes that the difference between the two models has to be larger to categorise as a *reel* change, than inside a cell with a small standard deviation.

When the standard deviation of the difference is calculated, the value can be used to determine if a detected change is caused by inaccuracies or a reel change. The obvious approach is to neglect changes between $\pm 3 \cdot \sigma_{diff}$ as they probably are caused by inaccuracies.

Problem statement 7

Through the previous chapters several subjects have been investigated with the purpose of answering the initial problem statement;

Which data and methods can be used to update the Danish Elevation Model?

The following will answer the initial problem statement by summarising the problem analysis and on the basis on this, a decision will be made of which data and methods that will be used onwards.

7.1 Summary

In Chapter 3, *The Danish Elevation Model* and Chapter 4, *GeoFyn data* an investigation of the primary elevation data sources were performed. This was done in order to get insight in how the data was generated and what can be expected of it. Based on a quality control of the DK-DEM/PointCloud, SDFE found that the vertical accuracy is within the specified 0.05 m. Based on the DK-DEM/PointCloud the DK-DEM/Terrain and DK-DEM/Surface are generated which are raster models with a cell size of 0.4×0.4 m. The GeoFyn data consists of an orthophoto, a photogrammetric generated point cloud, a DSM and a DTM. The accuracy of the point cloud is unknown as no check points were collected.

To get further familiar with both datasets they were compared and discussed in Chapter 5, *DEM comparisons*. The comparison between the DK-DEM/Surface and the DK-DEM/Terrain showed a good example of the difference between a DTM and a DSM and is seen as a best case scenario of a DTM. Following this a comparison between the GeoFyn DSM and GeoFyn DTM was made. This comparison showed that the GeoFyn DTM has several issues regarding vegetation and man-made shaped embankments and for this reason it will not be used onwards. The comparison between the GeoFyn DSM and DK-DEM/Surface showed which areas of the case area that have undergone changes, primarily the areas around the new highway exit and walkway bridge are changed. From the hillshade models of the GeoFyn DSM and DTM it was seen that the road is very uneven, which is thought to be caused by a bad point matching as the road has a lack of structure.

After the investigation of the primary data sources, methods for updating the Danish Elevation Model is described in Chapter 6, *Methods for the updating process*. To determine the quality of the UAV point cloud a quality assessment can be done by using check points, but as no check points are available the quality of the UAV data can not be assessed on the basis of this method. The relative precision can be checked by a method where the UAV data will be divided into cells and a best fit plane will be calculated through an adjustment for every cell. Based on the adjustment statistical measures can be calculated, which can be used to assess the quality of the UAV data and remove outliers. As the data is thought to contain outliers a robust adjustment should be used, but the type is yet to be decided.

Based on the investigation of the categorisation methods it was found that several approaches exists for each method. Based on the GeoDanmark vector data a coarse categorisation of the area can be made fairly simple. However it is important to consider when the vector data was updated last, as the data will not reflect the changes happened after this date. Another option for categorisation is to use point cloud filtering, where the aim is to separate terrain from non-terrain points. The issues regarding areas which mostly contain vegetation and buildings remain with this solution, and therefore the method can not necessary stand alone in a categorisation. For image classification it is found that the maximum likelihood classification method should give the best result, as it is supervised. This is not documented, as the classification has not been carried out. When performing the classification it is not decided whether the GeoFyn or GeoDanmark orthophoto should be used, as an investigation is needed in order to determine this.

In the investigation of the change detection methods it was found that it can be approached in a few different ways. Common for the approaches is to subtract the elevations of the DK-DEM from the adjusted planes of the GeoFyn data and use the standard deviation of the two products in order to calculate a standard deviation of the difference. The standard deviation of the difference can be used to determine if a change has occurred. The standard deviation of the GeoFyn product is obtained thorough the adjustment of the planes and results in a cell-by-cell standard deviation. From the specification of the DK-DEM a 0.05 m standard deviation can be expected, which entails that a global standard deviation for the DK-DEMs is used. Based on this the standard deviation of the difference will vary from cell to cell.

7.2 Choice of data and methods

Based on the problem analysis a choice of which data and methods to use in the remaining part of the project will be made and clarified in the following. This will lead to the problem statement. Of the documented data and methods it is decided to use the following onwards:

- Data
 - UAV point cloud
 - DK-DEM
- Methods
 - Adjustment of planes for quality assessment
 - Subtraction and evaluation of differences on a cell basis for change detection

The data foundation for updating the DK-DEM is chosen to be the UAV point cloud from GeoFyn. The point cloud is the raw data and therefore no unknown data processing has taken place, which is preferable for the project group. It is unavoidable to use the DK-DEM when detecting changes, which is why this data source also is used.

As it was experienced that the UAV point cloud has several inaccuracies, it is decided that a quality assessment is necessary. For this reason it is chosen to carry through with the quality assessment based on adjustment of best fit planes. This will quantify the quality, but also be the basis of removal of outliers.

As time do not permit it, it is decided not to prioritise a categorisation of the GeoFyn dataset. In Chapter 13, *Discussion* it will be considered which tests that can be performed in order to determine which categorisation methods that possibly could be used.

Change detection is seen as the essential element of the project and therefore it is prioritised. The change detection will be based on a subtraction between the UAV dataset and the DK-DEM and the difference will be evaluated based on the accuracy of the two products. As the quality of the DK-DEM/Surface and the DK-DEM/Terrain is good, one of these products will be used. A further analysis is needed in order to determine which of the products should be used. Depending on the results and if time allow it the DK-DEM/PointCloud could also be taken into consideration.

Based on the described decisions the following problem statement can be formed:

How can the chosen methods, for quality assessment and change detection of terrain, be implemented for updating the Danish Elevation Model?

To answer the problem statement several analyses have to be made in order to determine the final methods, variables and data that should be used for the quality assessment and change detection. Further questions or problems can arise, which need to be answered during the coming analyses.

In the following chapter the method for answering the problem statement will be described.

Method 8

The present chapter will describe the method of the following chapters, which treat the implementation and final decisions of the choices made in Chapter 7, *Problem statement*. Through the following chapters the aim is to answer the problem statement and get closer to the goal of updating the Danish Elevation Model.

In Figure 8.1 the structure of the final part of the project is illustrated, which is a continuation of the structure showed in Chapter 2, *Problem analysis method*. Followed by the figure the overall method for each step will be described and in the relevant chapter the method will be further explained.

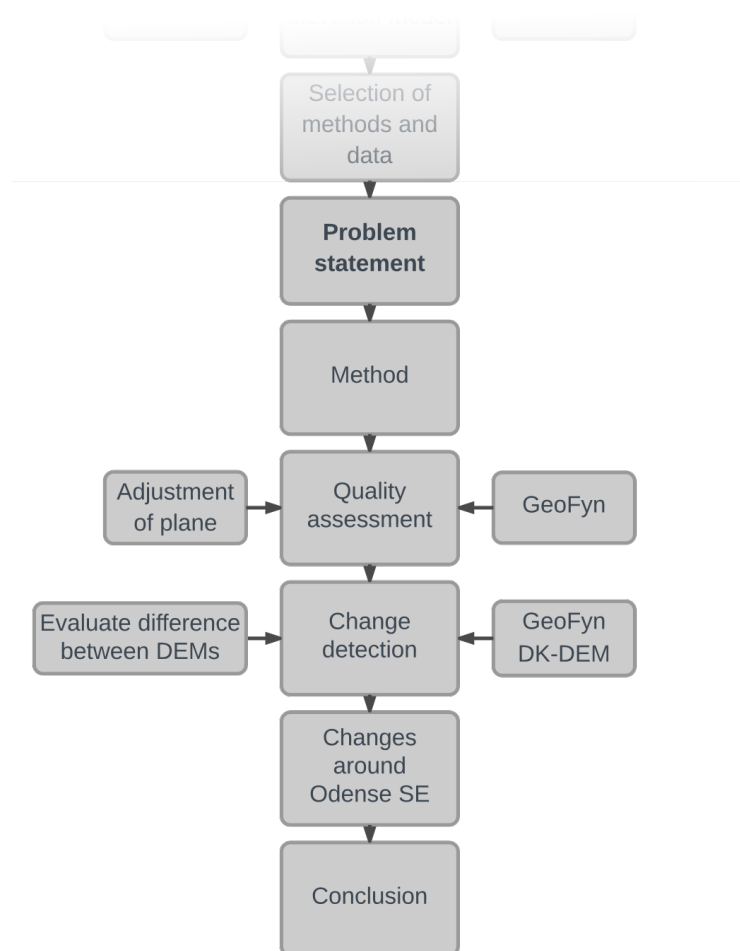


Figure 8.1: Structure diagram of the final part of the project

The problem statement rise the question about implementation. Implementation comprise everything from early considerations about methods and data sources to programming and execution of the final calculations.

To test the chosen methods and data it is decided to use the MATLAB desktop environment, which enable the project group to program routines for different experiments of the chosen methods and data. The developed MATLAB scripts are not seen as the final products for SDFE, but a demonstration of the implementation of the methods. It is thought that SDFE want the final algorithm to be written in a non-proprietary programming language, but this will not be focused on. During the programming of the scripts the effort of optimising and streamlining them will not be prioritised.

8.1 Quality assessment

In the problem analysis in was concluded that a quality assessment of the GeoFyn UAV data is necessary to quantify the quality of the dataset. The quality of the dataset is of interest as the information did not follow with the dataset. It is also of interest as a value expression the quality is a step on the way to filter the dataset for outliers.

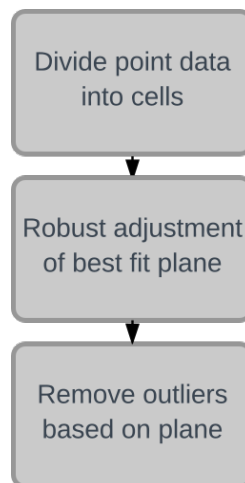


Figure 8.2: Quality assessment

The quality assessment will be carried out through three steps; dividing the data into cells, calculation a best fit plane by adjustment methods and removing outliers in each cell based on the precision of the data, see Figure 8.2. The main part of this step concerns the adjustment and the applied theory will be described.

8.2 Change detection

In the chapter concerning change detection it will be investigated in which areas of Odense SE that a change has happened. This results in a raster mask with the values “0” and “1”, where “1” determines where the Danish Elevation Model should be updated. The change detection will be performed for four test areas and several tests will be carried out concerning which data to use and how the difference from the subtraction is evaluated. The choice of data concerns which of the DK-DEM products that will be used.

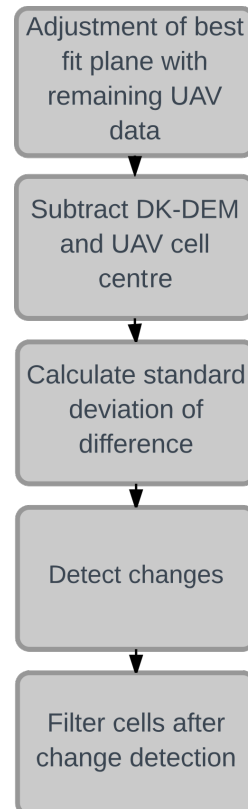


Figure 8.3: Change detection

The change detection is covered in five steps, see Figure 8.3. Firstly an adjustment of planes will be calculated for the dataset without outliers. Based on the planes the elevation of the centre of the cells is found, which will represent the UAV data in the change detection. Followed by this is the subtraction of the DK-DEM from the UAV data. For every subtraction the standard deviation of the difference can be calculated. The standard deviation of the difference is a result of the standard deviation of the compared products. For the UAV data different contributions for the standard deviation exists, which will be discussed. When the differences are calculated and the standard deviation of the difference is found, the change detection can be carried out. This will result in a mask where it is determined for every cell if a change has happened or not. The raster mask will be evaluated for the result from both the DK-DEM products and finally the mask will be filtered to avoid “salt and pepper” noise, where individual cells are surrounded by cells with a different value.

8.3 Changes around Odense SE

When the change detection has been carried out for the test areas and the final choices are made, the whole process will be carried out for the whole case area - Odense SE. This should give the final answer to where changes has happened and where the Danish Elevation Model should be updated, but it will also clarify where some of the weaknesses are in the algorithm.

8.4 Conclusion and discussion

Finally a conclusion will sum up the results and the experiences attained during the project. A discussion will consider further tests of categorisations and what they could include.

Quality assessment 9

The present chapter will focus on how the quality assessment can be implemented in the algorithm, which will clarify the quality of the UAV data and based on this remove outliers, resulting in a new dataset. The chapter will both focus on the theory used in the algorithm and the obtained results from calculations and tests.

Figure 9.1 shows the steps included in the quality assessment along the centre line. Each step, apart from *Read UAV point cloud*, are specified in the light gray boxes.

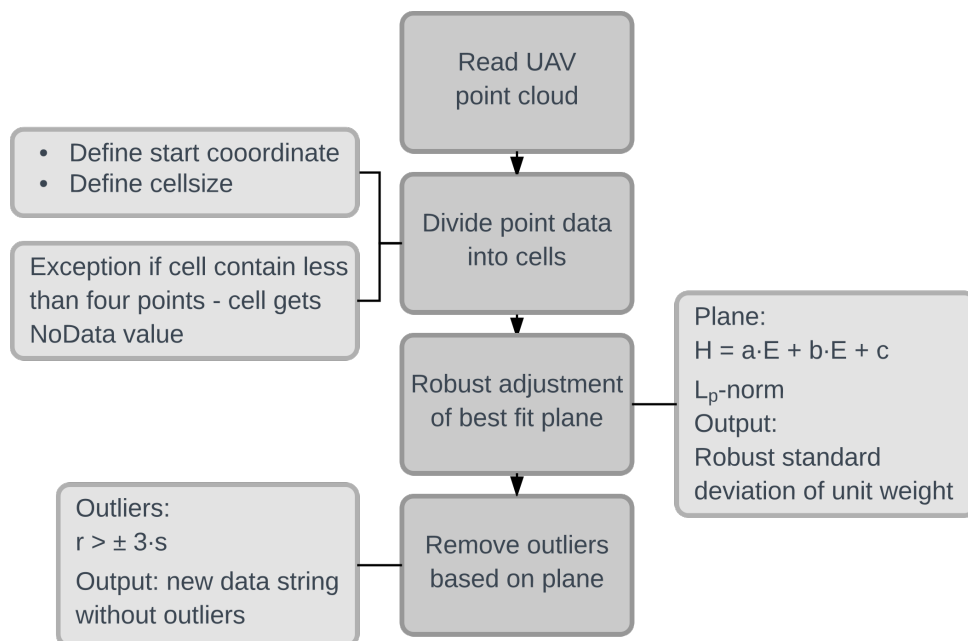


Figure 9.1: Workflow illustrating the steps of quality assessment

The first step *Read UAV point cloud (LAS)* involve loading a .LAS-file into MATLAB and reading it, resulting in a matrix containing E,N,H. The .LAS-file contain more information than this, but only E,N,H is needed here. The file is read with a tool called *lasdata*. [Kumpumki, 2014] This process will not be commented on further.

Secondly the data is divided into cells, which make it possible to perform an adjustment of a plane for the points inside the cell, but also prepare the dataset for comparison with the DK-DEM. The considerations and technique for dividing the data into cells is described in

the following section.

When the data is in a grid it is possible to carry out the third step; robust adjustment of a plane. This is done for two reasons; to assess the quality and to detect and remove outliers. To decide which L_p -norm to minimise according to, a test is carried out with different norms. Even though it has been decided that a robust adjustment will be used, the test will also evaluate a L_2 -norm least square adjustment. The adjustments will be evaluated based on the standard deviation of unit weight.

Lastly points which are outliers in the adjustment will be removed, but the number of removed outliers depend of the chosen L_p -norm. Therefore the test of different L_p -norms will also be evaluated on how many outliers are removed. Finally it will be summarised what L_p -norm is chosen and the results will be presented.

9.1 Dividing point cloud into cells

When dividing the UAV data into cells it is important to consider the cell size and the starting point of the grid. The points are divided into cells based on the E- and N-coordinates and as the coordinates are increasing going east and north the starting point of the UAV data grid is defined at the lower left corner for convenience. The points are divided into cells so the adjustment of a plane can be carried out for each cell and so the comparison with the DK-DEM can be made.

To begin with it is decided to use the DK-DEM raster models for comparison with the UAV data for change detection, which affects the choices when dividing the UAV data into cells. The cell size have to cover an area such that enough points fall within it. It is obvious to try a cell size of 0.4×0.4 m, as the DK-DEM raster has this format. A random chosen area of 100×100 m contain 785213 points, which on average gives 13 point per cell if the cells are 0.4×0.4 m. This seems as a reasonable amount of points per cell and the data processing will carry through with this cell size.

The starting point of the UAV data can be arbitrarily chosen as long as the E and N coordinates are dividable by the cell size. If the starting point is not dividable with the cell size it will be shifted until it is.

After these choices have been made the UAV data can be divided into cells. When dividing the data into cells a minimum criterion has been set concerning the number of points inside the cell. If the cell contain less than four points, then a NoData value of '-9999' will be given to that cell and no coordinates will be within it. This is done as an adjustment of a plane for a cell that has less than four points can either not be performed or the adjustment is not overdetermined. If the cell contain a low number of points, e.g. four or five points, the adjustment is carried through, even though a higher number of redundancy is preferred.

The MATLAB script for dividing the data into cells can be seen in Appendix D, *data_to_cells.m* and also be found in Appendix B, *Digital appendix*.

9.2 Robust adjustment of plane

As the data is divided into cells the quality assessment of the UAV data can be carried out. The purpose of the quality assessment, which is based on an adjustment of a plane, is to quantify the precision of the data. As it is expected that the data contain outliers a robust adjustment will be made, which also will form the basis of removal of outliers.

The following will firstly introduce the theory of robust adjustment of a plane in relation to handling the UAV data. Secondly a test with different L_p -norms will be carried out. The test will be evaluated based on the standard deviation of unit weight and the quality of the data will be assessed.

The robust adjustment is performed with the MATLAB function “*planfit.m*”, which can be seen in Appendix E, *planfit.m* and also be found in Appendix B, *Digital appendix*.

9.2.1 Robust adjustment of plane and implementation

To perform a robust adjustment the method Iteratively Reweighted Least Squares (IRLS) is used. This method reweights the observations and minimise the residuals according to a L_p -norm. By using IRLS outliers which normally would affect an unweighted least square adjustment can be eliminated. The IRLS technique with the L_p -norm is a variation of a least square adjustment and will be described in the following.

The elevation (H) for a point on a plane can for a given position (E, N) be calculated with the following equation:

$$H = aE + bN + c \quad (9.1)$$

Where

H, E, N : Coordinates of the points

a, b : Slope coefficients

c : H -axis intercept

Eq. (9.1) is not a general equation of a plane, which means that the following adjustment can not handle vertical planes. This should not be a problem as it is unlikely that a vertical plane will occur in the case area. Because of this the scripts will have no safeguards against vertical planes and the adjustment can potentially fail. To prevent this a more general equation for a plane could be used, but this will make the adjustment more complicated, as the observation equations will be over-parametrised. This results in the normal equations being singular. To make the normal equations invertible, a constraint of the normal vectors length, $[a \ b \ -1]^T$, have to be made. [Cederholm, 2016a] As mentioned vertical planes are not expected to occur and for this reason the more simple approach is used.

As the E and N coordinates of the UAV data are in the reference system ETRS89, UTM zone 32N a reduction of the coordinates, in order to avoid numerical problems in MATLAB, is needed. This can be done by subtracting the mean coordinates of all involved points to the m'th E, N and H coordinate:

$$\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_m \end{bmatrix} - \frac{\sum_{i=1}^m E_i}{m} \quad \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_m \end{bmatrix} = \begin{bmatrix} N_1 \\ N_2 \\ \vdots \\ N_m \end{bmatrix} - \frac{\sum_{i=1}^m N_i}{m} \quad \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_m \end{bmatrix} = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_m \end{bmatrix} - \frac{\sum_{i=1}^m H_i}{m}$$

In the previous section the division of the points into cells were described and in each cell at least four points are present. This makes it possible to set-up a overdetermined system of m observation equations: [Wolf and Ghilani, 1997, p. 177]

$$\begin{aligned} h_1 + r_1 &= ae_1 + bn_1 + c \\ h_2 + r_2 &= ae_2 + bn_2 + c \\ &\vdots \\ h_m + r_m &= ae_m + bn_m + c \end{aligned} \tag{9.2}$$

In matrix notation this can be written as: [Wolf and Ghilani, 1997, p. 181]

$$\mathbf{b} + \mathbf{r} = \mathbf{A} \times \mathbf{x}$$

$$\begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_m \end{bmatrix} + \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix} = \begin{bmatrix} e_1 & n_1 & 1 \\ e_2 & n_1 & 1 \\ \vdots & \vdots & 1 \\ e_m & n_m & 1 \end{bmatrix} \times \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{9.3}$$

Where

b : vector with h-coordinates of the points in the cell

r : vector with the residuals

A : Design matrix

x : Vector with the unknown coefficients

After the observation equations are put in matrix notation, the normal equations can be set-up and solved by a weighted least square adjustment: [Wolf and Ghilani, 1997, p. 183]

$$\begin{aligned} \hat{\mathbf{x}} &= (\mathbf{A}^T \times \mathbf{W} \times \mathbf{A})^{-1} \times \mathbf{A}^T \times \mathbf{W} \times \mathbf{b} \\ \hat{\mathbf{x}} &= \mathbf{N}^{-1} \times \mathbf{A}^T \times \mathbf{W} \times \mathbf{b} \\ \hat{\mathbf{x}} &= [\hat{a} \ \hat{b} \ \hat{c}]^T \end{aligned} \tag{9.4}$$

Where

W : Weight matrix with weights on the diagonal

The estimated residuals can be calculated by:

$$\hat{\mathbf{r}} = \mathbf{A} \times \hat{\mathbf{x}} - \mathbf{b} \quad (9.5)$$

The weight of each point can be calculated according to different approaches, for example if they are independently measured and the variances are known the weight for each point can be expressed as $w_i = \frac{1}{\sigma_i^2}$. If the variance of the points are unknown the weight can be set to $w_i = 1$, which is the same as doing an unweighted least square adjustment. [Wolf and Ghilani, 1997, pp. 156-157] Using IRLS the weight iteratively changes according to the following weight function: [Cederholm, 2016b]

$$w_i = |\hat{r}_i|^{p-2} \quad (9.6)$$

Where

$|\hat{r}_i|$: Absolute value of the i'th residual

p : The chosen L_p -norm, where $1 < p < 3$

It is seen that if $p = 2$, w_i becomes:

$$w_i = |\hat{r}_i|^{2-2} = |\hat{r}_i|^0 = 1 \quad (9.7)$$

This is the same as the unweighted case. If $p = 1$, w_i becomes:

$$w_i = |\hat{r}_i|^{1-2} = |\hat{r}_i|^{-1} = \frac{1}{|\hat{r}_i|} \quad (9.8)$$

This means that if the residual is large it will get a smaller weight. From eq. (9.8) it can be seen that a problem will arise if $r_i = 0$, as a division by zero will occur. To prevent this a very small value is added to the residual. The value added is the machine epsilon, *eps*, which is “the distance from 1.0 to the next larger double-precision number” multiplied by 100 to avoid numerical problems. [MathWorks, 2017] [Cederholm, 2016b] The final weight function will be:

$$w_i = (eps \cdot 100 + |\hat{r}_i|)^{p-2} \quad (9.9)$$

The L_1 -norm is quite strict regarding what is seen as an outlier and for this reason an array of other norms and weight functions exist. Generally the bigger the norm the less robust the adjustment will be. For example the $L_{1.3}$ -norm is less robust but more forgiving in relation to which points are seen as outliers. An example of the weight function for the L_2 -, $L_{1.3}$ - and L_1 -norm can be seen in Figure 9.2

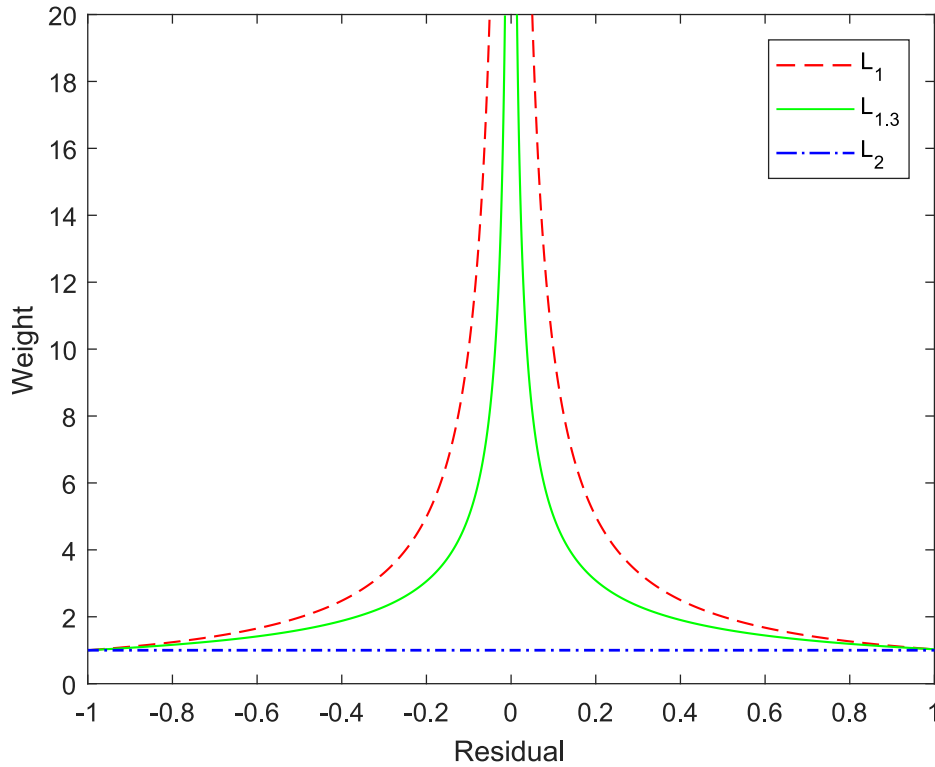


Figure 9.2: Weight function of three L_p -norms

As seen in Figure 9.2 the L_2 -norm gives every point the same weight. The L_1 -norm gives points with a low residual a large relative weight. The $L_{1.3}$ -norm is similar to the L_1 -norm but as seen, the points must have a lower residual before it will get a large relative weight.

The implementation of IRLS can be done through the following steps: [Cederholm, 2016b]

1. Choose L_p -norm
2. Calculate an unweighted least square adjustment
3. Calculate residuals with eq. (9.5)
4. Set up a new weight matrix with eq. (9.9)
5. Calculate a least square adjustment with new weight matrix
6. Repeat step 3. to 5. until a stop criterion is met

The stop criterion will in this case both be a minimum value describing the difference between the current and the previous iteration and a maximum number of allowed iterations. Only one of the criteria have to be fulfilled in order to stop the iteration, but the main reason behind the maximum number of allowed iterations is to prevent an eternal loop.

The value describing the difference between the current and the previous iteration will be calculated as the maximum absolute deviation between the estimated coefficients of the current and previous iteration. It is chosen that if the value is below 0.001 the loop

will terminate. For a 0.4×0.4 m plane this value will only let the H -axis intercept, c , change by 0.001 m and the vertical distance for the slope coefficients, a and b , change by 0.001×0.4 m = 0.0004 m. This stop criterion is thought to be small enough in order to avoid inaccuracies in the robust adjustment. The maximum number of allowed iterations is set to 150. These criteria makes it possible to form the following logic sentence, which if true will terminate the loop:

$$\max(|\hat{\mathbf{x}}|) < 0.001 \mid i = 150$$

Where

i : number of iterations

After the loop is terminated, the final residuals are calculated, which are used to determine the standard deviation of unit weight. The standard deviation of unit weight is used to evaluate whether or not the points are outliers. If the L_2 -norm is used, hence a non-robust method, the standard deviation of unit weight is given by eq. (6.6) and if a L_p -norm leading to a robust result is used, it is given by eq. (6.7). Both equation are repeated below:

$$\hat{\sigma}_0 = \sqrt{\frac{\hat{\mathbf{r}}^T \mathbf{C} \hat{\mathbf{r}}}{m - n}} \quad (6.6)$$

$$s = \frac{\text{median}(|(\hat{\mathbf{r}} - \text{median}(\hat{\mathbf{r}})|))}{0.6745} \quad (6.7)$$

The denominator in eq. (6.7) is found such that $s \approx \sigma_0$ when the number of observations is large. To see if this is correct a small test with random generated observations have been created in MATLAB. The observations are 10,000 2D points distributed such that the y-coordinate is affected by a random error with a mean of 0 and a standard deviation of 1. The points are used in a L_1 - and L_2 -norm adjustment calculating a best fit 2D line. It is expected that σ_0 for the L_2 -norm adjustment should be approximately the same as s for the L_1 -norm adjustment. The results are:

$$\begin{aligned} \hat{\sigma}_0 &= 1.011 \\ s &= 1.005 \end{aligned}$$

From this it can be seen that the number in the denominator of eq. (6.7) is reasonable. The script “*test_sigma_0_vs_s.m*” which the test is performed in can be seen in Appendix J, *test_sigma_0_vs_s.m* and also be found in Appendix B, *Digital appendix*.

9.2.2 Test: Standard deviation of unit weight

A test of different L_p -norms for adjustment of planes is carried out in the following. It is expected that the dataset is subject to outliers and this can be confirmed or disconfirmed by comparing the results of a non-robust and a robust adjustment. The different norms are also tested to decide which one to use for the robust adjustment, if it turns out the data is affected by outliers.

The tests are carried out for four test areas. The test areas are chosen to have different characteristics, as it previously was illustrated that for example the surface of the highway road was inaccurate. The four areas are 20×20 m corresponding to 50×50 cells and represent low vegetation, field, highway road and a steep slope, see Figure 9.3.



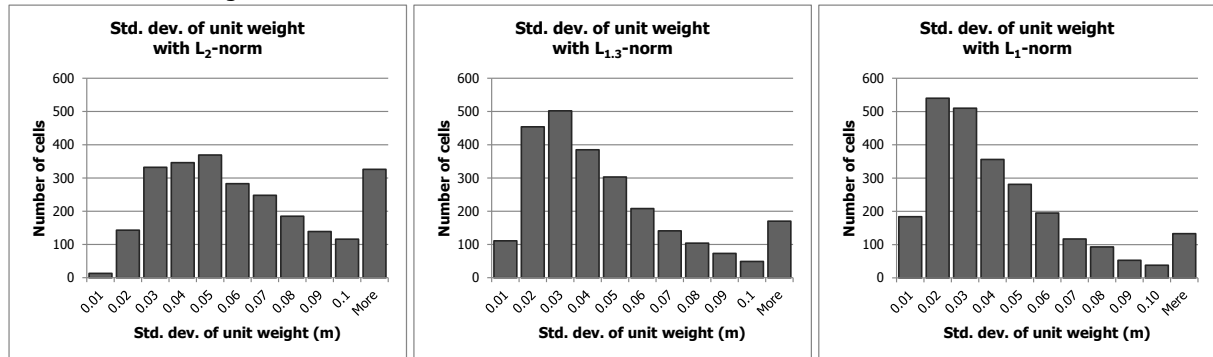
Figure 9.3: Four test areas representing areas with different characteristics. The background displays the UAV orthophoto from GeoFyn

Three different norms are tested; L_2 , $L_{1.3}$ and L_1 . The planes are calculated with L_2 -norm to see the precision of the data without weights. This will also create a reference for the results of the other norms. The L_1 -norm is chosen as it is the norm that is least sensitive to outliers. The $L_{1.3}$ -norm is also tested, which also is robust, but not as sensitive to outliers.

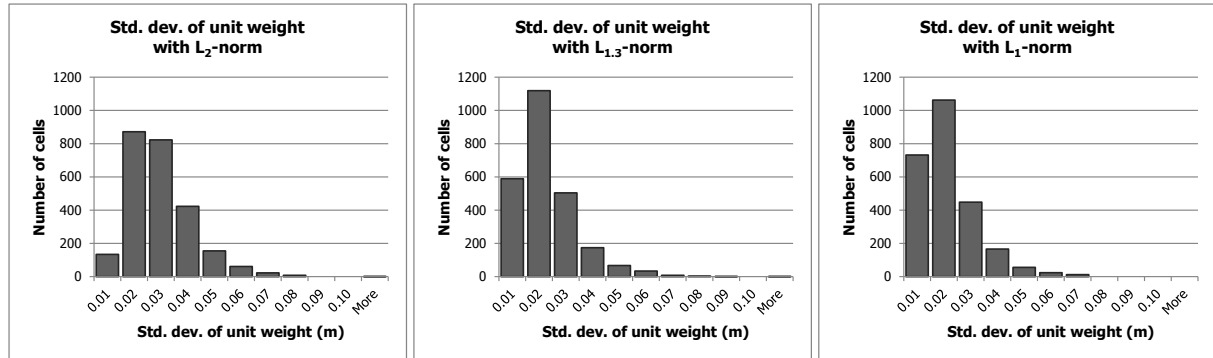
The planes are calculated for each of the test areas according to the theory described above. For every adjusted plane the standard deviation of unit weight is calculated and all values are

illustrated in a histogram, see Figure 9.4. The average of the standard deviation of unit weight for each scenario are also listed in Table 9.1. The average is calculated as the mean value of the squared σ_0 or s , where the square root was taken of the mean value afterwards.

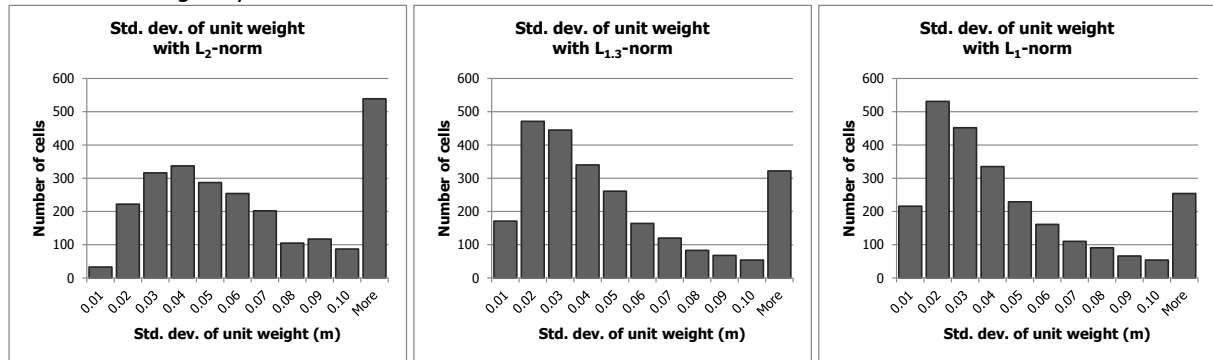
Test area 1 - low vegetation



Test area 2 - field



Test area 3 - highway road



Test area 4 - steep slope

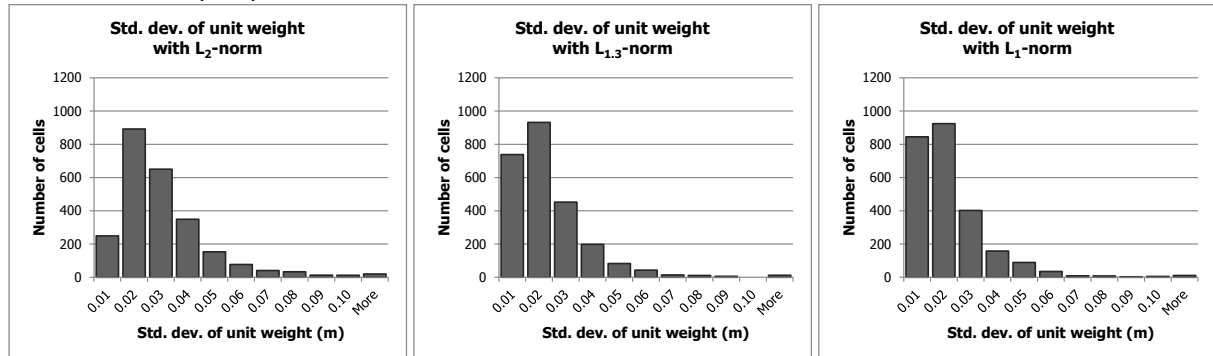


Figure 9.4: Standard deviation of unit weight illustrated for the four test areas based on the different L_p -norms. NB The vertical axes vary between the test areas

Table 9.1: Average of the standard deviation of unit weight

Test area	L_2	$L_{1.3}$	L_1
1 - low vegetation	0.071 m	0.057 m	0.052 m
2 - field	0.027 m	0.021 m	0.020 m
3 - highway road	0.095 m	0.079 m	0.072 m
4 - steep slope	0.031 m	0.024 m	0.023 m

Quality based on L_2 -norm

The first column of histograms shows the standard deviation of unit weight based on the L_2 -norm, hence all observation are evenly weighted. It clearly shows that the precision of the data in test area 1 and test area 3 is lower than test area 2 and test area 4. For test area 2 and 4 the interval with most values is narrower than test area 1 and 3 and it also peaks at a lower standard deviation. This quantifies the inaccuracies on the road and also that the surface representing low vegetation is uneven.

Test area 2 and test area 4 shows similar results, which indicates that the adjustment of a plane goes well for sloped areas too. The average of the standard deviation of unit weight for area 2 gives 0.027 m and is the lowest for the four test areas. This is seen at the best possible quality of the dataset.

Check for outliers with L_1 -norm and $L_{1.3}$ -norm

For all test areas it is seen that the histogram shifts to the left when going from L_2 -norm to $L_{1.3}$ -norm and again to L_1 -norm, hence the residuals, which are weighted, and therefore the standard deviation of unit weight decrease for the robust adjustments. This is expected when the dataset is subject to outliers, why the results confirm the assumptions about outliers in the UAV dataset.

Minimizing according to the L_1 -norm sometimes leads to too low weights for observations that are not outliers, hence it is too strict. This issue is illustrated in Figure 9.5 as a 2D line, but the same apply to a 3D plane. It is seen for the L_1 -adjustment that the middle point has no influence on the line, hence it has been given a very low relative weight. The line based on $L_{1.3}$ -norm is slightly shifted towards the middle point and based on L_2 -norm even more. [Cederholm, 2016b]

Based on this it is proposed to continue with the $L_{1.3}$ -norm, as the influence of the outliers have been lowered, which clearly shows when comparing the standard deviation of unit weight between L_2 and $L_{1.3}$. The strict nature/disposition of L_1 and the fact that the standard deviation of unit weight only improves slightly from $L_{1.3}$ supports the decision. In the following it is showed how the different L_p -norms affect the number of removes points and based on this and the present suggestion a final choice will be made.

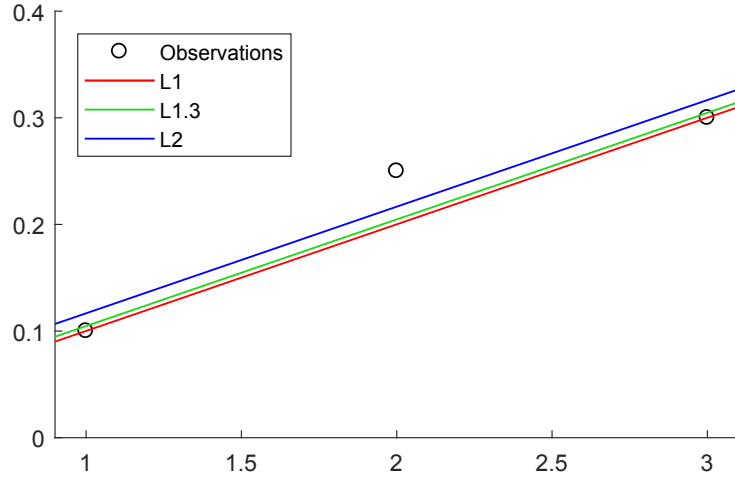


Figure 9.5: Test of different L_p -norms for adjustment of a 2D plane. The figure is based on [Cederholm, 2016b]

9.3 Remove outliers

As the planes are calculated for all cells a removal of outliers can be carried out. Outliers are detected based on the standard deviation of unit weight, which is calculated for every plane, either with the normal or robust equation. It is desired to remove outliers to construct a new dataset, that has been filtered for the expected inaccuracies in the UAV dataset.

The detection of outliers is based on statistical methods. It is presumed that the residuals are normal distributed with mean value $\mu = 0$ and variance $\sigma^2 = \sigma_0^2$ or $\sigma^2 = s^2$ depending on the adjustment being robust or not, hence

$$\text{Ordinary: } \hat{r} \sim \mathcal{N}(0, \sigma_0^2) \text{ or robust: } \hat{r} \sim \mathcal{N}(0, s^2)$$

The interval for detecting outliers is commonly set to $\pm 3 \cdot \sigma$ as the probability is:

$$P(-3 \cdot \sigma_0 \leq \hat{r} \leq 3 \cdot \sigma_0) = 0.997 \quad (9.10)$$

Hence for a normal distributed dataset 0.3 % of the data is presumed to be outliers. [Eriksen et al., 2015]

The MATLAB script for removing outliers can be seen in Appendix F, *remove_outliers.m* and also be found in Appendix B, *Digital appendix*.

9.3.1 Test: Number of outliers

The chosen L_p -norm for the adjustment of planes impacts the number of detected outliers, which is why a test of different norms will be carried out. As it was seen in Section 9.2.2, *Test: Standard deviation of unit weight* the standard deviation of unit weight was smaller for L_1 than L_2 , hence $\pm 3 \cdot \sigma$ leads to a smaller interval. As the interval is smaller more points will be detected as outliers.

The test is carried out for the same four test areas as presented in Section 9.2.2, *Test: Standard deviation of unit weight* and L_2 , $L_{1.3}$ and L_1 is tested. The test will contribute to the decision about which norm to use onwards. The following illustrates how many outliers that are removed for each cell and also a percentage of removed point inside the test area. The number of outliers for each cell is found by a simple count and displayed in histograms, see Figure 9.6. The percentage is calculated by summarising the number of outliers for all cells (2500 cells) and diving by the total number of points in the test area. The percentages are shown in Table 9.2 along with the total number of points in the test area.

Figure 9.6 shows that hardly any points are removed when using the L_2 -norm. For test area 1 and 3, which was the least precise areas, 0.1 % and 0.2 % respectively have been removed. As all observations have entered into the adjustment with equal weights, it is understandable that only a few points have been removed. Large residuals leads to a larger standards deviation of unit weight and this leads to a larger interval for detecting outliers.

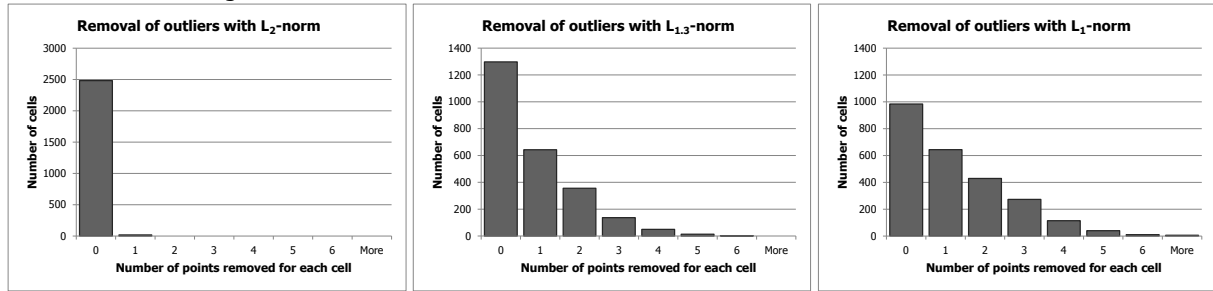
The result from the $L_{1.3}$ -norm and the L_1 -norm shows that considerably more points have been removed, even three or more points per cell for a number of cells. For $L_{1.3}$ -norm around 6.0 % have been removed, but in roughly half of the 2500 cells no points are removed. For the L_1 -norm the tendency is very similar, but there is less cells where no points are removed. The percentage of removed points has also gone up to around 8.0 %. The additional percents that are removed are points with smaller residuals, as the interval for removal of points is narrowed with L_1 caused by the lower standard deviation of unit weight.

The results of the removal of outliers is as expected, the L_1 -norm removes the most points and L_2 -norm the least. As the difference is not large between $L_{1.3}$ and L_1 , it is chosen to continue with the $L_{1.3}$ -norm as the “worst” outliers must be removed with the $L_{1.3}$ -norm.

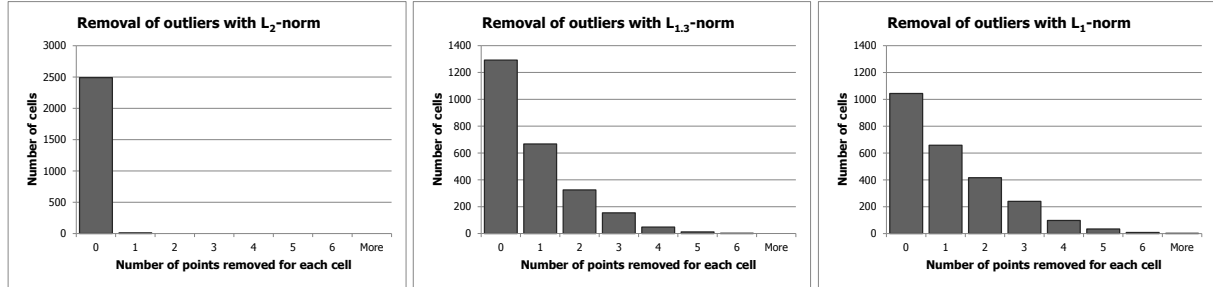
Table 9.2: Percentage-wise removal of outliers for the four different areas with three different norms

Test area	Points total	L_2	$L_{1.3}$	L_1
1 - low vegetation	35529	0.1 %	5.8 %	8.7 %
2 - field	35295	0.0 %	5.8 %	8.0 %
3 - highway road	41968	0.2 %	6.0 %	8.6 %
4 - steep slope	35267	0.0 %	5.7 %	7.9 %

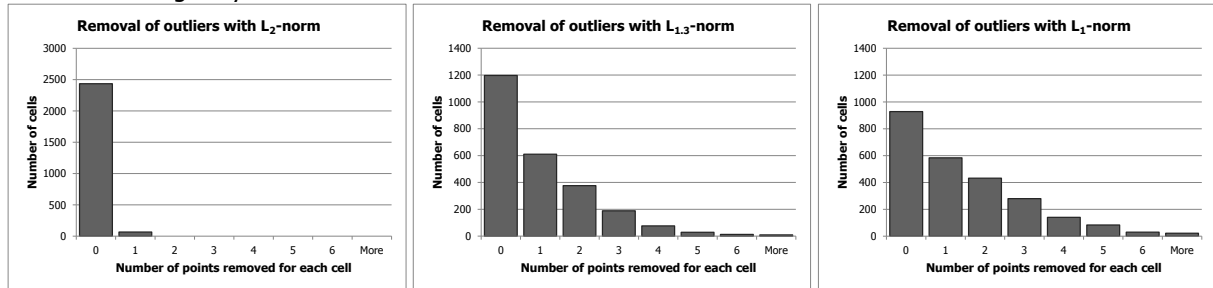
Test area 1 - low vegetation



Test area 2 - field



Test area 3 - highway road



Test area 4 - steep slope

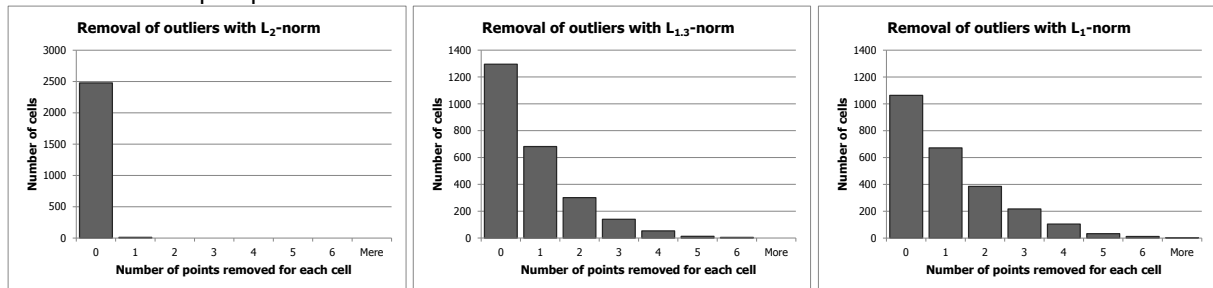


Figure 9.6: Histogram showing the number of removed points for each cell. NB The vertical axes vary between the L_p -norms

9.4 Summary

In this chapter it was described how the UAV data is divided into cells, the theory behind a robust adjustment and the test of L_p -norms. Based on the tests of the L_p -norms it was seen that test area 1 and test area 3 had a lower precision, hence the data quality is worse at these areas. Test area 2 and test area 4 had a comparable and higher precision, hence a better quality than test area 1 and 3, which was expected and now quantified.

From Figure 9.5 it was seen that by using the L_1 -norm points can mistakenly be categorised as an outlier and thereby not influence the adjustment. For this reason it is proposed to use the $L_{1.3}$ -norm which is less robust, but more forgiving regarding points which should not be seen as outliers. This is supported by the average of standard deviation of unit weight, seen in Table 9.1, as they do not change considerably.

Points which have a residual larger than $\pm 3 \cdot \sigma$ is seen as being outliers in the adjustment and will be removed. It was seen that when using the L_2 -norm almost no points were removed, this can be expected since the standard deviation in this case is not robust. When the L_1 - and $L_{1.3}$ -norms are used, considerably more points are removed as the robust standard deviation is used. As expected the L_1 -norm is more strict against outliers, as a higher percentage of points have been removed. The difference between the L_1 - and $L_{1.3}$ -norm is not significant and it is thought that using a more forgiving norm is the best approach. For this reason the $L_{1.3}$ -norm will be used for removal of the relative outliers.

The result of the quality assessment is a new UAV dataset without outliers. If a cell after the removal of outliers has less than four points, then the cell will get the NoData value “-9999” and no coordinates will be within it.

Change detection 10

The present chapter will describe how the change detection will be implemented in the algorithm. Several steps have to be performed in order to determine whether a change has occurred between the DK-DEM and the UAV data. Figure 10.1 shows the involved steps and as seen the first process begin from where the last quality assessment step ended.

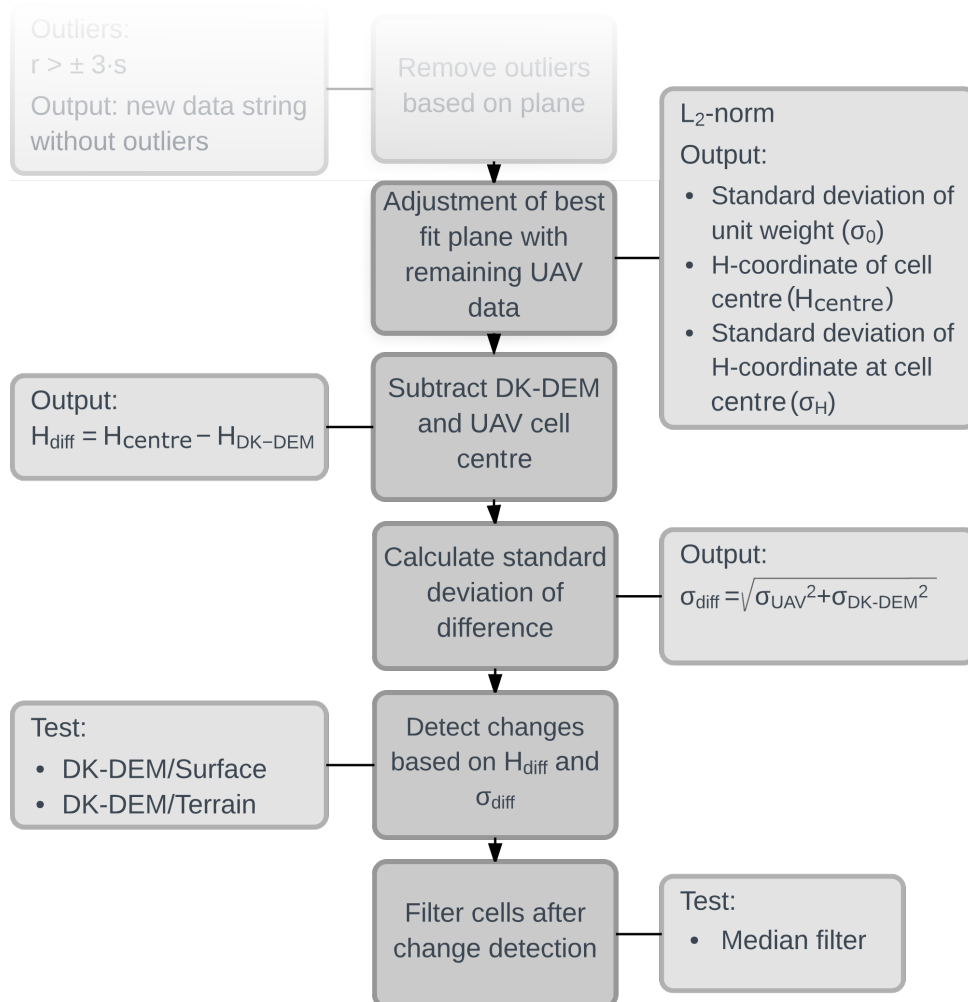


Figure 10.1: Workflow illustrating the steps needed in order to detect changes

Each step of the workflow showed in Figure 10.1 will be described with focus on how it is implemented, the theory involved, which tests that are performed and the results of them.

In the first step of detecting changes an adjustment of planes with L_2 -norm for the new UAV dataset without outliers will be carried out. From this adjustment the standard deviation of unit weight, the H-coordinate at the cell centre and its standard deviation, σ_H , can be calculated. The standard deviation of unit weight is used to see which impact the removal of outliers have had on the UAV dataset. This will be done by comparing a new histogram with the previous histogram showed in Section 9.2.2, *Test: Standard deviation of unit weight*. The elevation of the cell centre and its standard deviation will be used in the following steps.

In the next step the elevations of the DK-DEM/Surface and DK-DEM/Terrain will be subtracted from the elevations of the UAV cell centres. Both DK-DEM products are used to see how the elevation differences differ, as the final change detection step rely on this. Test with both DK-DEM products will be carried out during change detection.

To calculate the standard deviation of the difference the global standard deviation of the DK-DEMs and the standard deviation of the UAV data is used. The standard deviation of the UAV data, σ_{UAV} will be based on the adjustment. Tests with both the standard deviation of the H-coordinate at the cell centre, σ_H , and the standard deviation of unit weight, σ_0 will be performed. Additionally a definition error, σ_{def} , will be introduced.

After the elevation differences and standard deviations of the differences have been calculated, the change detection step can be performed. This is done by calculating an interval in which the difference of the DEMs are not seen as a change. This mean that if the elevation difference is larger than this interval a change is thought to have occurred. The interval will be based on the standard deviation of the difference. The result of the change detection will be a raster mask showing where changes and no changes have happened.

It is expected that the resulting raster mask from the change detection step can have a “salt and pepper” noise, meaning that some of the cells which are detected as a change are isolated and surrounded by non-changed cells or vice versa. A single or few of these surrounded cells can be seen as being wrongly detected and need to be removed. This can be done by a filter, which will be tested in the final step.

As mentioned the present chapter will conduct a number of tests where the parameters for change detection are varied. Based on the results of the tests a final selection of parameters can be made. The parameters are listed in Table 10.1, which is divided into *Data* and *Error contributions for σ_{diff}* .

Table 10.1: Parameters for change detection

Data	
DK-DEM/Surface	
DK-DEM/Terrain	
UAV point cloud	
Error contributions for σ_{diff}	
DK-DEM	σ_{DK-DEM}
UAV	σ_H
UAV	σ_0
UAV	σ_{def}

Table 10.1 shows the data, which comprise the DK-DEM raster models and the UAV point cloud. For the error contributions the dataset and the associated standard deviations are listed, where different solutions exist for the UAV dataset. For every test in the following sections the table will be showed, where the used datasets and error contributions for the relevant change detection is ticked off. Additionally it will be written which parameters that are tested.

10.1 Adjustment of planes with remaining data

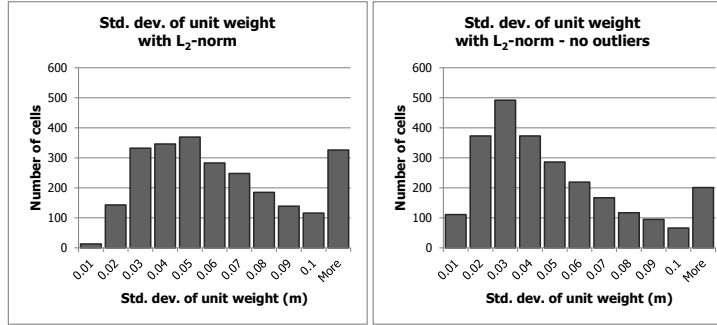
After the original UAV dataset have been quality assessed and outliers removed a new dataset is created, which will be used throughout the remaining part of the algorithm. The new outlier free UAV dataset is already divided into cells. As the data no longer contain any points which are seen as outliers, an unweighted least square adjustment will be performed. The output of the adjustment is for each cell the standard deviation of unit weight, σ_0 , the estimated plane variables, $[\hat{a} \ \hat{b} \ \hat{c}]^T$ and the covariance matrix, $\Sigma_{\hat{x}}$, for the estimated plane.

The MATLAB script for adjustment of planes with remaining data can be seen in Appendix G, *plan_fit_data.m* and also be found in Appendix B, *Digital appendix*.

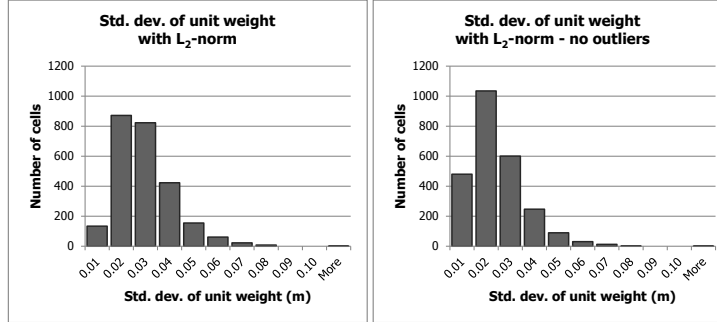
10.1.1 Data quality

In the following the standard deviation of unit weight is commented on, as it is expected that the quality of the dataset has improved. In Figure 10.2 a comparison between the standard deviation of unit weight before and after removal of outliers is illustrated. Additionally the average standard deviation of unit weight can be seen in Table 10.2.

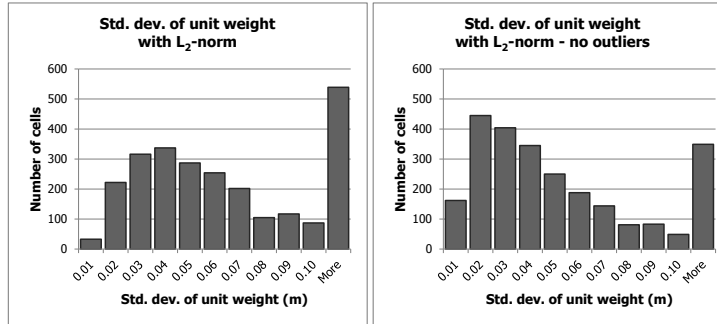
Test area 1 - low vegetation



Test area 2 - field



Test area 3 - highway road



Test area 4 - steep slope

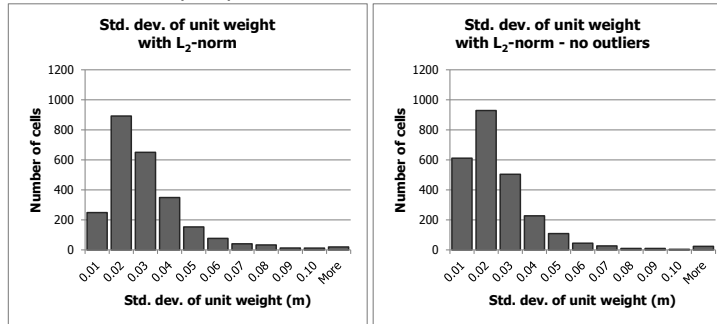


Figure 10.2: Histograms showing the standard deviation of unit weight for all cells based on the L_2 -norm with and without outliers

Table 10.2: The average standard deviation of unit weight for the L_2 -norm with and without outliers

Test area	L_2 - all data	L_2 - no outliers
1 - low vegetation	0.071 m	0.059 m
2 - field	0.027 m	0.022 m
3 - highway road	0.095 m	0.080 m
4 - steep slope	0.031 m	0.025 m

As it can be seen in Figure 10.2 and Table 10.2 removing outliers will generally improve the relative precision of the UAV dataset, which was the goal. It can be seen that the areas which have the lowest data quality, test area 1 and test area 3, benefit the most from having outliers removed, which is expected. Even though the UAV dataset has improved some inaccuracies may still be present, which can affect the change detection. An example of this can be seen in Figure 10.5, where the highway road still seems to have some quality issues. The adjusted planes on the road should be close to horizontal, but for some of the planes this is not the case. The inaccuracies will later be discussed when the results of the change detection are examined.

10.1.2 3D planes

The results of the plane adjustments can be visualized by 3D plane plots, see Figure 10.3 to 10.6. The planes are not exactly 0.4×0.4 m as they are only plotted to the extent of the points in the cell. From inspecting the plane plots it is thought that they are a good representation of the UAV data.

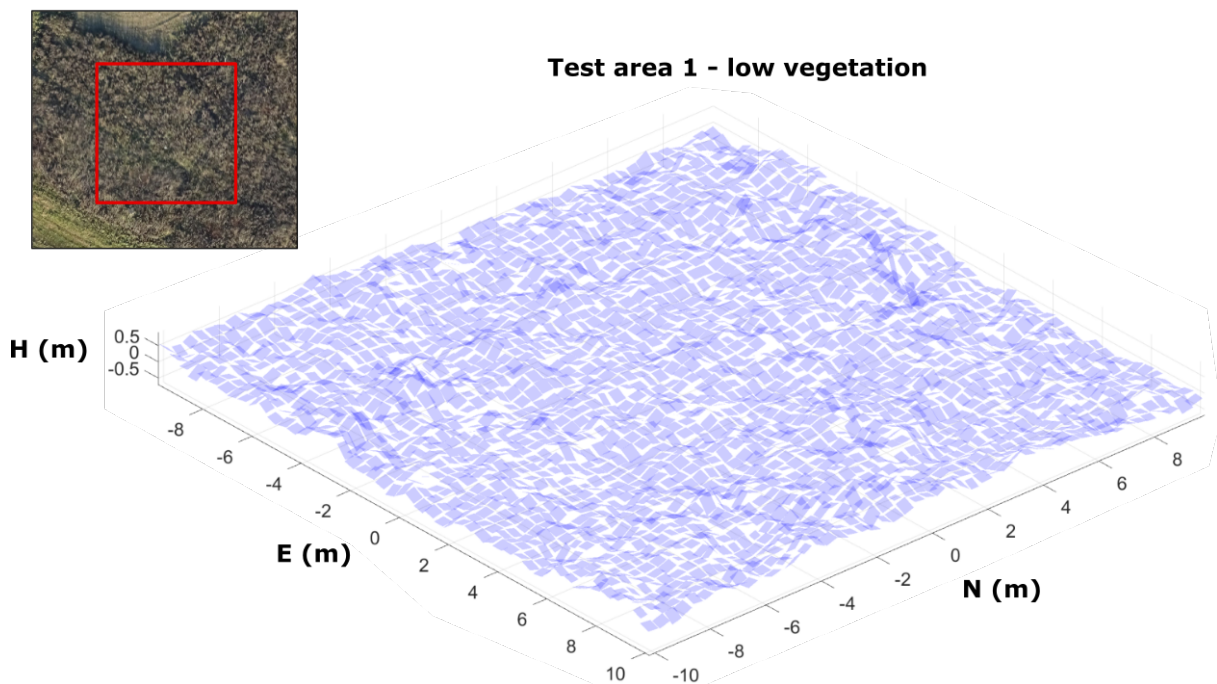


Figure 10.3: 3D plane plot of test area 1 - low vegetation. In the upper left corner the UAV orthophoto is showed with a box marking the test area. NB The E,N,H coordinates are reduced by the mean coordinates of the area

In Figure 10.3 it is seen that the low vegetation in test area 1 is represented by an uneven surface caused by the tilt of the planes, which is expected as the data quality is low.

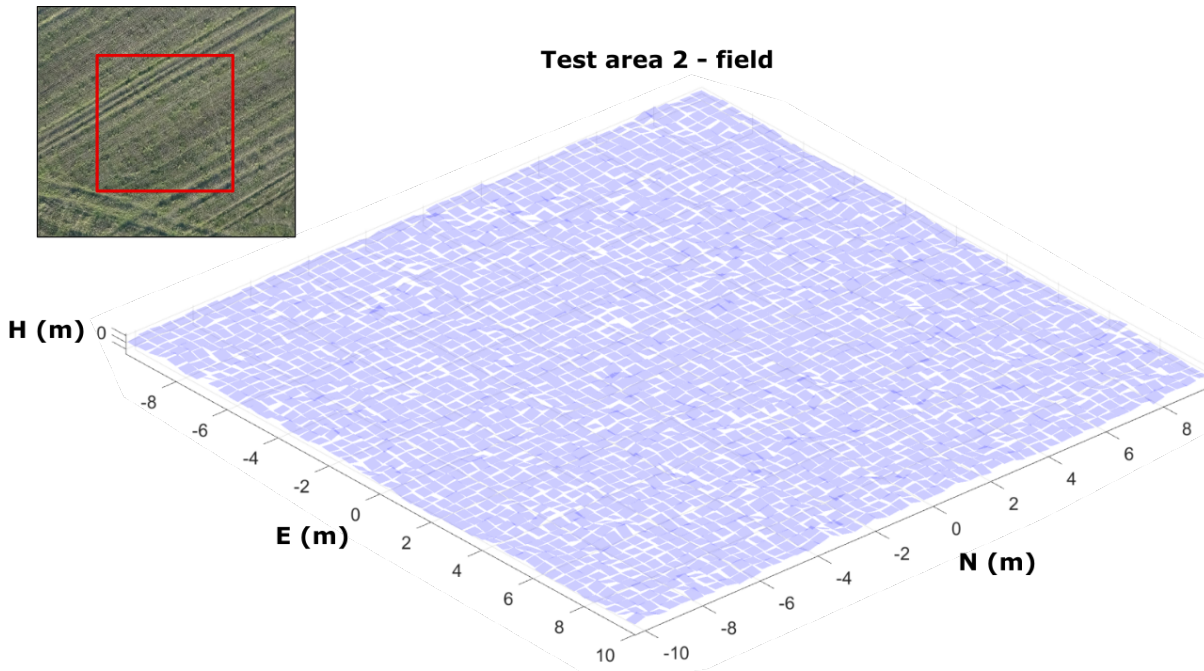


Figure 10.4: 3D plane plot of test area 2 - field. In the upper left corner the UAV orthophoto is showed with a box marking the test area. NB The E,N,H coordinates are reduced by the mean coordinates of the area

In Figure 10.4 it is seen that the planes of test area 2 are close to horizontal, which is expected as the area is flat and it has the highest data quality.

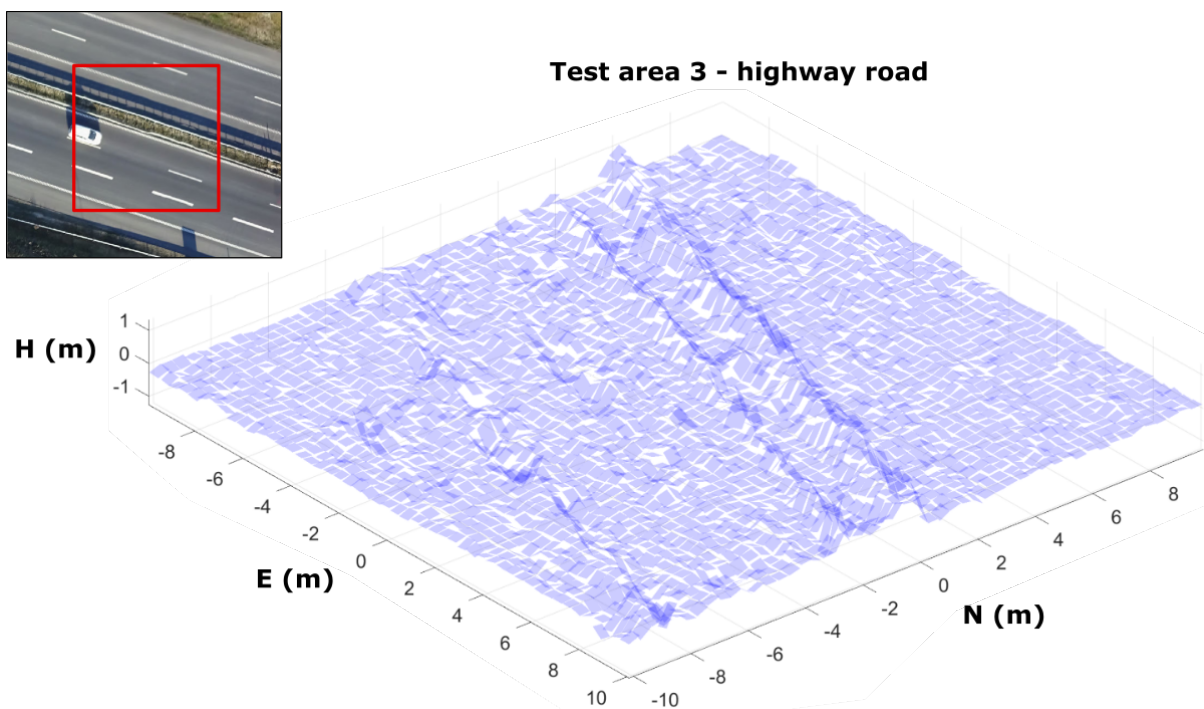


Figure 10.5: 3D plane plot of test area 3 - highway road. In the upper left corner the UAV orthophoto is showed with a box marking the test area. NB The E,N,H coordinates are reduced by the mean coordinates of the area

In Figure 10.5 the mentioned inaccuracies of test area 3 can be seen. Additionally the guardrails separating the lanes are seen as the two peaks and are not to be mistaken for the inaccuracies. This shows a good example of how the generally low data quality of the highway road will affect the adjustment of planes for these areas.

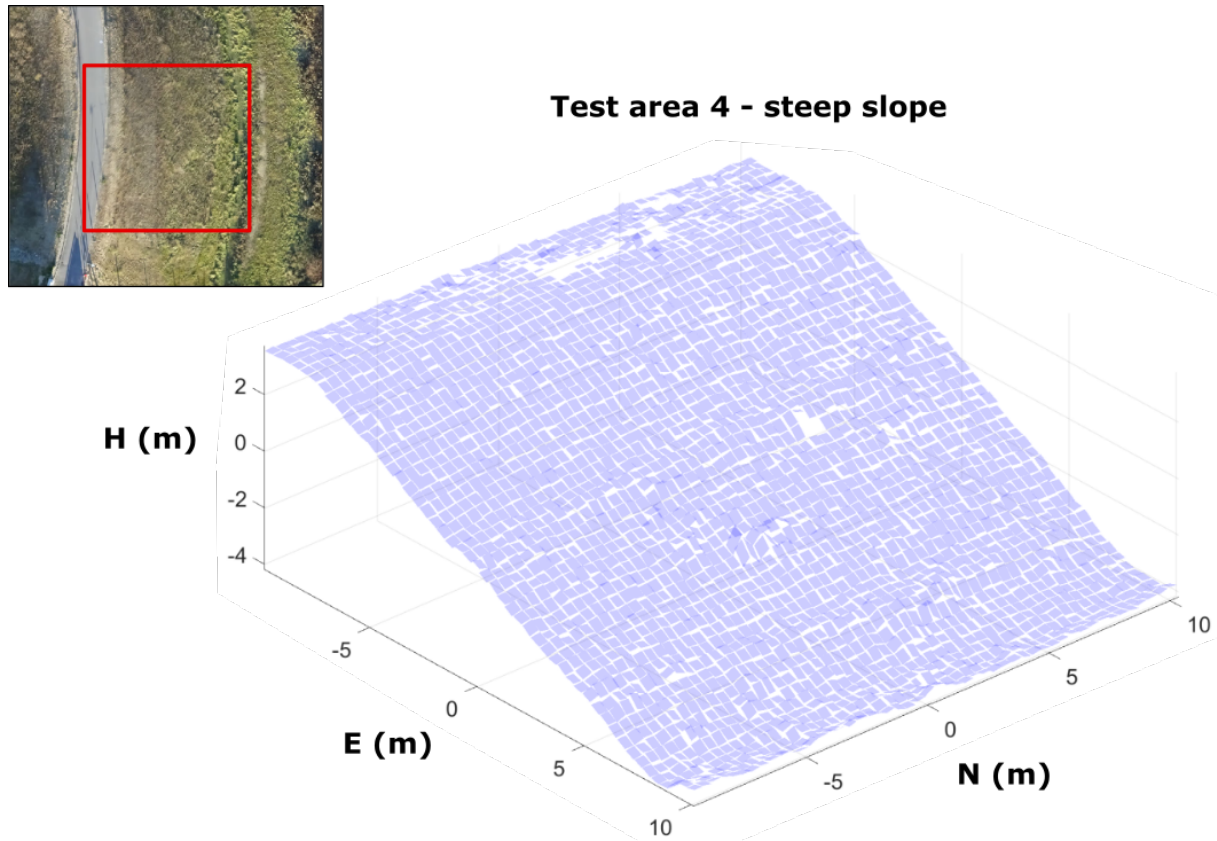


Figure 10.6: 3D plane plot of test area 4 - steep slope. In the upper left corner the UAV orthophoto is showed with a box marking the test area. NB The E,N,H coordinates are reduced by the mean coordinates of the area

In Figure 10.6 the adjusted planes of test area 4 can be seen and generally it seems as the surface is even, which is caused by the high data quality. A few cells do not have an adjusted plane, which is caused by the cells having too few points when the plane adjustment was performed.

10.1.3 UAV cell centres

The estimated plane variables are used to calculate the elevation at the centre of each cell. This can be done by using eq. (9.1):

$$h_{centre} = \hat{a}e_{centre} + \hat{b}n_{centre} + \hat{c} \quad (10.1)$$

Where

$$\begin{aligned} h_{centre} &: \text{Elevation at cell centre} \\ e_{centre}, n_{centre} &: \text{e and n-coordinate of cell centre} \\ \hat{a}, \hat{b}, \hat{c} &: \text{Estimated plane variables} \end{aligned}$$

h_{centre} is a reduced coordinate and to get it back to the vertical datum DVR90 the mean of all H-coordinates in the test area have to be added:

$$H_{centre} = h_{centre} + \frac{\sum_{i=1}^m H_i}{m}$$

The calculated H_{centre} will be used when subtracting the two DEMs, which is described in Section 10.2, *Subtraction of DEMs*.

After the elevation at each cell centre has been calculated the standard deviation of the elevation will be estimated. This can be done by using the *general law of propagation of variances*: [Wolf and Ghilani, 1997, p. 84]

$$\sigma_H \approx \sqrt{\mathbf{A} \times \Sigma_{\hat{\mathbf{x}}} \times \mathbf{A}^T} \quad (10.2)$$

Where:

$$\begin{aligned} \sigma_H &: \text{Standard deviation of H-coordinate at cell centre} \\ \mathbf{A} &: \text{Design matrix which contain } [e_{centre} \ n_{centre} \ 1], \text{ see eq. (9.3)} \\ \Sigma_{\hat{\mathbf{x}}} &: \text{Covariance matrix for the estimated plane variables} \end{aligned}$$

$\Sigma_{\hat{\mathbf{x}}}$ is calculated by: [Wolf and Ghilani, 1997, p. 221] [Cederholm, 2000, pp. 39-44]

$$\Sigma_{\hat{\mathbf{x}}} = \sigma_0^2 \times N^{-1} \quad (10.3)$$

Where

$$\begin{aligned} \sigma_0^2 &: \text{Variance factor} \\ N^{-1} &: \text{The inverse of the matrix of normal equations, see eq. (9.4)} \end{aligned}$$

The standard deviation of the H-coordinate at the cell centres will later be used for calculating the standard deviation of the differences, which is described in Section 10.3, *Standard deviation of the difference*.

10.2 Subtraction of DEMs

After the H-coordinates of the cell centres, H_{centre} , has been calculated the elevation difference between the DK-DEMs and the UAV cell centres can be found. First the DK-DEMs have to be imported to MATLAB which stores them as matrices. The DK-DEMs can then be cropped to fit the test areas, which easily can be done as the lower left corner of the test areas are known. After this the DK-DEM and H_{centre} matrices can be subtracted from each other and to be systematic the DK-DEM will always be subtracted from the UAV cell centres:

$$H_{diff} = H_{centre} - H_{DK-DEM} \quad (10.4)$$

After the DK-DEM have been subtracted from the UAV cell centres new matrices with the elevation differences is created. These can be exported to ASCII files that can be read as rasters in ESRI ArcMap. [ESRI, 2017]

An example of the subtraction is seen in Figure 10.7, which shows the subtraction in test area 4 for both the DK-DEM/Terrain and the DK-DEM/Surface. Only test area 4 is shown as it is a good illustration of the difference between using the DK-DEM/Surface and the DK-DEM/Terrain.

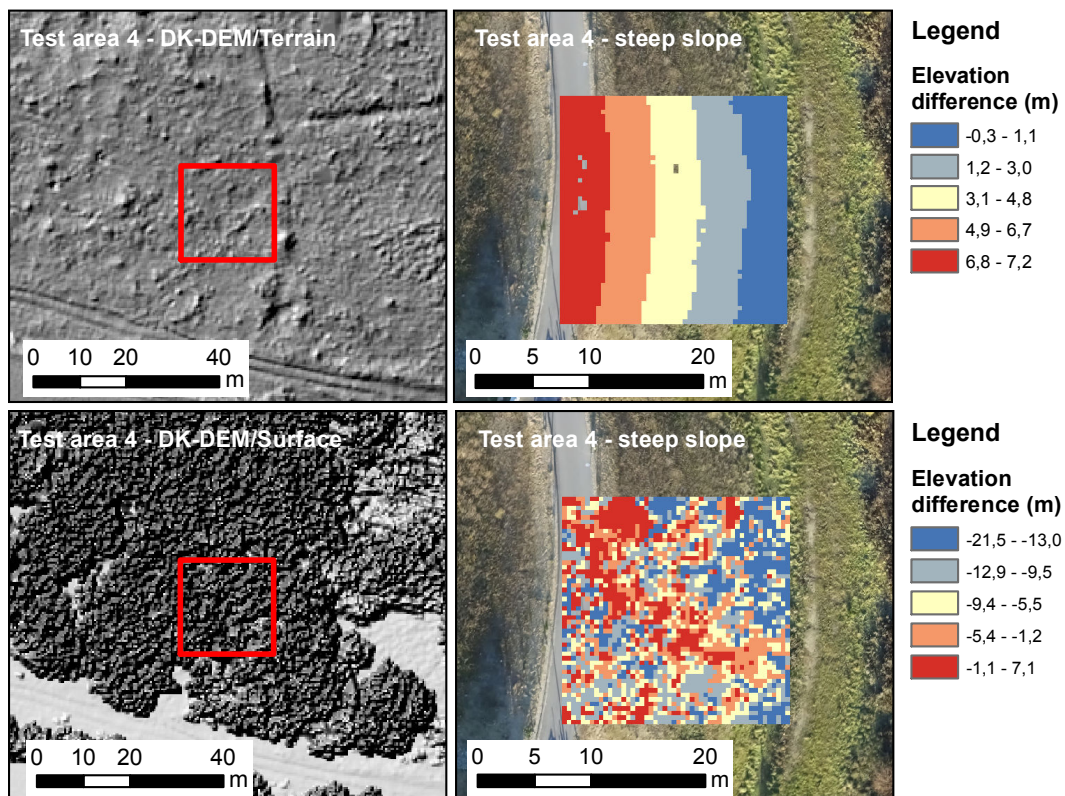


Figure 10.7: Subtraction of the DK-DEMs from the UAV data. The figures to the left show the hillshade models of the DK-DEM/Terrain and DK-DEM/Surface. The upper right figure is the DK-DEM/Terrain subtracted from the UAV data and the lower right is the DK-DEM/Surface subtracted from the UAV data. The background of the right figures is the UAV orthophoto. NB The scale of the legends are different

It can be seen from Figure 10.7 that the characteristics of the elevation differences is dissimilar between the usage of the DK-DEM/Terrain and the DK-DEM/Surface. The difference is an obvious result of the product used; the upper right figure is the difference between the previous and present terrain and the lower right figure is the difference between the previous surface (treetops) and present terrain. For this reason it is worth considering which DK-DEM product that is used, as the change detection is dependent on the elevation difference. This will be further discussed in Section 10.4, *Detect changes* and a choice will be made regarding which DK-DEM product that should be used.

10.3 Standard deviation of the difference

The elevation difference is a result of the subtraction of the two datasets. The standard deviation of the difference also need to reflect the accuracy of both datasets, why a contribution from each dataset is taken into account. The contribution from the different datasets is listed in Table 10.3, where it is specified if the standard deviation is global or on cell basis along with the source for the standard deviation.

Table 10.3: Standard deviation of the datasets

Dataset	Standard deviation	Source
DK-DEM/Surface	Global	Specification ($\sigma_{DK-DEM}=0.05$ m)
DK-DEM/Terrain	Global	Specification ($\sigma_{DK-DEM}=0.05$ m)
UAV	Cell-by-cell	σ_H
	Cell-by-cell	σ_0
	Global	$\sigma_{def}=0.05$ m

As mentioned in Section 6.3, *Change detection* the contribution from the DK-DEM raster models is based on the specifications from SDFE and is a global value. For the UAV dataset it was mentioned that a cell-by-cell standard deviation is preferable, which can be found based on the adjusted planes. In Section 10.1, *Adjustment of planes with remaining data* the standard deviation of the H-coordinate at the cell centres, σ_H , was expressed, which is one option for the contribution from the UAV dataset. Another option is the standard deviation of unit weight, σ_0 , which also is based on the adjustment. A standard deviation of the difference will be calculated with each of these contributions to see what difference it makes. Lastly σ_{def} is listed in Table 10.3, which is a global definition error. This error is listed as neither σ_H or σ_0 takes systematic errors and definition of the objects on the surface into account, as the adjustment presumes that the observations are independent and that the errors follow a normal distribution. The definition error will be further elaborated on in Section 10.3.1, *Approximation of the standard deviation of the difference*.

The standard deviation of the H-coordinate at the cell centres, σ_H , has been calculated and the size is evaluated in the following. σ_H is calculated for every cell in the four test areas and the average of the standard deviations for each test area is shown in Table 10.4.

Table 10.4: Average of the standard deviation, σ_H

Test area	Average of standard deviation
1 - low vegetation	0.016 m
2 - field	0.006 m
3 - highway road	0.021 m
4 - steep slope	0.007 m

As seen in Table 10.4 there is a large difference between test area 2 and 4, which have a lower σ_H , and test area 1 and 3. The same tendency was also seen when looking at the standard deviation of unit weight. For test area 2 and 4 σ_H is very low, also lower than what seems reasonable. For the adjustment of the planes it is presumed that the observations are independent. For a photogrammetric based point cloud the points are most likely correlated, but in which way is unknown. This conflict with the assumptions of the observations being independent, which is why σ_H can not be fully trusted. Nonetheless the standard deviation of the difference will be calculated with σ_H as a contribution. The choice of also testing σ_0 as a contribution is supported by this fact too, as σ_0 was larger and might represent the accuracy of the UAV dataset better.

10.3.1 Approximation of the standard deviation of the difference

The standard deviation of the difference, σ_{diff} , is calculated for every cell in the area, and differ for every cell based on the contributions listed in Table 10.3. σ_{diff} is found by the *special law of propagation of variances*, which is described in Section 6.3, *Change detection* and calculated with eq. (6.10).

To illustrate the size of σ_{diff} an approximation is calculated in the following with three different options of the contribution for the UAV dataset. For σ_H the values in Table 10.4 is used. For σ_0 the values in the third column in Table 10.2 is used, which is the average standard deviation of unit weight based on the L_2 -norm adjustment without outliers. Based on eq. (6.10), which is repeated below, σ_{diff} is found and showed in Table 10.5.

$$\sigma_{diff} = \sqrt{\sigma_{DK-DEM}^2 + \sigma_{UAV}^2} \quad (6.10)$$

The third option for the UAV error contribution contains the definition error, where an extra element is added to eq. (6.10) - see eq. (10.5).

$$\sigma_{diff} = \sqrt{\sigma_{DK-DEM}^2 + \sigma_0^2 + \sigma_{def}^2} \quad (10.5)$$

Table 10.5: Approximation of σ_{diff} for the four test areas with different contribution for the UAV dataset

σ_{diff}	Contribution from UAV dataset		
Test area	σ_H	σ_0	σ_0 and σ_{def}
1 - low vegetation	0.052 m	0.077 m	0.092 m
2 - field	0.050 m	0.055 m	0.074 m
3 - highway road	0.054 m	0.094 m	0.107 m
4 - steep slope	0.050 m	0.056 m	0.075 m

In Table 10.5 it is seen that σ_{diff} with σ_H contribution is around 0.05 m for all areas. This means that the contribution from σ_H drowns in the contribution from the DK-DEM. It is desired that σ_{diff} reflect the accuracy of both products, which is the case when the contribution is σ_0 . Especially for test area 1 and 3 σ_{diff} is effected by the inaccuracies of the UAV data, expressed with σ_0 . These values seems more representative of σ_{diff} , but the actual influence is seen when the change detection is carried out.

The definition error is introduced, because it is thought that by only using the error contributions based on the adjustment some incorrect detections may occur if the dataset have a low quality. σ_H and σ_0 is based on the adjustment, where it is assumed that the observations are independent and that the dataset only is effected by random normal distributed errors. This is not the case for a photogrammetric dataset, where systematic errors often are present. Additionally the definition of the ground or objects is not taken into account either.

The definition error can be based on different parameters. For example it can be based on a categorisation such that areas with different characteristics get different definition errors. It could also be based on the size of σ_0 for each cell, hence the quality of the data. Time do not permit a thorough investigation of how the definition error can be determined and for this reason a qualified guess for the value is chosen to be $\sigma_{def} = 0.05$ m. The project group is aware that the definition error is not a standard deviation, however it will treated so in the following tests.

In Table 10.5 the definition error, σ_{def} , is a contribution together with σ_0 . The standard deviation of the difference, σ_{diff} , is as expected larger for all four test areas, as σ_{def} has been added to every test area regardless of the quality of the data inside the area.

The three different options for σ_{diff} will be tested in the change detection and the actual influence of the choice of UAV error contribution will be illustrated.

10.4 Detect changes

After the standard deviation of the differences have been calculated the change detection can be performed. This is done by comparing the elevation difference at each cell with the standard deviation of the difference discussed in Section 10.3, *Standard deviation of the difference*. In MATLAB the following inequality is made:

$$|\mathbf{H}_{\text{diff}}| > \sigma_{\text{diff}} \cdot 3 \quad (10.6)$$

σ_{diff} is multiplied with three, based on statistical methods which were discussed in Section 9.3, *Remove outliers*. It is expected that if an absolute elevation difference is within this interval a change can not be determined, as the difference might be a result of the inaccuracies in the subtracted DEMs.

The inequality (10.6) will in MATLAB produce a binary matrix where the indices with “1” is a *change* and the indices with “0” indicates that *no change* has occurred. Cells with a NoData value will get a *no change* value, as very few or no points are within the cell.

A number of tests will be carried out to determine which parameters to use in a final change detection. For every test a change detection is performed where the chosen test parameters are varied.

The first test concern the use of either the DK-DEM/Surface or DK-DEM/Terrain for change detection. As mentioned in Section 10.3, *Standard deviation of the difference* three different options of σ_{diff} is calculated. A second test will compare σ_{diff} based on the standard deviation of the H-coordinate at the cell centres, σ_H , and σ_{diff} based on the standard deviation of unit weight for the adjusted planes, σ_0 . The third test compares when σ_{diff} is based on only σ_0 and when the global definition error, σ_{def} , is added.

The MATLAB script for change detection can be seen in Appendix H, *change_detection.m* and also be found in Appendix B, *Digital appendix*.

10.4.1 Test: Change detection with the DK-DEM products

The aim of the first test is to decide whether the DK-DEM/Surface or the DK-DEM/Terrain should be used for change detection. As it was mentioned in Section 10.2, *Subtraction of DEMs* the change detection is dependent on which DK-DEM product that is used, as the elevation differences gives different results.

The change detection has been conducted with the parameters ticked off in Table 10.6, where Figure 10.8 shows the change detection with the DK-DEM/Surface and Figure 10.9 shows the change detection with the DK-DEM/Terrain.

It is seen in Figure 10.8 and 10.9 that the change detections give different results, especially at areas which previously was not only terrain for example test area 1 and test area 4. Both of these test areas were previously covered with high vegetation, which explains the appearance of the change detection seen in Figure 10.8 for test area 1 and 4. This is because the surface has changed, as the vegetation at these areas have been removed. For this reason it is correct

Table 10.6: Parameters used for change detection when testing the two DK-DEM products

Data			
DK-DEM/Surface		×	Test parameter
DK-DEM/Terrain		×	Test parameter
UAV point cloud		×	
Error contributions for σ_{diff}			
DK-DEM	σ_{DK-DEM}	×	
UAV	σ_H	×	
UAV	σ_0		
UAV	σ_{def}		

that the change detection deem most cells as being a *change*. For test area 4 the few *no change* cells are probably caused by the slope being at the same elevation as some of the high vegetation in the DK-DEM/Surface. For test area 4 this creates a “salt and pepper” noise, which can be removed by filtering the result of the change detection. This will be discussed in Section 10.5, *Filter change detected cells*.

When comparing Figure 10.8 with Figure 10.9 it can be seen that test area 1 and 4 have the most distinct differences, which as mentioned is caused by the areas previously having high vegetation. Test area 2 and test area 3 were previously terrain surfaces, which explains why the figures are quite similar at these test areas. Test area 2 is a field which may explain why some of the cells get the *change* value, especially close to what looks like tractor tracks, as some of the soil may have been relocated. In test area 3 it can be seen that the low data quality affects the change detection and causes some of the cells on the road to be detected as a *change*, which is an error. Furthermore it can be seen that the guardrail is detected as a change when using both DK-DEM products. This is not expected as it is thought that by using the DK-DEM/Surface more cells with the *no change* value would be present. It is uncertain why this is the case.

At test area 1 in Figure 10.9 cells with a *no change* value is caused by what is thought to be actual terrain surfaces scattered throughout the low vegetated area. At test area 4 in Figure 10.9 the bottom of the slope has cells with a no change value, which makes sense as only the sloped terrain is new. Test area 4 is also a good example of the difference between using the DK-DEM/Surface and DK-DEM/Terrain for change detection; when using the DK-DEM/Surface almost every cell get a *change* value as the surface have changed, when using the DK-DEM/Terrain only the changed terrain will get a *change* value.

As SDFE mostly are interested in updating the DK-DEM/Terrain, the use of this model for making the change detection is preferable. The DK-DEM/Surface can mostly be used if the aim is to detect changes of the surface.

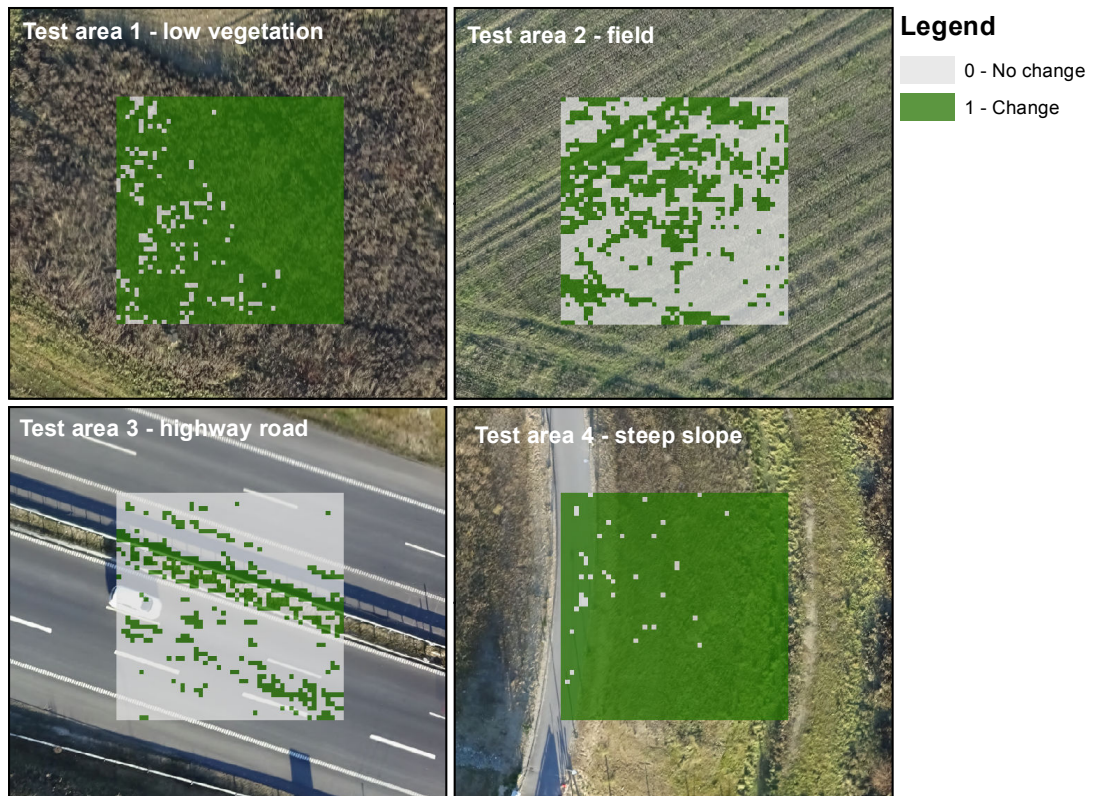


Figure 10.8: Change detection based on the DK-DEM/Surface and the σ_H error contribution. The background in the figures is the UAV orthophoto

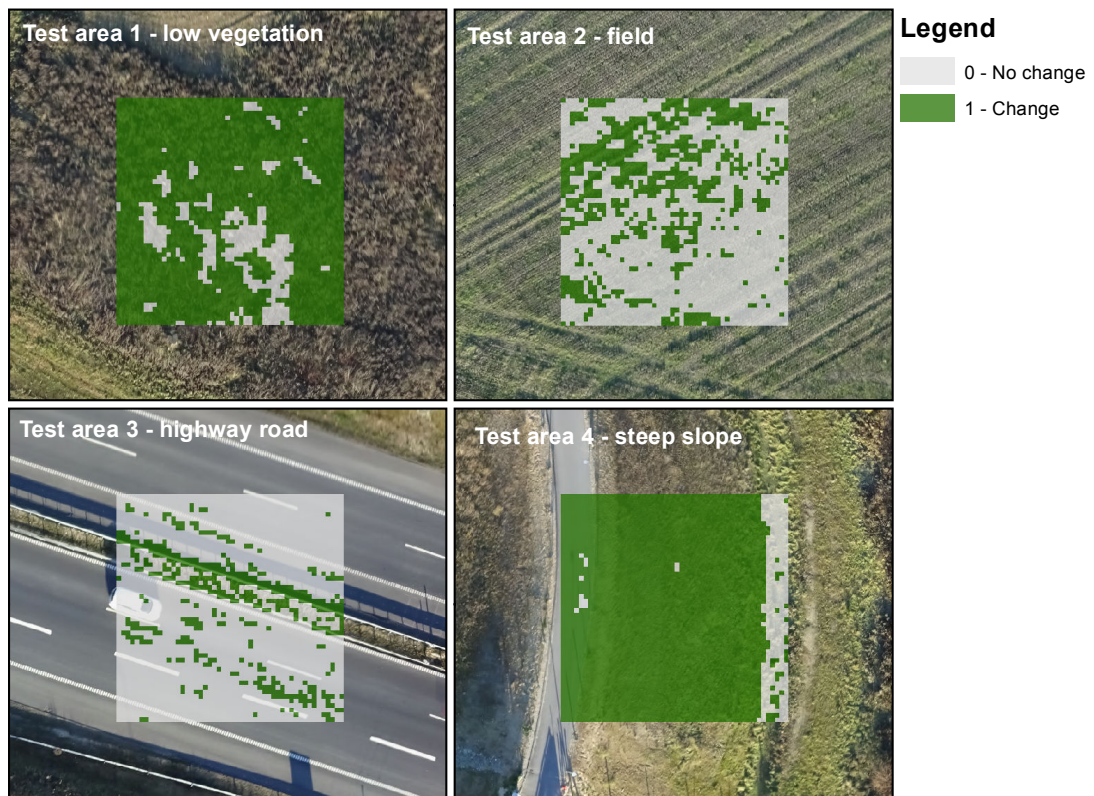


Figure 10.9: Change detection based on the DK-DEM/Terrain and the σ_H error contribution. The background in the figures is the UAV orthophoto

Change detection with both DK-DEM products

Previously it is only suggested to use one of the DK-DEM products to detect changes, but it is also a possibility to use both DK-DEM products. This could be a step on the way to remove the effect of non-terrain objects in the change detection. In this project it could be useful as a categorisation has not been performed.

The approach is to see whether a change has occurred based on both the DK-DEM/Terrain and DK-DEM/Surface. Only if this is the case the change is deemed correct. A situation where this could be useful is illustrated in Figure 10.10.

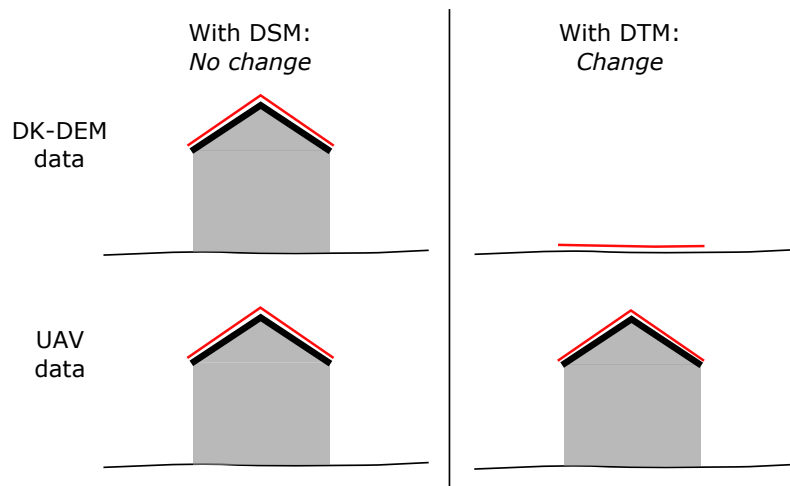


Figure 10.10: Situation where the change detection would benefit from being based on both the DK-DEM/Surface and DK-DEM/Terrain

In the illustrated situation a house, which both existed when the DK-DEM data and the UAV data was collected, is showed. The DK-DEM/Surface represents the roof and in the DK-DEM/Terrain the house is removed. The UAV data shows the surface, hence the roof is represented in this dataset. When a “change detection” of the situations above and below each other is made, the first situation with the DK-DEM/Surface results in a *no change*, which is correct. In the second situation with the DK-DEM/Terrain a *change* is detected. This is also correct, but it is not a change of the terrain, which is the main concern.

In a case where both the DK-DEM/Surface and DK-DEM/Terrain are used for the change detection, the result would be a *no change*, as they both need to have a *change* value before it is deemed an actual change. The benefit of the approach is that a change detection only based on the DK-DEM/Terrain would result in a *change*, which is incorrect.

The approach still have issues regarding situations similar to test area 1 with low vegetation. Both the DK-DEM/Surface and DK-DEM/Terrain will show a *change*, which then will lead to a *change*, even though the terrain actually not has changed.

The use of both DK-DEM products for change detection will not be implemented, as it will not solve issues regarding vegetation. It could be implemented as part of a final algorithm where a categorisation also would be performed. For this reason only the DK-DEM/Terrain will be used for the change detection and the following tests will be performed with this product.

10.4.2 Test: Change detection with σ_H or σ_0 as error contributions

The aim of the test is to see which influence on the change detection it has to use either σ_H or σ_0 as the UAV error contribution for σ_{diff} . As it was seen in Section 10.3, *Standard deviation of the difference* the size of σ_{diff} , and thereby the interval for detecting changes, differ when using the two different contributions, and therefore it will effect the results.

The change detection has been conducted with the parameters ticked off in Table 10.9, where Figure 10.11 shows the change detection with the σ_H error contribution and Figure 10.12 shows the change detection with the σ_0 error contribution.

Table 10.7: Parameters used for change detection when testing σ_H or σ_0 as UAV error contributions

Data			
DK-DEM/Surface			
DK-DEM/Terrain			
UAV point cloud			
Error contributions for σ_{diff}			
DK-DEM	σ_{DK-DEM}	×	Test parameter
UAV	σ_H	×	
UAV	σ_0	×	
UAV	σ_{def}		

When comparing test area 1 in Figure 10.11 and 10.12 more cells get a *no change* value, but the difference seem to be quite small. In areas such as test area 1 a categorisation, which removes vegetation etc., may be the only way to ensure that actual terrain get a correct evaluation of whether it is a change or not.

When comparing test area 3 in Figure 10.11 and 10.12 the thought is that by using σ_0 the low data quality of test area 3 will have a smaller impact on the change detection. As seen a small improvement is made. Fewer cells get a *change* value, but some cells still get a *change* value because of the low data quality. Test area 2 and 4 are almost the same when comparing the two figures, which is expected because the data quality is higher.

As especially test area 3 still is affected by the low data quality, the use of σ_0 as an error contribution may not entirely be enough to express the quality of the UAV data. For this reason the definition error, σ_{def} is introduced, where the impact is tested in the following section.

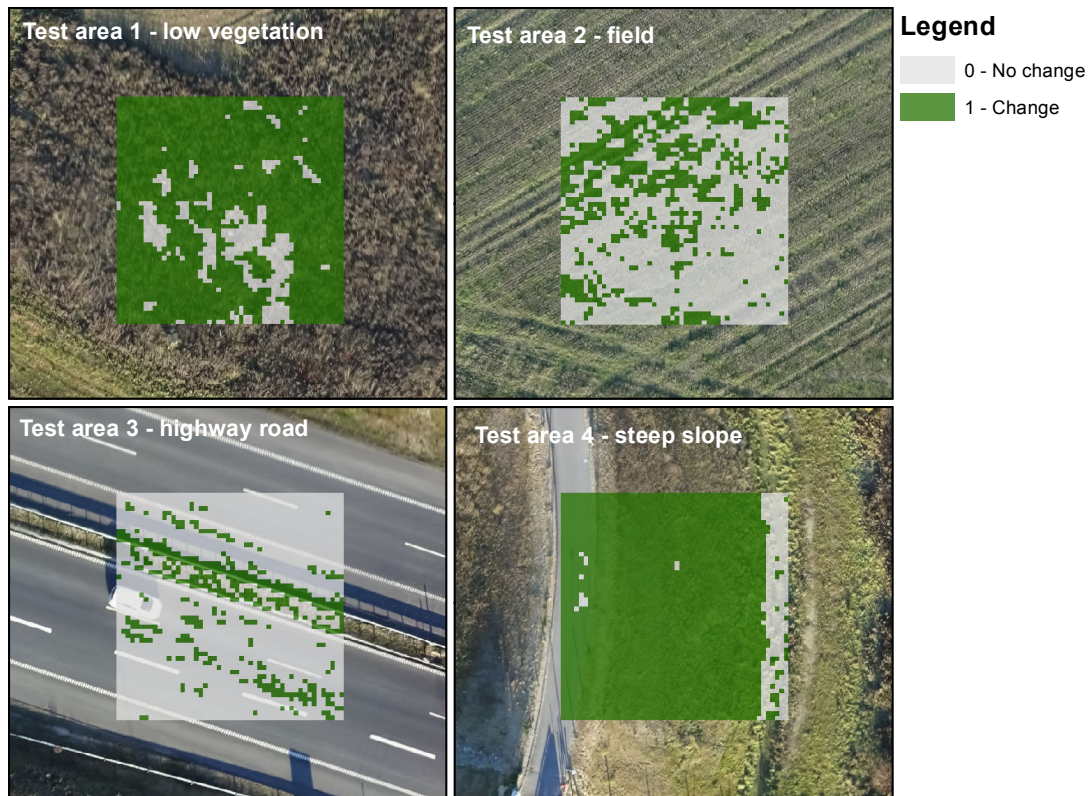


Figure 10.11: Change detection based on the DK-DEM/Terrain and the σ_H error contribution. The background in the figures is the UAV orthophoto

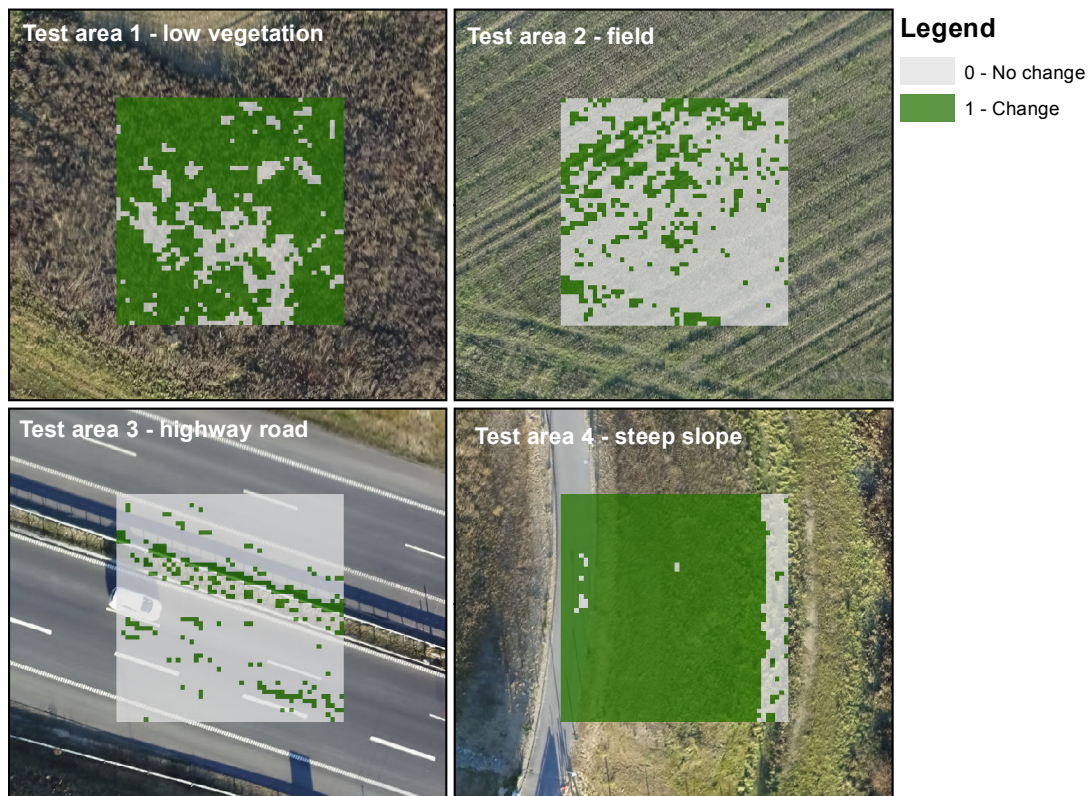


Figure 10.12: Change detection based on the DK-DEM/Terrain and the σ_0 error contribution. The background in the figures is the UAV orthophoto

10.4.3 Test: Change detection with the σ_0 and σ_{def} error contributions

The aim of the test is to see if an added definition error can solve the issues regarding low data quality. By using both σ_0 and a definition error, σ_{def} , as the error contribution for the UAV dataset, the interval for detecting changes gets larger, which effects the result of the change detection.

The change detection has been conducted with the parameters ticked off in Table 10.8, where Figure 10.13 shows the change detection with σ_0 as the error contribution and Figure 10.14 shows the change detection with both σ_0 and σ_{def} as the error contribution for the UAV data.

Table 10.8: Parameters used for change detection when testing σ_0 and σ_{def} as UAV error contributions

Data			
DK-DEM/Surface			
DK-DEM/Terrain		×	
UAV point cloud		×	
Error contributions for σ_{diff}			
DK-DEM	σ_{DK-DEM}	×	
UAV	σ_H		
UAV	σ_0	×	
UAV	σ_{def}	×	Test parameter

When comparing Figure 10.13 and Figure 10.14 it is seen that test area 1 have not changed considerably, which is expected as the UAV data represent low vegetation, hence the changes are actually changes and not just caused by low data quality. It is seen that test area 2 is different, which is because the quality of the data in this area is high and the elevation differences is low. This means that a definition error of 0.05 m may be too large in such an area. Test area 3 benefit from the additional definition error as many of the cells which in Figure 10.13 is a *change* now is a *no change* in Figure 10.14. The few remaining inaccurate cells can possibly get a *no change* value by using a filter, this is tested in the following section. Test area 4 do not benefit much from using the definition error.

Based on the discussion above, a global definition error is perhaps not the best approach. Currently this is the only option, which is why the chosen definition error, σ_{def} , together with σ_0 will be used for the change detection.

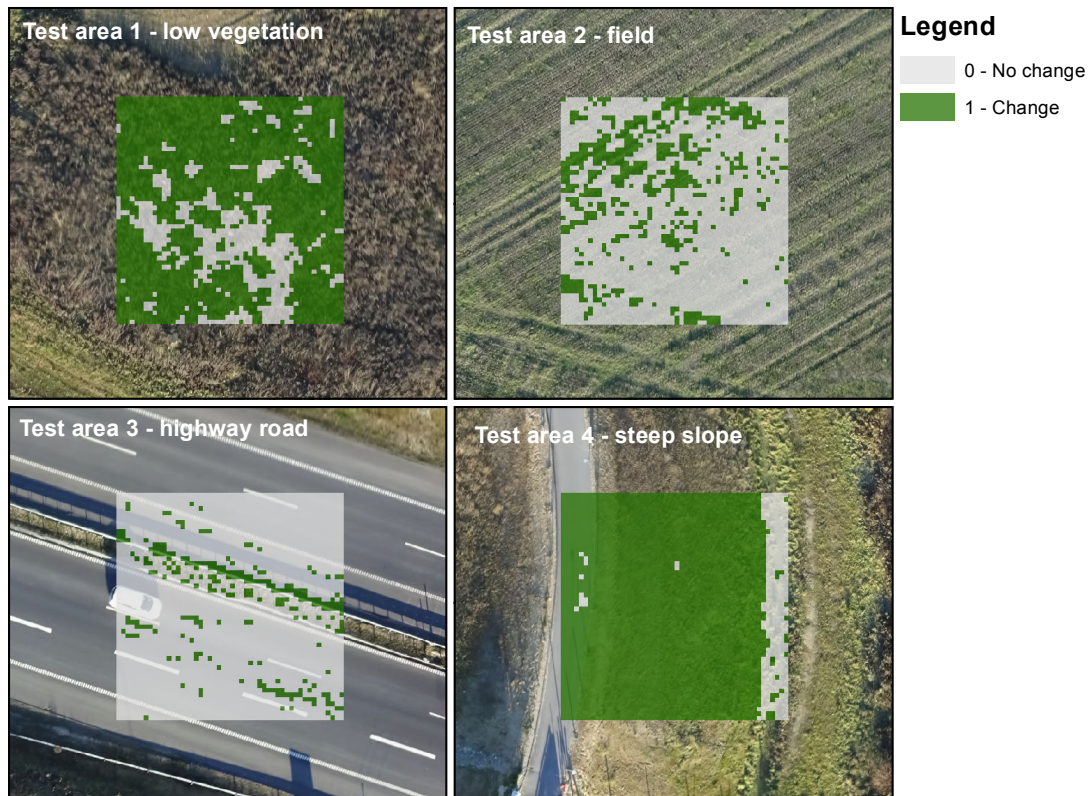


Figure 10.13: Change detection based on the DK-DEM/Terrain and the σ_0 error contribution. The background in the figures is the UAV orthophoto

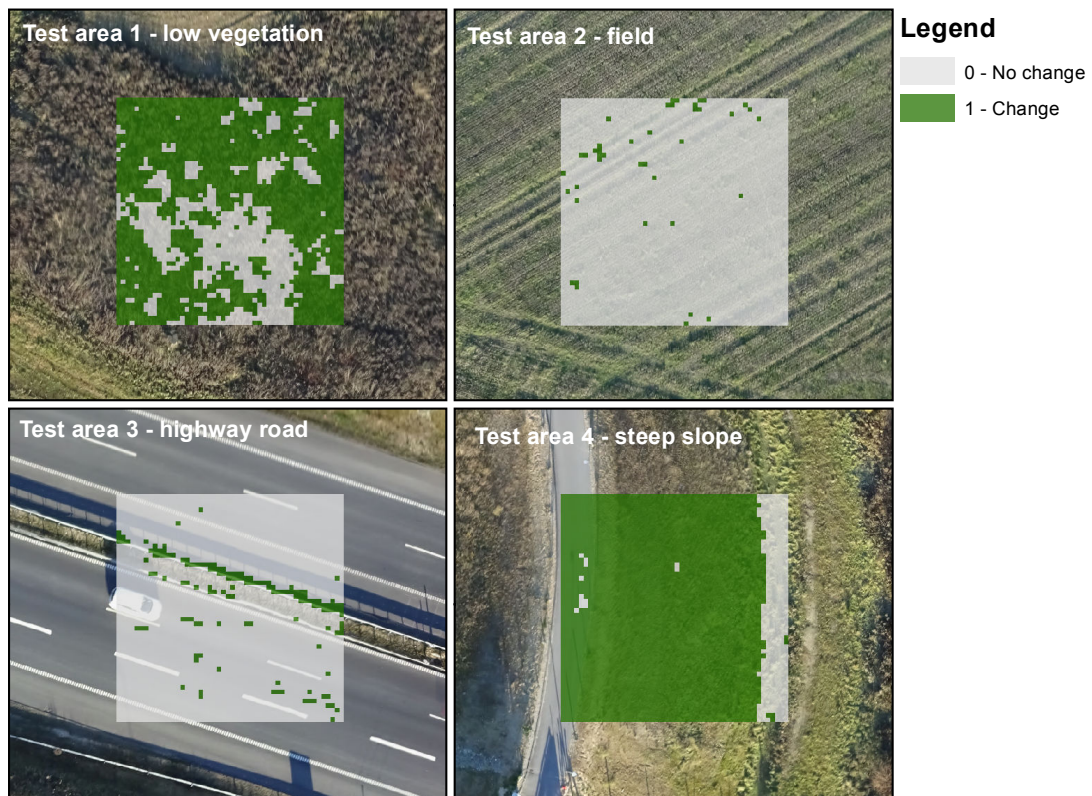


Figure 10.14: Change detection based on the DK-DEM/Terrain with σ_0 and σ_{def} as error contributions. The background in the figures is the UAV orthophoto

10.5 Filter change detected cells

In the change detection it was seen that the test areas, some more than others, were effected by “salt and pepper” noise. This noise can be removed by filtering methods, which are well known from image processing. Several filtering methods exist, but it is chosen to use the median filter, which will be described in the following. The purpose of the median filter is to remove the “salt and pepper” noise, but without distorting the changes that has been detected.

The MATLAB script for the median filter can be seen in Appendix I, *median_filter.m* and also be found in Appendix B, *Digital appendix*.

To check if a cell is “noise”, the neighbours to the cell is analysed. The window defining the neighbourhood can have any size and shape, but a quadratic window is often chosen, ($n \times n$), with n being an odd number. An odd number result in the checked cell being in the centre of the window. [Acharya and Ray, 2005, pp. 114-115] In Figure 10.15 an example of the median filter is illustrated. The values inside the window are sorted into numerical order and the median (middle) value is chosen as the new cell value. This is continued for all cells throughout the raster.

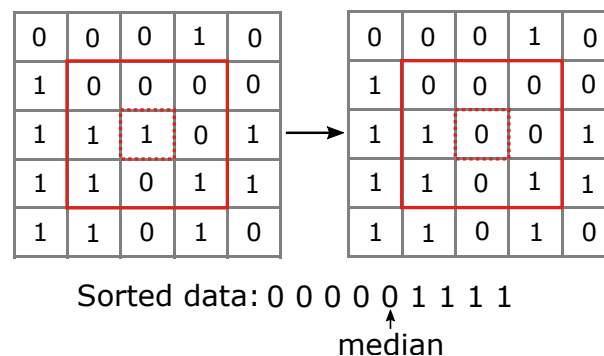


Figure 10.15: Illustration of the median filter for one cell (red dotted line). The window is outlined by the red line and the sorted data and median is shown

10.5.1 Test: Median filter

The aim of the test is to see how the median filter with two different search windows perform. It is chosen to test the median filter on the final change detection.

The change detection has been conducted with the parameters ticked off in Table 10.8, where Figure 10.16 shows the change detection after the 3×3 median filter and Figure 10.17 shows the change detection after the 5×5 median filter. The figures are compared to Figure 10.14, which is the same change detection but without the median filter.

The edges of the test areas may be distorted as the filter pads the edges with zeroes. [MathWorks, 2017] The distortion is seen as insignificant when a large area is used.

When comparing Figure 10.16 and Figure 10.17 it shows that by using a 5×5 window the filtering is more aggressive. For test area 3 the changes around the guardrail has been remove with the 5×5 window. As the guardrail in the UAV data actually is a change from

Table 10.9: Test

Data			
DK-DEM/Surface			
DK-DEM/Terrain		×	
UAV point cloud		×	
Error contributions for σ_{diff}			
DK-DEM	σ_{DK-DEM}	×	
UAV	σ_H		
UAV	σ_0	×	
UAV	σ_{def}	×	
Median filter			
Search window 3×3		×	Test parameter
Search window 5×5		×	Test parameter

the DK-DEM/Terrain, the change should not be removed by the median filter, hence it is too aggressive. In Figure 10.16 the “salt and pepper” noise on the road in test area 3 is removed, but the changes around the guardrail is maintained. For all test areas the median filter with 3×3 window manages to remove the “salt and pepper” noise, why there is no need to use the 5×5 window. In Chapter 11, *Changes in Odense SE* the result of the change detection will therefore be filtered with the median filter with 3×3 window.

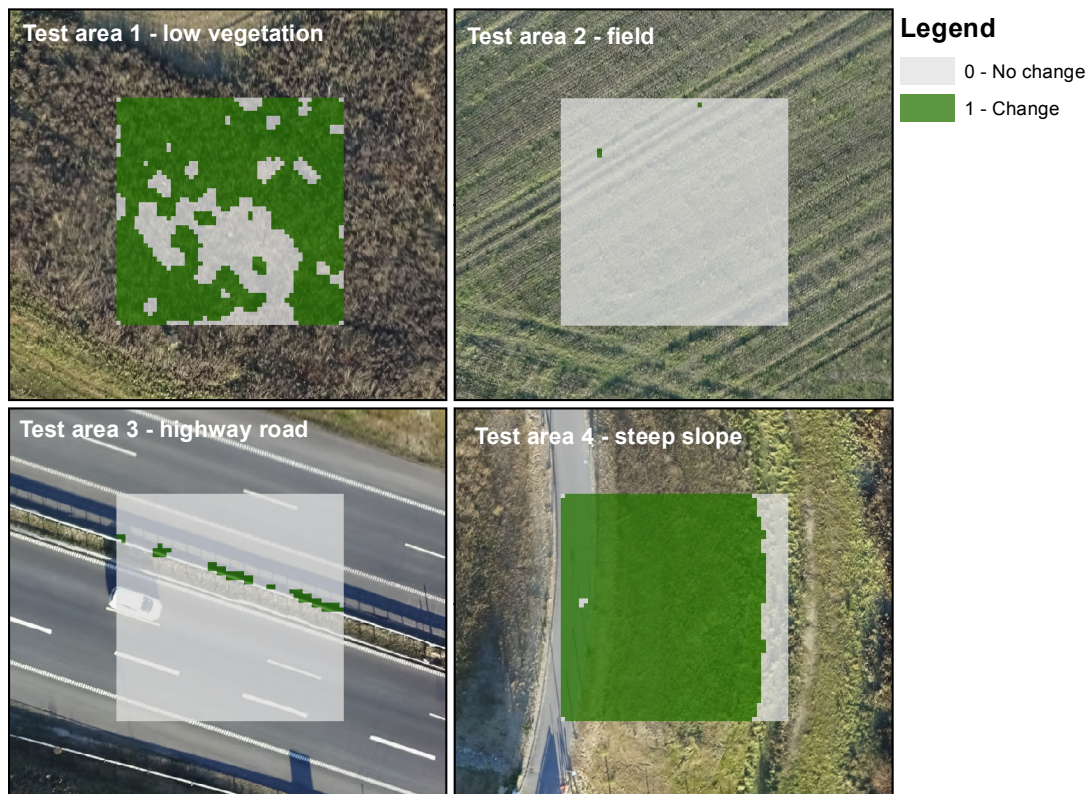


Figure 10.16: Median filter with 3×3 window of the change detection. The background in the figures is the UAV orthophoto

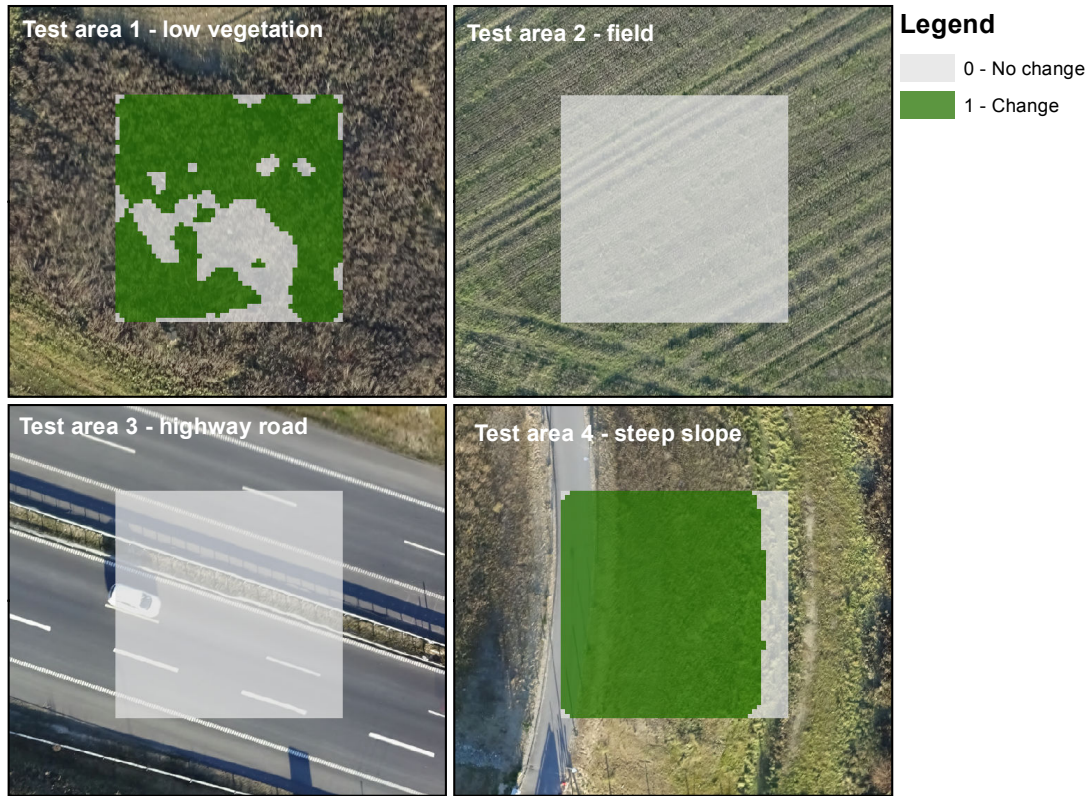


Figure 10.17: Median filter with 5×5 window of the change detection. The background in the figures is the UAV orthophoto

10.6 Summary

Through the present chapter the implementation of the methods for change detection has been described and the results has been displayed. A number of decision was still to be made, concerning the DK-DEM product and the contributions for the standard deviation of the difference.

Regarding the DK-DEM product it was decided to use the DK-DEM/Terrain, as the main interest is to update this model. By comparing the UAV data with this model it is detected if a change of the terrain has occurred, as long as the UAV data represent terrain. If the UAV data does not represent terrain, as test area 1 for example, the change detection is incorrect.

To evaluate the elevation differences the standard deviation of the difference was set up. It was chosen that the error contribution for the UAV data is the standard deviation of unit weight, σ_0 , and not the standard deviation of H-coordinate at the cell centre, σ_H . By making this choice a larger σ_{diff} is created, hence a larger interval where the elevation difference is not a change. Additionally a definition error was added, as it was seen that some cells still were incorrect after the change detection, but also because σ_0 does not incorporate systematic and definition errors. The definition error, σ_{def} , is set to a fixed value, 0.05 m, even though considerations about a varying value was made.

The final change detection was filtered to remove “salt and pepper” noise. This was carried out with a median filter, where a search window of 3×3 cell was chosen.

In Table 10.10 the final choice of parameters for the change detection is ticked off:

Table 10.10: Final choice of parameters

Data		
<hr/>		
DK-DEM/Surface		
DK-DEM/Terrain		×
UAV point cloud		×
<hr/>		
Error contributions for σ_{diff}		
<hr/>		
DK-DEM	σ_{DK-DEM}	×
UAV	σ_H	
UAV	σ_0	×
UAV	σ_{def}	×
<hr/>		
Median filter		
<hr/>		
Search window 3×3		×
Search window 5×5		
<hr/>		

The change detection shows good results, but as expected there are issues regarding areas with vegetation and presumably also other objects above terrain. To address these issues it is necessary with a different approach, for example a categorisation.

For areas with terrain the algorithm is capable of detecting changes only with the use of the UAV data itself and the DK-DEM/Terrain.

The following chapter will carry out the change detection for an area of 1×1 km within the case area Odense SE. The change detection will be made based on the methods and choices described in the previous chapters.

Changes in Odense SE 11

In this chapter the chosen parameters from Chapter 10, *Change detection* will be used to perform change detection on a larger area of Odense SE. This will be done in order to see how the change detection perform when an area with the size of an intended assignment is used. The area is chosen to be 1×1 km such that the most distinct differences will be present. The extent of the chosen area can be seen in Figure 11.1.

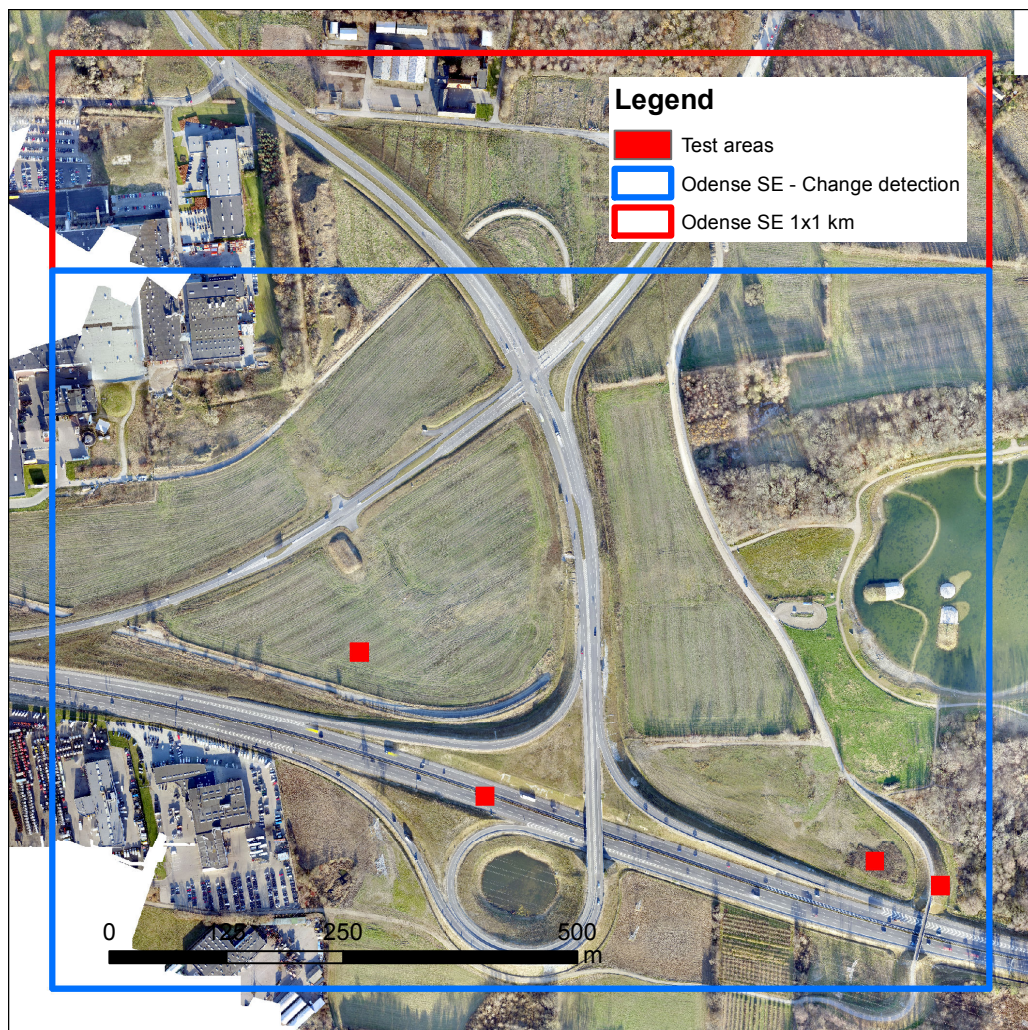


Figure 11.1: Extent of the chosen area. The red polygon shows the chosen area and the blue polygon show the area where the actual change detection is performed. The red squares are the four test areas. The background in the figure is the UAV orthophoto

As it can be seen in Figure 11.1 only the area which is covered by the blue polygon is detected for changes. This is because the script “*data_to_cells.m*”, which divide the data into to cells, is not optimised to handle a large dataset. Because of this the script was terminated before it finished, which resulted in that only the data inside the blue polygon was divided into cells. As optimising the scripts have not been prioritised the result from dividing the data into cells is acceptable, as most of the chosen area can be detected for changes. The result of the change detection with the chosen parameters can be seen in Figure 11.2



Figure 11.2: Change detection of the case area Odense SE. The background in the figures is the UAV orthophoto

In Figure 11.2 the GeoDanmark forest and building polygons have been added in order to illustrate where it is expected that no changes have occurred. The polygons can, as mentioned, be used to perform a coarse categorisation of the UAV data, which Figure 11.2 also is intended to illustrate.

As it can be seen in Figure 11.2, the algorithm successfully detect changes. Especially areas close to the new highway exit and the pond is seen as changes, which is expected.

Overall the result is deemed satisfactory, but issues with non-terrain objects still exists. In Figure 11.3 a good and bad example of the change detection is illustrated.

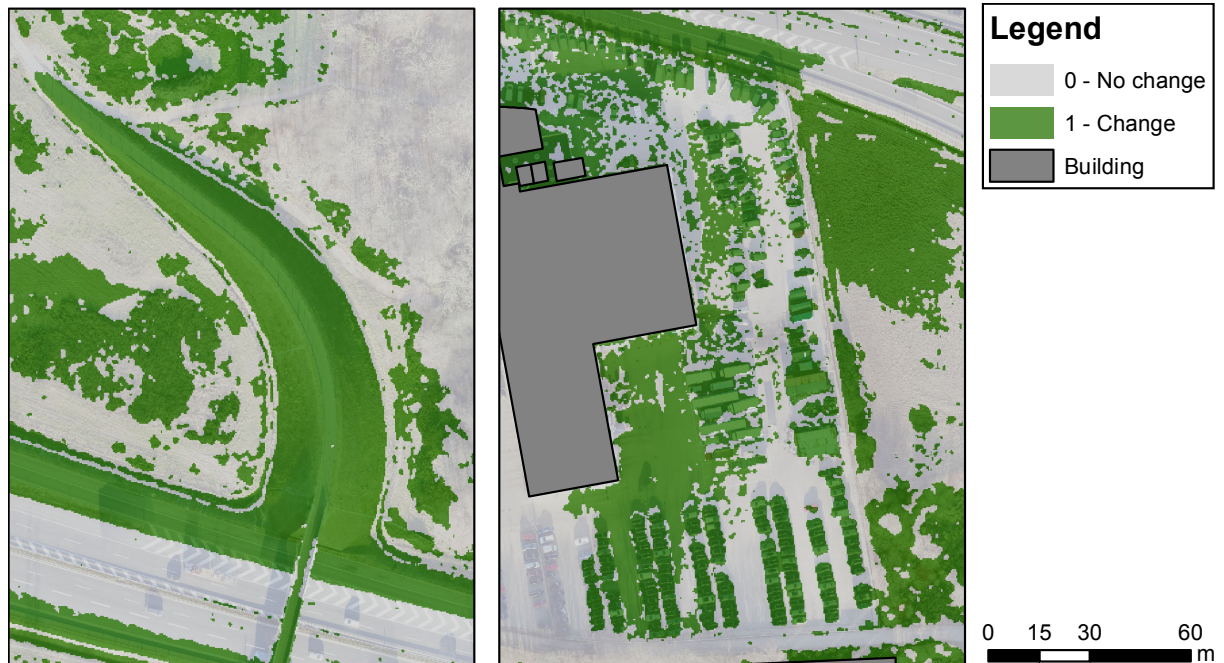


Figure 11.3: The figure to the left shows the new walkway bridge, which is seen as a good example of change detection. The figure to the right shows a parking lot close to a building, this is seen as a bad example of change detection

The left figure of Figure 11.3 represents the change detection that has occurred at the new walkway bridge. The change detection seen on the left figure is regarded as a successful example of change detection. This is because the change detection behave as expected and detect the walkway bridge as a changed. There are still some smaller areas with *change* value, which are not caused by changes of the terrain. After inspection it is seen that most of these areas cover vegetation, where well documented issues exist.

The right figure of Figure 11.3 illustrates the change detection which has occurred at a parking lot close to a large building. All cells with a *change* value on the parking lot are seen as being incorrect. This is concluded after inspecting the DK-DEM/Terrain and UAV data. The parking lot has similar quality issues as the highway road, as the surface is very homogeneous. Additionally non-terrain objects, such as parked cars and containers, also account for many of the cells which have a *change* value. It is uncertain how issues regarding incorrect change detection can be solved, but a manual quality control could potentially be performed by coarsely discarding cells which incorrectly are detected as a change. Doing a manual quality control is time consuming, but as the algorithm do not include a method for categorising the UAV data, it is thought to be an option for solving issues regarding non-terrain objects.

Conclusion 12

It has been achieved to detect changes within the case area Odense SE and the following will summarise the final part of the project and answer the problem statement;

How can the chosen methods, for quality assessment and change detection of terrain, be implemented for updating the Danish Elevation Model?

The quality assessment was carried out by performing a least squares adjustment with the $L_{1.3}$ -norm, which quantified the quality of the UAV data. This showed that the relative precision is between 0.03 m and 0.10 m depending on the area as it was seen in Table 9.1. The lower precision was seen in areas with poor point matching or vegetation, where as areas with a well defined surface, for example fields, had a higher precision. As it was expected that the dataset contain outliers a robust adjustment with the $L_{1.3}$ -norm was carried out, where outliers were given a lower weight. Based on this adjustment outliers were removed, which resulted in a new dataset which was used onwards.

The change detection was implemented by calculating the elevation differences between the two models for every cell and evaluate each difference individually. The evaluation of the difference for a cell was based on the accuracy of the data inside the same cell. The accuracy of the elevation difference was expressed with the standard deviation of the difference. Following different tests it was chosen that the standard deviation of the difference should be a product of the accuracy of the DK-DEM product, the precision of the UAV data found by the adjustment and an additional definition error to account for systematic errors and definition of the surface. Based on an interval, set up from the standard deviation of the difference, every elevation difference was determined to be a change or not. The change detection showed good results for areas with terrain. For the area with low vegetation the change detection was incorrect, but for the highway road, which showed to have a low data quality, it succeeded to get a correct change detection. For areas with terrain and a good data quality the change detection was also successful. As the areas where a change has occurred can be found, it is possible to update the Danish Elevation Model with UAV data for these areas.

A change detection was carried out for a larger area of the case area Odense SE, which showed similar results as the test areas and generally the algorithm behaved as expected.

All together a satisfying result has been obtained, where the developed algorithm is capable of detecting changes in areas with terrain. The change detecting can be carried out only with the use of the UAV data itself, which is desirable.

A remaining obstacle concern non-terrain objects, where the change detection possibly is incorrect. To solve this a categorisation is needed. This will enable the user to determine which part of the change detection that may be incorrect. As a categorisation have not been performed the proposed change detection is only seen as a step on the way to develop a final algorithm for updating the Danish Elevation Model.

Discussion 13

During the project several analyses of methods and data was not possible to conduct. For this reason this discussion will describe which analyses that further could be performed and what they encompass.

The project primarily concern change detection, which means that issues regarding non-terrain objects have not been solved. For this reason it is thought that an analysis of feature categorisation in UAV data is essential before an algorithm for updating the Danish Elevation Model with UAV data can be finalised. In Chapter 6, *Methods for the updating process* it was investigated which methods and data that can be used for the algorithm. Three methods for categorisation was described; vector analysis, point cloud filtering and image classification. Further investigations on vector analysis seem limited and the use of point cloud filtering was questioned when the method was described, but still not eliminated. A potential is seen with image classification, which is why additional ideas is unfolded. Using an image classification will not necessarily show the best result of a categorisation, as point cloud filtering or vector analysis may also be an good way to classify features in the UAV data. The following topics could be researched during an analysis of categorisation methods:

Point cloud filtering methods: Many point cloud filtering methods exist and the subject was only briefly discussed. For this reason a more thorough investigation of point cloud filtering methods could be interesting to perform. Many methods are made to handle LiDAR point clouds and for this reason it is expected that by using them on UAV data the result could be poorer. However, it is intriguing to see how some of the methods would perform and maybe adjust the methods to handle UAV data better.

Unsupervised vs. supervised image classification: One of the ideas in the project was to investigate how the unsupervised and supervised classification methods perform, with the UAV orthophoto, compared to each other. In order to do this individual test of the methods may need to be performed before they can be compared. Another idea was to use the GeoDanmark orthophoto, as it contain the NIR band which could make the categorisation more accurate. The categorisation with the orthophoto from the UAV and GeoDanmark could be compared and determine which data source that should be used.

Unsupervised and supervised image classification with additional data: During the investigation of the unsupervised and supervised image classification an idea about using elevation data in the classification emerged. For elevation models which are stored as raster it would be possible to use the elevation data when performing the image classification. Using elevation data in the image classification adds an extra coordinate dimension, which could lead to some interesting results, as the categorisation in addition to RGB values would consider the elevations. Vector data could potentially also be rasterised and used in the classification.

Besides analyses on categorisation methods other analyses could be performed. This could for example be an investigation of using different cell sizes for the plane adjustments when removing outliers. Furthermore a polynomial could be fitted to a larger cell instead of a plane, which potentially could give a better representation of the point cloud. This would be interesting as non-terrain points could be removed by this approach and act as a form of point cloud filtering, which keep terrain points.

With the present structure of the algorithm a manual quality control is essential and it is the only way to avoid issues regarding non-terrain objects. Even if a categorisation would be performed it is still thought that a manual quality control would be needed, as it is experienced that the UAV data in general have a varying data quality. The analyses of a manual quality control could for example investigate which approaches that minimise the time consumption or processes that could help during the quality control.

Bibliography

- Acharya and Ray, 2005.** Tinku Acharya and Ajoy K. Ray. *Image Processing - Principles and Applications*. John Wiley & Sons, Inc., 2005.
- ASPRS, 2011.** ASPRS. *LAS SPECIFICATION VERSION 1.4*, 2011. URL http://www.asprs.org/wp-content/uploads/2010/12/LAS_1_4_r13.pdf. seen 10-04-17.
- Balstrøm, Jacobi, and Bodum, 2010.** Thomas Balstrøm, Ole Jacobi, and Lars Bodum. *Bogen om GIS og geodata*. Volume 1, Issue 2. Forlaget GIS og Geodata, 2010.
- Briese, Pfeifer, and Dorninger, 2002.** Ch. Briese, N. Pfeifer, and P. Dorninger. *Applications of the robust interpolation for DTM determination*, Institute of Photogrammetry and Remote Sensing, Vienna University of Technology, 2002.
- Bækhoj, 2016.** Martin Bækhoj. *Aalborg Kommune er flyvende: Droner skal lette arbejdsbyrden*, 2016. URL <http://www.tv2nord.dk/artikel/aalborg-kommune-er-flyvende-droner-skal-lette-arbejdsbyrden>. seen 24-02-17.
- Cederholm, 2000.** Peter Cederholm. *Udjævning*. 2. edition, 1. revision. Aalborg Universitet - Institut for Planlægning, 2000.
- Cederholm, 2016a.** Peter Cederholm. *Statistical methods in surveying and mapping, lecture 11 - Least Squares with equality constraints*, Aalborg University, 2016a.
- Cederholm, 2016b.** Peter Cederholm. *Statistical methods in surveying and mapping, lecture 8 - Robust adjustment 1*, Aalborg University, 2016b.
- Chen, Gao, and Devereux, 01-14-2017.** Ziyue Chen, Bingbo Gao, and Bernard Devereux. *State-of-the-Art: DTM Generation Using Airborne LIDAR Data*, 01-14-2017. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5298723/pdf/sensors-17-00150.pdf>. seen 03-04-17.
- Danish Transport, Construction and Housing Authority, 2017.** Danish Transport, Construction and Housing Authority. *Bekendtgørelse om flyvning med droner i bymæssigt område*, 2017. URL <https://www.retsinformation.dk/Forms/R0710.aspx?id=183645#id78c32448-3839-448a-8047-2e559f430100>. seen 03-04-17.

- Dilmen, 2012.** Nevit Dilmen. *Composition of RGB from 3 Grayscale images*, 2012. URL https://en.wikipedia.org/wiki/Grayscale#/media/File:Beyoglu_4671_tricolor.png. seen 27-04-17.
- El-Ashmawy, 01-14-2017.** Khalid L.A. El-Ashmawy. *State-of-the-Art: DTM Generation Using Airborne LIDAR Data*, 01-14-2017. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5298723/pdf/sensors-17-00150.pdf>. seen 03-04-17.
- Eriksen, Tvedebrink, and Højbjerg, 2015.** Poul Svante Eriksen, Torben Tvedebrink, and Malene Højbjerg. *Lecture slides content from SM-course, lecture 1*, Aalborg University, 2015.
- Esri, 2017.** Esri. *Raster to Polygon*, 2017. URL <http://pro.arcgis.com/en/pro-app/tool-reference/conversion/raster-to-polygon.htm>. seen 27-04-17.
- ESRI, 2017.** ESRI. *ASCII to Raster*, 2017. URL <http://pro.arcgis.com/en/pro-app/tool-reference/conversion/ascii-to-raster.htm>. seen 22-05-17.
- Finansministeriet, 2012.** Finansministeriet. *Gode grunddata til alle - en kilde til vækst og effektivisering*, 2012. URL https://www.fm.dk/~media/publikationer/imported/2012/gode-grunddata-til-alle/web_grunddatatilalle_vaekstogeffektivisering_okt-2012.ashx. seen 20-02-17.
- Flatman, Rosenkranz, Evers, Bartels, Kokkendorff, Knudsen, and Nielsen, 2016.** Andrew Flatman, Brigitte Rosenkranz, Kristian Evers, Peter Bartels, Simon Kokkendorff, Thomas Knudsen, and Thorbjørn Nielsen. *Quality assessment report to the Danish Elevation Model (DK-DEM)*, 2016. URL <https://www.kortforsyningen.dk/sites/default/files/qualityassessmentdk-dem.pdf>. seen 09-02-17.
- GeoDanmark, 2014a.** GeoDanmark. *FOT specifikation 5.1 forenkelt udgave*, 2014. URL <http://www.geodanmark.dk/Materiale/files/fot5/dk-pdf-fot51-forenklet.pdf>. seen 18-04-17.
- GeoDanmark, 2017.** GeoDanmark. *Om GeoDanmark*, 2017. URL <http://www.geodanmark.dk/menu/organisation/om-geodanmark>. seen 20-02-17.
- GeoDanmark, 2014b.** GeoDanmark. *Ny forretningsmodel for FOTdanmark*, 2014. URL <http://www.geodanmark.dk/Materiale/files/fotdanmarks-forretningsmodel-2/fotdanmarks-forretningsmodel>. seen 20-02-17.
- Geodatastyrelsen, 2015.** Geodatastyrelsen. *Brug af GeoDanmark ortofoto med nærinfrarød lag*, 2015. URL <https://www.kortforsyningen.dk/sites/default/files/ortonircompressed.pdf>. seen 27-04-17.

- GeoFyn A/S, 2017.** GeoFyn A/S. *Hvad er Geo Fyn A/S?*, 2017. URL <http://www.geofyn.dk/index.php?tab=om>. seen 03-04-17.
- GeoNetworking, 2016.** GeoNetworking. *SenseFly eBee Mapping UAV*, 2016. URL <http://geonetworking.com/index.php/sensefly-ebest-mapping-uav-w-hig-res-rgb-camera.html>. seen 15-02-17.
- Google Earth, 2017.** Google Earth. *Aerial photo, 55°21'27.17"N, 10°25'8.36"E*, 2017. seen 29-05-17.
- Höhle, 2009.** Joachim Höhle. *Updating of the Danish Elevation Model by means of photogrammetric methods*, AAU - Department of Planning, 2009.
- JARS, 1996.** JARS. *Chapter 11 Image Processing - Classification*, 1996. URL <http://wtlab.iis.u-tokyo.ac.jp/~wataru/lecture/rsgis/rsnote/cp11/11-7-1.gif>. Japan Association of Remote Sensing, seen 30-04-17.
- Kršák, Blišťan, Pauliková, Pušárová, Kovanica, Palková, and Zelizňáková, 2015.** B. Kršák, P. Blišťan, A. Pauliková, P. Pušárová, L. Kovanica, J. Palková, and V. Zelizňáková. *Use of low-cost UAV photogrammetry to analyze the accuracy of a digital elevation model in a case study*, 2015. URL <http://www.sciencedirect.com/science/article/pii/S0263224116301749>. seen 16-03-17.
- Kumpumki, 2014.** Teemu Kumpumki. *LAS data reader / writer / converter*, 2014. URL <https://se.mathworks.com/matlabcentral/fileexchange/48073-lasdata>. Tampere University of Technology, seen 30-04-17.
- Küng, 2016.** Olivier Küng. *Webinar 2: Ground Control Points*, 2016. URL https://www.youtube.com/watch?v=Ylc_HSev3a4#t=37m52s. seen 16-03-17.
- Larsen, Madsen, and Maroukhas, 2016.** Henrik Brændskov Larsen, Nina Stahl Madsen, and Meletios-Apollon Maroukhas. *Unmanned aerial vehicle mapping at Opera Hedeland*, 2016.
- Leica, 2017.** Leica. *Leica iCON iXE3 - 3D System*, 2017. URL <http://leica-geosystems.com/products/machine-control-systems/excavator/leica-icon-ixe3---3d-system>. seen 29-03-17.
- Luhmann, Robson, Kyle, and Hayley, 2006.** T. Luhmann, S. Robson, S. Kyle, and I. Hayley. *Close Range Photogrammetry*. Whittles Publishing, 2006.
- MathWorks, 2017.** MathWorks. *eps*, 2017. URL <https://se.mathworks.com/help/matlab/ref/eps.html>. seen 15-05-17.
- MathWorks, 2017.** MathWorks. *2-D median filtering*, 2017. URL <https://se.mathworks.com/help/images/ref/medfilt2.html>. seen 30-05-17.

- Matthesen and Schmidt, 2014.** Anders Westh Matthesen and Kathrine Schmidt. *Terrain Modelling - DTM generation using UAVs*, AAU - Department Of Planning, 2014.
- Nath, Mishra, Dey, Kar, and Chakraborty, 2014.** Siddhartha Sankar Nath, Girish Mishra, Nilanjan Dey, Jainyaseni Kar, and Sayan Chakraborty. *A Survey of Image Classification Methods and Techniques*, 2014. URL <http://ieeexplore.ieee.org.zorac.aub.aau.dk/stamp/stamp.jsp?arnumber=6993023>. seen 30-04-17.
- Nelson, Reuter, and Gessler, 2009.** A. Nelson, H.I. Reuter, and P. Gessler. *Geomorphometry. Developments in Soil Science - Volume 33*. Elsevier, 2009. 3. DEM Production Methods and Sources.
- Pfeifer, 2008.** Norbert Pfeifer. *Topographic Laser Ranging and Scanning: Principles and Processing*. 1. edition. CRC Press, 2008. Chapter 11.
- Pix4D, 2016.** Pix4D. *DSM and Orthomosaic*, 2016. URL <https://support.pix4d.com/hc/en-us/articles/202557769-Menu-Process-Processing-Options-3-DSM-Orthomosaic-and-Index-DSM-and-Orthomosaic#label5&gsc.tab=0>. seen 21-03-17.
- Pix4D, 2017a.** Pix4D. *How to automatically generate a Digital Terrain Model (DTM)*, 2017. URL <https://support.pix4d.com/hc/en-us/articles/202560579-How-to-automatically-generate-a-Digital-Terrain-Model-DTM-#gsc.tab=0>. seen 21-03-17.
- Pix4D, 2017b.** Pix4D. *Step 1. Before Starting a Project > 1. Designing the Image Acquisition Plan > a. Selecting the Image Acquisition Plan Type*, 2017. URL <https://support.pix4d.com/hc/en-us/articles/202557459-Step-1-Before-Starting-a-Project-1-Designing-the-Image-Acquisition-Plan-a-Selecting-the-Image-Acquisition-Plan-Type#gsc.tab=0>. seen 28-05-17.
- Potůčková, 2016.** Markéta Potůčková. *Small Scale Mapping, lecture 2*, Aalborg University, 2016.
- Raasthøj, 2015.** Knud Raasthøj. *Droner i kommunal tjeneste*, 2015. URL <http://www.fyens.dk/erhverv/Droner-i-kommunal-tjeneste/artikel/2665756>. seen 24-02-17.
- Richards and Jia, 2006.** John A. Richards and Xiuping Jia. *Remote Sensing Digital Image Analysis - An Introduction*. 4th Edition. Springer-Verlag Berlin Heidelberg, 2006.
- Rising, 2016.** Jordan Rising. *LiDAR vs Photogrammetry*, 2016. URL <http://flight-evolved.com/lidar-vs-photogrammetry/>. seen 29-05-17.
- Schuermann, 2017.** Lucas Schuermann. *CDSS DevFest Data Science Track*, 2017. URL <https://learn.devfe.st/datascience/#5.4>. seen 27-04-17.

- SDFE, 2009.** SDFE. *Danmarks Højdemodel 2007, DHM-2007/Punktsky*, 2009. URL https://kortforsyningen.dk/sites/default/files/old_gst/DOKUMENTATION/Data/dk_dhm-2007_punktsky_okt_2014.pdf. seen 21-02-17.
- SDFE, 2015.** SDFE. *Danmarks Højdemodel, DHM/Punktsky*, 2015. URL https://www.kortforsyningen.dk/sites/default/files/dk_dhm_punktsky_v2_1_aug_2016.pdf. seen 21-02-17.
- SDFE, 2015.** SDFE. *Danmarks Højdemodel, DHM/Overflade*, 2015. URL https://www.kortforsyningen.dk/sites/default/files/dk_dhm_overflade_v2_1_aug_2016.pdf. seen 06-03-17.
- SONY, 2016.** SONY. *WX220 Compact Camera*, 2016. URL <http://www.sony.com/electronics/cyber-shot-compact-cameras/dsc-wx220>. seen 15-02-17.
- Teknologisk Institut, 2016.** Teknologisk Institut. *Kortlægning af droner i Danmark*, 2016. URL http://www.mynewsdesk.com/material/document/54374/download?resource_type=resource_document. seen 24-02-17.
- Terrasolid, 2016.** Terrasolid. *TerraScan User's Guide*, 2016. URL <https://www.terrasolid.com/download/tscan.pdf>. seen 19-04-17.
- The Danish Government, 2016.** The Danish Government. *Danmarks dronestrategi*, 2016. URL <http://ufm.dk/publikationer/2016/filer/dronestrategi-2016.pdf>. seen 22-02-17.
- The Danish Road Directorate, 2016.** The Danish Road Directorate. *Luftfoto af det nye tilslutningsanlæg Odense SE*, 2016. URL <http://vejdirektoratet.dk/DA/vejprojekter/tsa50/PublishingImages/TSA50.jpg>. seen 01-06-17.
- Vejdirektoratet, 2014.** Vejdirektoratet. *Fotos, film og kort*, 2014. URL <http://vejdirektoratet.dk/DA/vejprojekter/tsa50/Visualisering/Sider/default.aspx>. seen 08-03-17.
- Vejdirektoratet, 2015.** Vejdirektoratet. *Droner tæller trafikken*, 2015. URL <http://www.vejdirektoratet.dk/DA/om-os/nyheder-og-presse/nyheder/Sider/Droner-t%C3%A6ller-trafikken.aspx>. seen 24-02-17.
- Vejdirektoratet, 2016.** Vejdirektoratet. *Vejdirektoratet: Flyvende hjælp under anlæg af veje*, 2016. URL <http://ufm.dk/forskning-og-innovation/indsatsomrader/droner-i-danmark/cases/vejdirektoratet>. seen 27-02-17.
- Vosselman, 2008.** George Vosselman. *Encyclopedia of GIS*. Springer US, 2008. Laser Scanning.

- Wheaton, 2008.** Joseph M. Wheaton. *Uncertainties in morphological sediment budgeting of rivers*, School of Geography, University of Southampton, 2008.
- Wolf and Ghilani, 1997.** Paul R. Wolf and Charles D. Ghilani. *Adjustment computations : statistics and least squares in surveying and GIS*. 3rd Edition. John Wiley & Sons, 1997.
- Wu, 2017.** Dapeng Oliver Wu. *Unsupervised Classification algorithms*, 2017. URL <http://www.wu.ece.ufl.edu/books/EE/communications/UnsupervisedClassification.html>. seen 30-04-17.

Appendix

GeoFyn specification **A**

Quality Report



Generated with Pix4Dmapper Pro version 3.1.18



Important: Click on the different icons for:



Help to analyze the results in the Quality Report



Additional information about the sections



Click [here](#) for additional tips to analyze the Quality Report

Summary



Project	2016-11-28-odense
Processed	2017-01-20 09:13:24
Camera Model Name(s)	DSC-WX220_4.4_4896x3672 (RGB)
Average Ground Sampling Distance (GSD)	2.75 cm / 1.08 in
Area Covered	2.8483 km ² / 284.832 ha / 1.1003 sq. mi. / 704.198 acres

Quality Check



Images	median of 72633 keypoints per image	
Dataset	1215 out of 1271 images calibrated (95%), all images enabled	
Camera Optimization	0.58% relative difference between initial and optimized internal camera parameters	
Matching	median of 5186.97 matches per calibrated image	
Georeferencing	yes, 32 GCPs (32 3D), mean RMS error = 0.014 m	

Preview

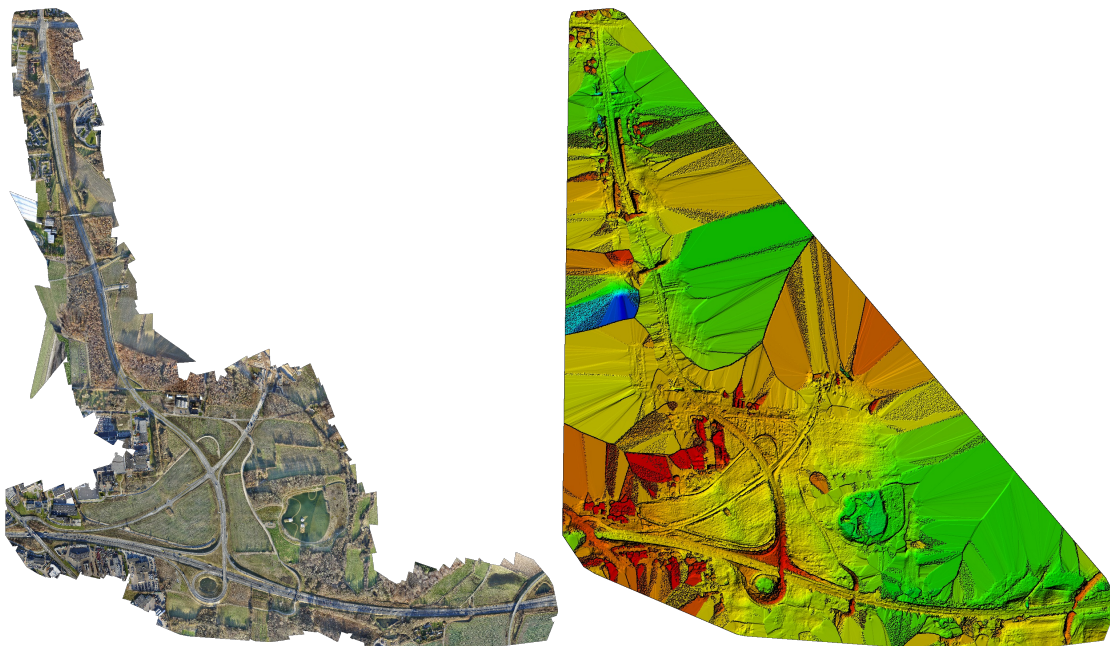


Figure 1: Orthomosaic and the corresponding sparse Digital Surface Model (DSM) before densification.

Calibration Details



Number of Calibrated Images	1215 out of 1271
Number of Geolocated Images	1271 out of 1271

? Initial Image Positions

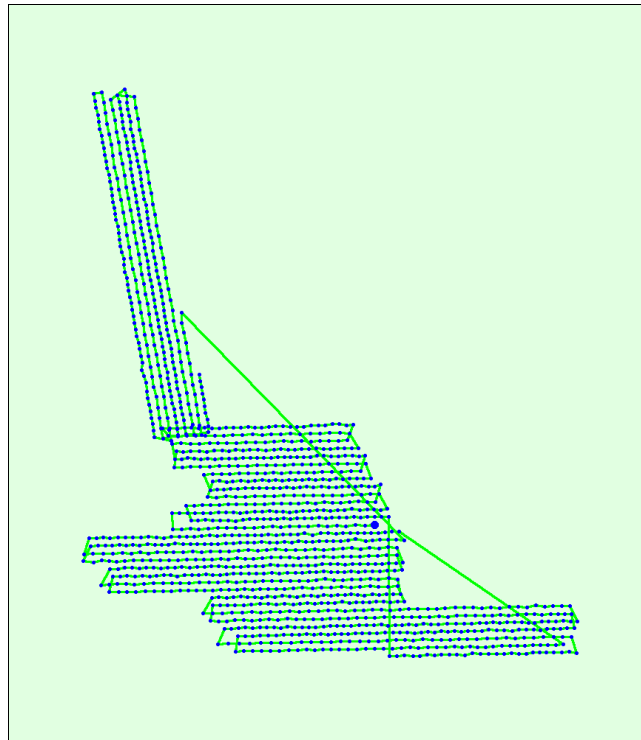


Figure 2: Top view of the initial image position. The green line follows the position of the images in time starting from the large blue dot.

? Computed Image/GCPs/Manual Tie Points Positions





Figure 3: Offset between initial (blue dots) and computed (green dots) image positions as well as the offset between the GCPs initial positions (blue crosses) and their computed positions (green crosses) in the top-view (XY plane), front-view (XZ plane), and side-view (YZ plane). Red dots indicate disabled or uncalibrated images.

 Overlap



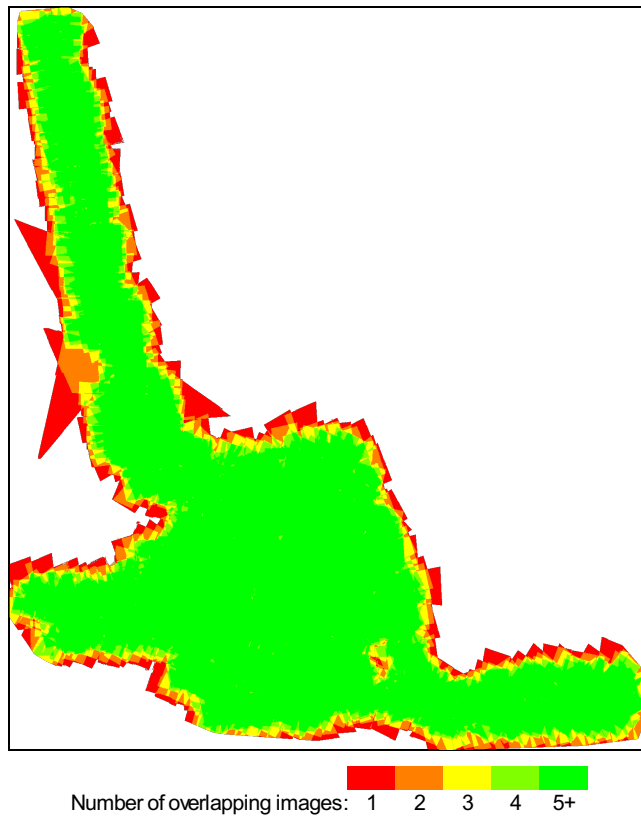


Figure 4: Number of overlapping images computed for each pixel of the orthomosaic. Red and yellow areas indicate low overlap for which poor results may be generated. Green areas indicate an overlap of over 5 images for every pixel. Good quality results will be generated as long as the number of keypoint matches is also sufficient for these areas (see Figure 5 for keypoint matches).

Bundle Block Adjustment Details

Number of 2D Keypoint Observations for Bundle Block Adjustment	6947371
Number of 3D Points for Bundle Block Adjustment	2895185
Mean Reprojection Error [pixels]	0.136

Internal Camera Parameters

DSC-WX220_4.4_4896x3672 (RGB). Sensor Dimensions: 6.170 [mm] x 4.627 [mm]

EXIF ID: DSC-WX220_4.4_4896x3672

	Focal Length	Principal Point x	Principal Point y	R1	R2	R3	T1	T2
Initial Values	3628.284 [pixel] 4.572 [mm]	2447.997 [pixel] 3.085 [mm]	1836.004 [pixel] 2.314 [mm]	0.012	-0.045	0.050	0.005	0.003
Optimized Values	3607.023 [pixel] 4.546 [mm]	2458.664 [pixel] 3.098 [mm]	1824.479 [pixel] 2.299 [mm]	-0.005	-0.007	0.005	-0.000	0.001

The number of Automatic Tie Points (ATPs) per pixel, averaged over all images of the camera model, is color coded between black and white. White indicates that, on average, more than 16 ATPs have been extracted at the pixel location. Black indicates that, on average, 0 ATPs have been extracted at the pixel location. Click on the image to see the average direction and magnitude of the re-projection error for each pixel. Note that the vectors are scaled for better visualization.

2D Keypoints Table

Number of 2D Keypoints per Image	Number of Matched 2D Keypoints per Image
----------------------------------	--

Median	72633	5187
Min	20217	48
Max	91263	21785
Mean	67530	5718

? 3D Points from 2D Keypoint Matches



	Number of 3D Points Observed
In 2 Images	2160271
In 3 Images	483586
In 4 Images	147326
In 5 Images	61303
In 6 Images	26128
In 7 Images	10928
In 8 Images	4210
In 9 Images	1127
In 10 Images	243
In 11 Images	35
In 12 Images	18
In 13 Images	6
In 14 Images	4

? 2D Keypoint Matches





Figure 5: Computed image positions with links between matched images. The darkness of the links indicates the number of matched 2D keypoints between the images. Bright links indicate weak links and require manual tie points or more images.

Geolocation Details

Ground Control Points

GCP Name	Accuracy XY/Z [m]	Error X[m]	Error Y[m]	Error Z [m]	Projection Error [pixel]	Verified/Marked
OK3 (3D)	0.020/ 0.020	-0.008	-0.034	0.001	0.535	8 / 8
OK6 (3D)	0.020/ 0.020	-0.017	-0.016	-0.002	0.520	9 / 9

VD9 (3D)	0.020/0.020	0.002	-0.006	-0.002	0.269	9 / 9
VD2 (3D)	0.020/0.020	-0.003	0.024	0.005	0.350	10 / 10
OK7 (3D)	0.020/0.020	-0.001	0.003	0.004	0.211	8 / 8
OK1 (3D)	0.020/0.020	0.011	0.006	-0.004	0.103	5 / 5
MB5 (3D)	0.020/0.020	-0.004	-0.019	-0.001	0.383	12 / 12
MB6 (3D)	0.020/0.020	-0.002	-0.002	-0.009	0.270	10 / 10
OK2 (3D)	0.020/0.020	0.005	-0.003	-0.008	0.680	8 / 8
OK5 (3D)	0.020/0.020	0.003	-0.004	0.001	0.532	7 / 7
OK8 (3D)	0.020/0.020	0.017	-0.002	-0.002	0.331	8 / 8
OK4 (3D)	0.020/0.020	0.020	-0.031	-0.001	0.287	10 / 10
MB4 (3D)	0.020/0.020	0.004	0.005	0.001	0.430	10 / 10
MB2 (3D)	0.020/0.020	0.033	-0.017	0.008	0.386	8 / 8
MB3 (3D)	0.020/0.020	0.001	-0.004	0.005	0.639	9 / 9
MB1 (3D)	0.020/0.020	0.008	-0.012	-0.000	0.258	6 / 6
VD7 (3D)	0.020/0.020	-0.000	0.020	0.001	0.502	7 / 7
VD6 (3D)	0.020/0.020	-0.008	0.015	0.004	0.240	5 / 5
VD8 (3D)	0.020/0.020	-0.009	-0.003	0.002	0.625	5 / 5
VD4 (3D)	0.020/0.020	0.009	0.036	0.000	0.717	9 / 9
VD3 (3D)	0.020/0.020	-0.033	0.035	-0.020	0.245	6 / 6
SU01 (3D)	0.020/0.020	-0.010	0.008	0.033	0.541	4 / 4
SU02 (3D)	0.020/0.020	0.003	-0.010	-0.059	0.301	5 / 5
SU03 (3D)	0.020/0.020	0.008	-0.002	0.012	0.266	6 / 6
SU04 (3D)	0.020/0.020	0.019	-0.028	0.006	0.477	5 / 5
SU05 (3D)	0.020/0.020	-0.007	0.010	0.002	0.446	4 / 4
SU06 (3D)	0.020/0.020	-0.014	0.008	0.022	0.370	4 / 4
SU07 (3D)	0.020/0.020	-0.014	0.000	0.005	0.909	6 / 6
SU08 (3D)	0.020/0.020	-0.024	0.011	-0.001	0.882	6 / 6
SU09 (3D)	0.020/0.020	-0.003	0.013	-0.002	0.166	7 / 7
SU10 (3D)	0.020/0.020	-0.009	0.000	-0.001	0.407	6 / 6
SU11 (3D)	0.020/0.020	-0.001	0.025	-0.031	0.272	3 / 3
Mean [m]		-0.000650	0.000784	-0.001038		
Sigma [m]		0.012867	0.016864	0.014642		
RMS Error [m]		0.012883	0.016882	0.014678		

Localisation accuracy per GCP and mean errors in the three coordinate directions. The last column counts the number of calibrated images where the GCP has been automatically verified v.s. manually marked.

? Absolute Geolocation Variance



Mn Error [m]	Max Error [m]	Geolocation Error X[%]	Geolocation Error Y[%]	Geolocation Error Z[%]
-	-6.06	0.00	0.00	0.00
-6.06	-4.85	0.00	0.00	0.00
-4.85	-3.64	0.09	0.00	0.17
-3.64	-2.42	0.94	0.43	1.11
-2.42	-1.21	4.00	4.25	5.53
-1.21	0.00	48.38	37.16	46.17
0.00	1.21	39.97	54.51	40.82
1.21	2.42	5.02	3.32	6.21
2.42	3.64	1.45	0.26	0.00
3.64	4.85	0.09	0.09	0.00
4.85	6.06	0.09	0.00	0.00
6.06	-	0.00	0.00	0.00
Mean [m]		0.405380	1.398848	38.321075
Sigma [m]		0.877605	0.721515	0.844096
RMS Error [m]		0.966708	1.573963	38.330370

Min Error and Max Error represent geolocation error intervals between -1.5 and 1.5 times the maximum accuracy of all the images. Columns X, Y, Z show the percentage of images with geolocation errors within the predefined error intervals. The geolocation error is the difference between the initial and computed image

positions. Note that the image geolocation errors do not correspond to the accuracy of the observed 3D points.

Geolocation Bias	X	Y	Z
Translation [m]	0.404833	1.382691	38.388459

Bias between image initial and computed geolocation given in output coordinate system.

Relative Geolocation Variance



Relative Geolocation Error	Images X[%]	Images Y[%]	Images Z[%]
[-1.00, 1.00]	82.74	88.01	89.88
[-2.00, 2.00]	95.66	98.47	99.66
[-3.00, 3.00]	99.15	99.83	100.00
Mean of Geolocation Accuracy [m]	1.016274	1.016274	1.378202
Sigma of Geolocation Accuracy [m]	0.137260	0.137260	0.330532

Images X, Y, Z represent the percentage of images with a relative geolocation error in X, Y, Z.

Geolocation Orientational Variance	RMS [degree]
Omega	5.155
Phi	4.090
Kappa	8.533

Geolocation RMS error of the orientation angles given by the difference between the initial and computed image orientation angles.

Initial Processing Details



System Information



Hardware	CPU: Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz RAM: 32GB GPU: NMDIA GeForce GTX 760 (Driver: 10.18.13.5362)
Operating System	Windows 10 Home, 64-bit


Coordinate Systems



Image Coordinate System	WGS84
Ground Control Point (GCP) Coordinate System	ETRS89 / UTMzone 32N
Output Coordinate System	ETRS89 / UTMzone 32N

Processing Options



Detected Template	 3D Maps
Keypoints Image Scale	Full, Image Scale: 1
Advanced: Matching Image Pairs	Aerial Grid or Corridor
Advanced: Matching Strategy	Use Geometrically Verified Matching: no
Advanced: Keypoint Extraction	Targeted Number of Keypoints: Automatic
Advanced: Calibration	Calibration Method: Standard Internal Parameters Optimization: All External Parameters Optimization: All Rematch: Auto, no Bundle Adjustment: Classic

Point Cloud Densification details



Processing Options



Image Scale	multiscale, 1/2 (Half image size, Default)
Point Density	Optimal
Minimum Number of Matches	3
3D Textured Mesh Generation	yes
3D Textured Mesh Settings:	Resolution: Medium Resolution (default) Color Balancing: no
Advanced: 3D Textured Mesh Settings	Sample Density Divider: 1
Advanced: Matching Window Size	7x7 pixels
Advanced: Image Groups	group1
Advanced: Use Processing Area	yes
Advanced: Use Annotations	yes
Advanced: Limit Camera Depth Automatically	no
Time for Point Cloud Densification	03h:26m:17s
Time for 3D Textured Mesh Generation	31m:08s

Results



Number of Processed Clusters	2
Number of Generated Tiles	6
Number of 3D Densified Points	157423701
Average Density (per m ³)	133.32

DSM, Orthomosaic and Index Details



Processing Options



DSM and Orthomosaic Resolution	1 x GSD (2.75 [cm/pixel])
DSM Filters	Noise Filtering: yes Surface Smoothing: yes, Type: Sharp
Raster DSM	Generated: yes Method: Inverse Distance Weighting Merge Tiles: yes
Orthomosaic	Generated: yes Merge Tiles: yes GeoTIFF Without Transparency: no Google Maps Tiles and KML: no
Raster DTM	Generated: yes Merge Tiles: yes
DTM Resolution	5 x GSD (2.75 [cm/pixel])
Contour Lines Generation	Generated: yes Contour Base [m]: 0 Elevation Interval [m]: 0.25 Resolution [cm]: 10 Minimum Line Size [vertices]: 20
Time for DSM Generation	03h:07m:47s
Time for Orthomosaic Generation	04h:36m:49s
Time for DTM Generation	03h:10m:
Time for Contour Lines Generation	24s

SV: Specialeemne

Sten Frandsen [sfr@odense.dk]

Sendt: 7. februar 2017 11:04**Til:** Henrik Brændskov Larsen; Jesper Gaardboe Jensen [jgj@geofyn.dk]**Cc:** Nina Stahl Madsen; Andrew Flatman [anfla@sdf.dk]; Eskil Kjærshøj Nielsen [eskni@sdf.dk]; Hans Hansen [hans@dronekompagniet.dk]**Vedhæftede filer:** Paspunkter Endelig.ID (148 B) ; Paspunkter Endelig.MAP (3 KB) ; Paspunkter Endelig.DAT (3 KB) ; Paspunkter Endelig.TAB (542 B) ; Supplerende paspunkter.ID (108 B) ; Supplerende paspunkter.MAP (2 KB) ; Supplerende paspunkter.TAB (423 B) ; Supplerende paspunkter.DAT (2 KB)

Hej Nina og Henrik.

Jeg prøver herunder at besvare jeres spørgsmål

1. Hvad har formålet med droneflyvningen været?

Droneflyvningen er et projekt i GeoFyn-regi, hvor vi har 3 cases. De uploadede data er til brug for case 1+2.

- Indsamling af data til brug for ajourføring af terrænmodel (DTM) i et område hvor der er sket markante terrænreguleringer (anlæg af ny motorvejstilslutning). Efterfølgende skal den nyopmålte DTM primært bruges i ScalgoLive til beregning af klimascenarier og på sigt skal den indgå i en nationale DHM.
- Indsamling af data til brug for ajourføring af vektordata (GeoDanmark) – primært brug af ortofoto.
- Optagelser af fotos/videoer til brug for visualiseringer/kommunikation.

Hvilken drone samt planlægningssoftware anvender i?

Drone: fixed wing senseFly eBee.

Payload: Sony DSC-WX220, 18.2 MP, RGB, senseFly modified.

Mission planner software: senseFly eMotion 2 - version 2.4.12

Anvender dronen kameraet Sony DSC-WX220 og hvor meget zoom er der brugt (brændvidde/focal length)?

Ja, det fremgår af Quality Report PDF'en - søg efter Internal Camera Parameters.

Hvad var flyvehøjden?

Planlagt flyvehøjde 100 meter / 2,8 cm GSD.

Hvilken type Ground Control Points har i anvendt, og kan vi få en koordinatliste for dem?

Vi har selv udarbejdet vore signalerede GCP'er, dels i kraft af hvide plastiklåg (fra plastikspande) og dels af hvide bemalede cirkler på faste underlag. Til begge signaleringer er diameteren mellem 22-29 cm. Desuden er der indmålt supplerende naturlige GCP'er, som bemalede kvadratiske vejpunkter og nedløbsriste.

Alle punkter er indmålt med Leica præcisions-GPS.

Laz filerne synes umiddelbart at være komprimeret, er det muligt at få en rå punktsky for et udvalgt område?

Svar fra Andrew: i kan anvende LasTools til at konvertere LAZ filerne til andre formater. Se her:

<https://rapidlasso.com/laszip/>

Vi håber, at i finder materialet spændende og vil brug det i jeres projekt. Jeg vedhæfter GCP signaleringsplan med type og koordinater.

Såfremt i måtte have yderligere spørgsmål, er i meget velkommen til at skrive.

Med venlig hilsen

Sten Frandsen

GIS-medarbejder

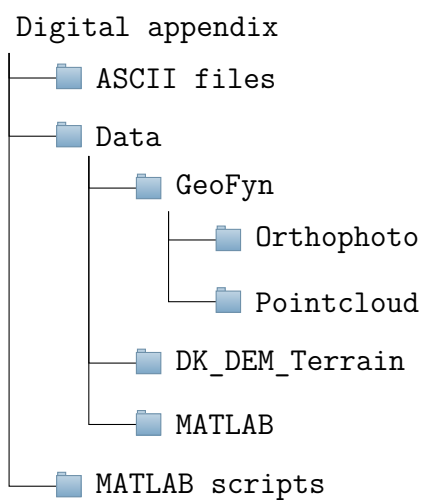
Direkte tlf. 6551 2563

Mobil 2869 1372

sfr@odense.dk

Digital appendix B

In the following the folder structure of the digital appendix can be seen.



In order to run the change detection algorithm, the MATLAB script “*change_detection_algorithm.m*” can be used. This script will run all the individual MATLAB scripts and produce an ASCII file with the change detection. This ASCII file can be read by ESRI ArcMap with the “ascii to raster” tool.

select_data.m C

```
1  %%% select_data.m %%%
2
3  % This script import the UAV point cloud and selects the points ...
   inside a
4  % given area
5
6  las_data=lasdata...
7  ('2016-11-28-odense_group1_densified_point_cloud_part_1-6.las'); % ...
   UAV point cloud
8  eq = las_data.x;
9  nq = las_data.y;
10 zq = las_data.z;
11
12
13 %Test area 1 - Low vegetation
14 case_1_e = 590502;
15 case_1_n = 6134584;
16
17 %Test area 2 - Field
18 case_2_e = 589952;
19 case_2_n = 6134808;
20
21 %Test area 3 - Highway road
22 case_3_e = 590086;
23 case_3_n = 6134654;
24
25 %Test area 4 - Steep slope
26 case_4_e = 590572;
27 case_4_n = 6134558;
28
29 %Odense SE 1x1 km
30 Odense_SE_e = 589634;
31 Odense_SE_n = 6134458;
32
33 e_start=Odense_SE_e;
34 n_start=Odense_SE_n;
35 e_end=e_start+1000;
```

```
36 n_end=n_start+1000;
37 pointsInRange = ((e_start < eq) & (eq < e_end)) & ((n_start < nq) & ...
    (nq < n_end));
38
39 data=[eq(pointsInRange) nq(pointsInRange) zq(pointsInRange)];
40
41 clearvars -except data case_4_e case_4_n las_data
```

data_to_cells.m D

```
1  %%% data_to_cells.m %%%
2
3  % See Section 9.1 in project report (also mentioned in Section 6.3.1)
4  % This script divide data into cells.
5  % The script loads the point cloud data which was imported in ...
   select_data.m
6
7  %%% Input:
8  % To run the script the correct dataset need to be chosen and the ...
   cellsize and coordinates of the lower left corner and upper right ...
   corner need to be specified.
9
10 %%% Output:
11 % The output is a string (cell_data) starting from the first cell in ...
   the lower left corner and moving through the columns and then ...
   moving to the next row. The string has the following format; #of ...
   points in cell_1 e1 e2 e3 .. n1 n2 n3 .. h1 h2 h3 #of points in ...
   cell_2 ....
12
13 load('test_case_1_590502_6134584_20.mat'); % Load data
14 data = sortrows(data,2); % Sort data based on n-coor. from small to big
15 cellsize = 0.4; % Size of grid cells [m]
16 e_start = 590502; % Easting coordinate for lower left corner [m]
17 n_start = 6134584; % Northing coordinate for lower left corner [m]
18 e_end = e_start+20; % Easting coordinate for upper right corner ...
   [m] - specified from the size of the areas
19 n_end = n_start+20; % Northing coordinate for upper right ...
   corner [m] - specified from the size of the areas
20
21 columns = (e_end-e_start)/cellsize; % Calculate number of columns ...
   in grid
22 rows = (n_end-n_start)/cellsize; % Calculate number of rows in grid
23 num_of_cells = rows*columns; % Number of cells in grid
24
25 cell_n=0; % Initial value for updating the northing coordinate
26 cell_data = zeros((3*length(data(:,1)))+int64(num_of_cells),1); % ...
   Initialise vector with zeroes for storing the new data divided ...
```

```

        into cells
27 cell_data_from=1;    % First "from" index
28 e_cell_mid=[];       % Initialize vector with middle e-coordinate of cell
29 n_cell_mid=[];       % Initialize vector with middle n-coordinate of cell
30
31 data_stop = length(data(:,1)); % Stop criterion when selecting the ...
    n-coor for the cell
32 e_coor = [];         % Initialize vector for e-coor.
33 n_coor = [];         % Initialize vector for n-coor.
34 h_coor = [];         % Initialize vector for h-coor.
35
36 for i = 1:rows % Outer for-loop runs the number of rows
37
38     cell_e=0; % Initial variable for updating the easting coordinate.
39             % Resets when changing to a new row
40
41     n_cell_start = n_start+cell_n;          % n-coor for start of ...
        cell, increases by cellsize for each run
42     n_cell_end = n_start+cellsize+cell_n; % n-coor for end of cell, ...
        increases by cellsize for each run
43     data_sorted_y = [];                    % Initialize vector for ...
        data inside the cell where n-coor is sorted
44
45     for j = 1:columns % Inner for-loop runs the number of columns
46
47         e_cell_start=e_start+cell_e;        % e-coor for start of ...
            cell, increases by cellsize for each run
48         e_cell_end=e_start+cellsize+cell_e; % e-coor for end of ...
            cell, increases by cellsize for each run
49
50         e_cell_mid = [e_cell_mid;e_cell_start+(cellsize/2)]; % ...
            Vector with middle e-coor of cell
51         n_cell_mid = [n_cell_mid;n_cell_start+(cellsize/2)]; % ...
            Vector with middle n-coor of cell
52
53         if j==1 && i==1 % for the first cell (1,1) k=1
54             k=1;
55             while (data(k,2)) > n_cell_start && (data(k,2)) ≤ n_cell_end ...
                %select data which is inside the given interval
56                 data_sorted_y = [data_sorted_y;(data(k,:))]; %the ...
                    selected data is stored in data_sorted_y
57             k=k+1;
58             end
59             elseif j==1 % For the existing rows.
60                 % The loop is only run once for every row.
61                 while (data(k,2)) > n_cell_start && (data(k,2)) ≤ ...
                    n_cell_end %select data which is inside the given interval
62                     data_sorted_y = [data_sorted_y;(data(k,:))]; %the ...
                        selected data is stored in data_sorted_y

```

```

63         if k==data_stop %breaks loop when stop criterion is met, ...
           which is the number of points in the loaded data
64         break
65     end
66     k=k+1;
67     end
68 end
69
70 %%% Now the selected data is sorted according to the e-coor.
71 if j==1 % Is only run once for every row
72     data_sorted_x=sortrows(data_sorted_y,1); %data is sorted ...
           after e-coor
73 end
74 data_stop2=length(data_sorted_y(:,1)); %Stop criterion is ...
           set according to the number of points in data_sorted_y, ...
           hence number of points in the row
75
76 if j==1 %for the first cell in each row g=1
77     g=1;
78     while ((data_sorted_x(g,1)) > e_cell_start && ...
79           (data_sorted_x(g,1) ≤ e_cell_end)) % select data ...
           which is inside the given interval
80
81         %The data inside the given cell is placed in separate ...
           vectors for e,n,h
82         e_coor=[e_coor,data_sorted_x(g,1)]; % Vector with e-coor
83         n_coor=[n_coor,data_sorted_x(g,2)]; % Vector with n-coor
84         h_coor=[h_coor,data_sorted_x(g,3)]; % Vector with h-coor
85     g=g+1;
86     end
87 else % for the remaining cells in each row
88     while ((data_sorted_x(g,1)) > e_cell_start && ...
89           (data_sorted_x(g,1) ≤ e_cell_end)) %select data ...
           which is inside the given interval
90
91         %The data inside the given cell is placed in separate ...
           vectors for e,n,h
92         e_coor=[e_coor,data_sorted_x(g,1)]; % Vector with e-coor
93         n_coor=[n_coor,data_sorted_x(g,2)]; % Vector with n-coor
94         h_coor=[h_coor,data_sorted_x(g,3)]; % Vector with h-coor
95
96     if g==data_stop2 %breaks loop when stop criterion is met
97         break
98     end
99     g=g+1;
100 end
101 end
102 count=length(e_coor); % Counts number of points in cell
103

```



```

104     cell_e = cell_e+cellsize; % Increase value for updating ...
        easting coordinates
105
106     if count ≤ 3 % If less than 4 points in the cell it is not used
107         count = -9999; %NoData value
108         e_coor = [];
109         n_coor = [];
110         h_coor = [];
111     end
112
113     if i==1 && j==1 % Finds first index of "to"
114         cell_data_to = length([count, e_coor, n_coor, h_coor]); ...
            % First index for "to" index
115     else
116         cell_data_to=cell_data_to+length([count, e_coor, ...
            n_coor,h_coor]); % Every other "to" index
117     end
118
119     cell_data(cell_data_from:cell_data_to) = [count, e_coor, ...
        n_coor,h_coor]; % Vector for storing the points
120
121     cell_data_from=cell_data_to+1; % Updates the "from" index
122
123     %Reset vectors
124     e_coor = [];
125     n_coor = [];
126     h_coor = [];
127 end
128
129     cell_n = cell_n+cellsize; % Increase variable for updating n-coor
130
131 end
132 e_coor_mean=mean(data(:,1)); % find mean e-coordinate of the area
133 n_coor_mean=mean(data(:,2)); % find mean n-coordinate of the area
134 h_coor_mean=mean(data(:,3)); % find mean h-coordinate of the area
135 cell_data=cell_data(1:cell_data_to); % Deletes excess zeroes
136
137 clearvars -except cell_data cellsize columns e_cell_mid e_coor_mean...
138     e_end e_start h_coor_mean n_cell_mid n_coor_mean n_end ...
        n_start...
139     num_of_cells rows % Clear unnecesary variables

```

planfit.m E

```
1  %%% function planfit.m %%%
2
3  % See Section 9.2 in the project report
4  % This function runs an adjustment to find the best fit plane ...
   through a number of 3D points with ordinary least squares (OLS).
5  % The function can both handle robust and not robust adjustments.
6
7  %%% Input:
8  % A vector 'l' (from remove_outliers2.m or plan_fit_data2.m )which
9  % contain the e, n, h coordinates to the point in the cell ...
   (observations).
10 % The number of points is called n.
11 % The first n elements in the vector contains 1. coordinates to the ...
   points.
12 % The next n elements in the vector contains 2. coordinates to the ...
   points.
13 % The next n elements in the vector contains 3. coordinates to the ...
   points.
14 %
15 % The size of the vector is (3n)x(1)
16 %
17 %     ln:     Least absolute deviation norm (L_p), if ln = 2 it's ...
   normal least
18 %             squares else it's iterative least squares (IRLS) with ...
   the L_p norm
19 %     out:     If out = 1, outliers in the adjustment will be removed
20 %     plot:     If plot = 1 a plot with planes is made else no plot ...
   will be made
21
22 %%% Output: The output is specified when the function is called
23 %     it:             Number of iterations
24 %     count_outliers: Count of how many points that are removed
25 %     pix_data:       Vector with remaining points
26 %     xhat:           Solution vector with coefficients;    xhat = [a ...
   b c]'.
27 %
   The planes normal vector is given by:    n = [a ...
   b -1]'.

```

```

28 %   sigma_xhat2:    Covariance matrix for the elements
29 %   sigma_0:       Standard deviation of unit weight
30
31
32 function [it,count_outliers,pix_data,xhat,sigma_xhat2,sigma_0] = ...
    planfit(l,ln,out,plot)
33
34 n = size(l,1)/3; % Number of points in column vector
35
36 % Observations
37 x = l(1:n);      % 1. coordinates of points
38 y = l(n+1:2*n);  % 2. coordinates of points
39 z = l(2*n+1:3*n); % 3. coordinates of points
40
41 Q=eye(length(x)); % Initial weight matrix for the points
42 A = [x y ones(n,1)]; % Design matrix (partial derivatives of function)
43 xhat_old=0;       % Initial variable to determine if the loop ...
    should be breaked or not
44 it=0;            % initial variable to count the number of ...
    iterations
45 while 1
46     it=it+1;      % Update the number of iterations
47
48     N=A'*Q*A;     % Normal equation
49
50     xhat = inv(N)*A'*Q*z; %Solve least squares
51
52     r = A*xhat-z; % calculate residuals
53
54     Q = diag((ones(size(r))*100*eps+abs(r)).^(ln-2)); % Update ...
    weight matrix according to the L_n
55
56 if max(abs(xhat-xhat_old)) < 0.001 || it==150 % If the previous ...
    iteration has not changed or if the iterations is equal to 10; ...
    break loop
57     break;
58 end
59     xhat_old = xhat;
60 end
61
62 s02 = (r'*Q*r)/(n-3); % Variance factor (not robust)
63
64 sigma_0=sqrt(s02); % Standard deviation of unit weight (not robust)
65
66
67
68 sigma_xhat2 = s02*inv(N); % Covariance matrix for estimated variables
69
70 if out == 1 % Remove outliers
71     if ln < 2

```

```

72         sigma_0 = median(abs(r-median(r)))/0.6745; % Use robust ...
           standard deviation of unit weight if using IRLS
73     end
74
75     %Find outliers from IRLS
76     outliers1 = abs(r) < 3*sigma_0;      % Find outliers (logic vector)
77     outliers2 = repmat(outliers1,3,1);  % make logic vector*3
78     l_no_outliers=l(outliers2);         % Remove points which are ...
           outliers
79     count = length(l_no_outliers)/3;     % New number of points in ...
           column vector
80     count_outliers = sum(~outliers1);   % Counts how many outliers ...
           that are removed
81
82     if count ≥ 4 % If there are more than 4 points after removal of ...
           outliers make new vector with points, if not return NoData value.
83         pix_data=[count;l_no_outliers];
84     else
85         pix_data=-9999;
86     end
87 else % If using ordinary least squares without IRLS return 0, as no ...
           points have been removed.
88     pix_data = 0;
89     count_outliers = 0;
90 end
91
92 if plot == 1 %Plots plane
93     [X,Y] = meshgrid(x,y);
94     Z = xhat(1)*X + xhat(2)*Y + xhat(3);
95     s=surf(X,Y,Z, 'FaceColor', 'b', 'FaceAlpha',0.2);
96     s.EdgeColor = 'none';
97     hold on
98     axis equal
99 else
100 end

```


remove_outliers.m F

```
1  %%% remove_outliers.m %%%
2
3  % See Section 9.3 in the project report
4  % This script removes outliers based on an adjustment with a given ...
   L_p norm
5  % The script uses the cell_data which was made in data_to_cells.m
6
7  %%% Input:
8  % cell_data, which contain the data sorted in cells.
9  % The chosen L_p-norm for the adjustment need to be specified.
10
11 %%% Output:
12 % The output is a new a string of data without outliers.
13 % The new string is also called (cell_data) and has the same structure
14
15 % if data_to_cells.m have not just been run, the saved workspace is ...
   loaded
16
17 L_norm=1.3;           % Which L norm to use
18 num_p=cell_data(1); % Index of number of points in cell
19 count = 0;           % Variable for counting how far in the data ...
   vector we are
20 NoData = -9999;      % No data value
21 new_cell_data=zeros(length(cell_data),1); % Initialise vector with ...
   zeroes for storing the new data with outliers removed
22 cell_data_from=1; % First "from" index
23 %plane_data=zeros(num_of_cells,4); % Initialize vector for storing ...
   data from planes
24 %sigma_0_vec=zeros(num_of_cells,4); % Initialize vector for storing ...
   data from planes
25
26 h = waitbar(1, 'Please wait removing outliers...'); % Process bar
27 for i=1:num_of_cells % Runs the number of cells in data
28     if rem(i,1000) == 0
29         waitbar(i/num_of_cells,h)
30     end
31
```

```

32     if num_p == NoData
33         % If a cell with 3 or less points is found a plane should ...
           not be fitted and NoData value for the plane data should ...
           be inserted
34
35         if i==1 % Find start index for the first cell
36             cell_data_to = 1; % First index for "to" index
37         else
38             cell_data_to = cell_data_to + 1; % Every other "to" index
39         end
40
41         new_cell_data(cell_data_to:cell_data_from) = NoData; % Makes ...
           new_cell_data where outliers have been removed
42
43         cell_data_from=cell_data_to+1; % Updates the "from" index
44
45         % sigma_0_vec(i) = NoData; % Updates standard deviation of ...
           unit weight with NoData if cells have less than 3 points
46     else
47         e_coor = cell_data(count+2:count+num_p+1)-e_coor_mean; ...
           % Find e-coor for each cell and subtract the ...
           mean e-coord (avoids numerical problems)
48         n_coor = ...
           cell_data(count+num_p+2:count+num_p+3)-n_coor_mean; % ...
           Find n-coor for each cell and subtract the mean n-coor ...
           (avoids numerical problems)
49         h_coor = ...
           cell_data(count+num_p+4:count+num_p+5)-h_coor_mean; % ...
           Find h-coordinate for each cell
50         l = [e_coor; n_coor; h_coor]; % Vector with e, n, h coordinates
51
52         [r,r,adj_cell_data,r,r,r]=planfit(l,L_norm,1,0);
53         % Makes a best fit plane with ordinary least squares (OLS) ...
           according to the L1.3 norm (depending on the specified norm)
54         % The planfit.m functions removes outliers when using IRLS
55
56         if i==1 % Find start index
57             cell_data_to = length(adj_cell_data); % First index for ...
           "to" index
58         else
59             cell_data_to=length(adj_cell_data)+1; % Every ...
           other "to" index
60         end
61
62         new_cell_data(cell_data_from:cell_data_to) = adj_cell_data; ...
           % Makes new_cell_data where outliers have been removed
63
64         cell_data_from=cell_data_to+1; % Updates the "from" index
65
66         %plane_data(i,:)=[num_p count_outliers sigma_0 it];

```



```

67         % Matrix with data from plane adjustment uncomment and tell the
68         % funtion to output it
69
70         %sigma_0_vec(i) = sigma_0;
71         % Updates standard deviation of unit weight vector
72         % (used to create ascii file)
73     end
74
75     if i == num_of_cells % Break loop when all cells have been run
76         break
77     end
78
79     if num_p == -9999          % If cell is not used update the ...
        count = count+1;      % Update variable counting how ...
        far in the data vector we are
80     num_p = cell_data(count+1); % update variable for the number ...
        of points in cell
81
82     else
83         count = count+num_p*3+1; % Update variable counting how ...
        far in the data vector we are
84     num_p = cell_data(count+1); % update variable for the number ...
        of points in cell
85
86     end
87 end
88
89 %%% Overwrites and clear variables
90 cell_data=new_cell_data(1:cell_data_to);
91 % Updates cell_data and deletes excess zeroes
92 close(h)
93 clearvars -except matlab_file filepath cell_data cellsize columns ...
    e_cell_mid e_coor_mean ...
94     e_start h_coor_mean n_cell_mid n_coor_mean n_start ...
        num_of_cells rows
95     % Clear unnecessary variables

```


plan_fit_data.m G

```
1  %%% plan_fit_data.m %%%
2
3  %See Section 10.1 i the project report
4  %This script calculates best fit planes for the remaining data after ...
    outliers have been removed.
5  %The script is based on an least square adjustment with L_2 norm
6  %The script uses the cell_data which was made in remove_outliers.m
7
8  %%Input:
9  %cell_data, which contain the data without outliers sorted in cells.
10
11 %%Output:
12 %The output is the elevation, H, for the centre of each cell ...
    (H_mid_plane) and the coefficients of the plane (xhat) and ...
    standard deviation of unit weight (sigma_0) for the adjusted plane.
13
14 tic % Start timer
15 Case = matlab_file(length(matlab_file)-37:length(matlab_file)-32); ...
    %Case name
16 num_p=cell_data(1);          % Index of number of points in cell
17 count = 0;                   % Variable for counting how far ...
    in the data vector we are
18 plane_data=zeros(num_of_cells,1); % Initialize vector for storing ...
    data from planes
19 H_mid_plane=zeros(num_of_cells,1); % Initialize vector for storing ...
    height at middel of cell
20 L_norm=2;                    % Which L norm to use
21 NoData = -9999;              % No data value
22
23 h = waitbar(1,'Please wait adjusting planes...');
24 for i=1:num_of_cells % Runs the number of cells in data
25     if rem(i,1000) == 0
26         waitbar(i/num_of_cells,h)
27     end
28
29     if num_p == -9999 % If a cell with 3 or less points is found a ...
        plane should not be fitted and a NaN number for the plane ...
```

```

data should be inserted
30
31     sigma_0 = NoData*-1; % Updates standard deviation of unit ...
        weight with NoData if cells have less than 3 points
32
33     H_mid_plane(i) = NoData; % Update H-coor in middle of cell ...
        with NoData value if cells have less than 3 points
34
35     plane_data(i)= sigma_0; % Update vector for storing data ...
        from planes
36
37 else
38     e_coor = cell_data(count+2:count+num_p+1);           % Find ...
        e-coordinate for each cell
39     n_coor = cell_data(count+num_p+2:count+num_p*2+1);   % Find ...
        n-coordinate for each cell
40     h_coor = cell_data(count+num_p*2+2:count+num_p*3+1); % Find ...
        h-coordinate for each cell
41     l = [e_coor; n_coor; h_coor]; % Vector with e, n, and h ...
        coordinates
42
43     [r,r,r,xhat,r,sigma_0]=planfit(l,L_norm,0,0); % Makes a best ...
        fit plane with ordinary least squares (OLS) using the ...
        planfit.m fuction
44
45     H_mid_plane(i) = ((e_cell_mid(i)-e_coor_mean)*xhat(1)+...
46         (n_cell_mid(i)-n_coor_mean)*xhat(2)+xhat(3))+h_coor_mean; ...
        %Calculates height at middel of plane
47
48     plane_data(i)= sigma_0; % Update vector for storing data ...
        from planes
49
50 end
51
52 if i == num_of_cells % Break loop when all cells have been run
53     break
54 end
55
56 if num_p == -9999           % If cell is not used update the ...
    count by only one indices
57     count = count+1;         % Update variable counting how ...
        far in the data vector we are
58     num_p = cell_data(count+1); % update variable for the number ...
        of points in cell
59 else
60     count = count+num_p*3+1; % Update variable counting how ...
        far in the data vector we are
61     num_p = cell_data(count+1); % update variable for the number ...
        of points in cell
62 end

```

```

63 end
64
65 H_mid_plane_matrix = flipud(vec2mat(H_mid_plane,columns)); ...
    %Transform h_mid_plane from vector to matrix and flips it as the ...
    cells initially starts from the lower left corner
66 TimeSpent = toc; % Ends timers
67 close(h)
68 clearvars -except TimeSpent filepath Case Case2 cellsize columns ...
    rows e_end e_start...
69     H_mid_plane H_mid_plane_matrix n_end n_start num_of_cells ...
        plane_data...
70     NoData TimeSpent % Clear unnecesary variables

```


change_detection.m H

```
1  %%% change_detection.m %%%
2
3  % See Section 10.2, 10.3 and 10.4 i the project report
4  % This script read the DK-DEM raster model and crops the model to ...
    the extent of the test area.
5  % Afterwards the script calculates the difference between the two ...
    models(DK-DEM and UAV data) and the standard deviation of the ...
    difference
6
7  %%% Input:
8  % DK-DEM model as a .tif file
9  % Elevation of centre of cells (UAV data) from plan_fit_data.m
10 % Standard deviation of unit weight (sigma_02_matrix)
11 % The standard deviation of DK-DEM need to be specified
12 % The 'standard deviation' for the definiton error need to be specified
13
14 %%% Output:
15 % The output is a binary matrix where "1" is a change and "0" is no ...
    change
16
17
18 %%% Loads and crops the raster DEM to the extent of the test area
19 filepath_DK_DEM='C:\Users\hbl\Dropbox\P10\Digital ...
    appendix\DATA\DK_DEM_Terrain\Merge_DTM_1km_6134_589.tif'; % ...
    Insert filepath to raster DEM
20 DK_DSM = imread(filepath_DK_DEM); % loads raster file to matrix
21
22 DEM = filepath_DK_DEM(length(filepath_DK_DEM)-19:...
23     length(filepath_DK_DEM)-17); % type of DEM used
24
25 block_e_start = str2num...
26     (filepath_DK_DEM(length(filepath_DK_DEM)-6:...
27     length(filepath_DK_DEM)-4))*1000; % Find lower left e-coordinae ...
    of DEM block
28 block_n_start = str2num...
29     (filepath_DK_DEM(length(filepath_DK_DEM)-11:...
```



```

30     length(filepath_DK_DEM)-8))*1000; % Find lower left n-coordinae ...
        of DEM block
31
32 [block_rows,block_columns]=size(DK_DSM); % Find rows and columns of ...
        DEM block
33
34 diff_e_cells = round((e_start - block_e_start) / cellsize); % ...
        Calculates matrix column position of test area (upper left corner)
35 diff_n_cells = round(((block_n_start+block_rows*cellsize) - ...
        (n_start+rows*cellsize)) / cellsize); % Calculates matrix row ...
        position of test area (upper left corner)
36
37 DK_DEM_crop = DK_DSM(diff_n_cells:diff_n_cells+rows-1,...
38     diff_e_cells:diff_e_cells+columns-1); % Crops DK-DEM to the ...
        extent of the test area, as a matrix
39
40 DK_DEM_crop_vec = reshape(flipud(DK_DEM_crop)',rows*columns,1); % ...
        Crops DK-DEM to the extent of the test area as a vector
41
42
43 %% Calculates standard deviation of the difference and makes change ...
        detection
44 var_DK_DSM = 0.05^2; % Variance of the DK-DEM
45 var_def = 0.05^2; % 'Variance' of definition error
46
47 DEM_diff_matrix_0 = H_mid_plane_matrix - DK_DEM_crop; %Difference ...
        between the two DEMs as matrix
48
49 Sigma_02_matrix = (flipud(vec2mat(plane_data(:,1).^2,columns))); % ...
        Variance factor of the planes as a matrix
50
51 Sigma_diff_matrix_0 = sqrt(Sigma_02_matrix+var_DK_DSM+var_def)*3; % ...
        Standard deviation of the difference times three, according to ...
        the special law of error propogation, as a matrix
52
53 Change_detect_matrix_0 = abs(DEM_diff_matrix_0) > ...
        Sigma_diff_matrix_0; % Detect changes based on the differences ...
        and the standard deviation of difference, as a binary matrix

```

median_filter.m I

```
1  %%% median_filter.m %%%
2  %Median filter of change detection
3
4  % See Section 10.5 i the project report
5  % This script run a median filter of the change detection from ...
   change_detection.m
6
7  %%% Input:
8  % Binary matrix showing 'change' and 'no change' - ...
   Change_detect_matrix_0.
9  % The size of the search window need to be specified - preferable an ...
   odd number.
10
11 %%% Output:
12 % The output is a binary matrix where "1" is a change and "0" is no ...
   change.
13 % The output is written to an ASCII file
14
15 window = '3';
16 m = 3; % Size of window
17 n = m;
18
19 A = Change_detect_matrix_0; % The binary matrix from change_detection.m
20
21 Change_detect_filter = medfilt2(A, [m n]); % The median filter is run
22
23
24 %%% Writes results to ASCII files -- median filter change detection
25
26 % Writes file with "1" for a change and "0" for no change
27 filenameIn = 'aux_delete.txt'; % Auxillery file filename for storing ...
   the matrix
28 filenameOut=strcat...
29     ('Change_detect_',Case,'_',DEM,'_median_filter_',window,'.txt'); ...
   % Final ascii file filename
30 dlmwrite(filenameIn,Change_detect_filter,' '); % Writes auxillery ...
   ascii file with space as delmiter for matrix
```

```

31 fidin = fopen(filenameIn, 'rt'); % Opens auxillary file
32 fidout = fopen(strcat...
33     (filepath,filenameOut), 'wt'); % Opens final file
34 fprintf(fidout, 'NCOLS %d \n',columns); % Writes header
35 fprintf(fidout, 'NROWS %d \n',rows);
36 fprintf(fidout, 'XLLCORNER %d \n',e_start);
37 fprintf(fidout, 'YLLCORNER %d \n',n_start);
38 fprintf(fidout, 'CELLSIZE %4.2f \n',cellsize);
39 fprintf(fidout, 'NODATA_VALUE %d \n',NoData); % Header end
40 while true % Adds each line from the auxillary file after header
41     thisline = fgets(fidin);
42     if ~ischar(thisline); break; end %When end of file break loop
43     fwrite(fidout, thisline);
44 end
45 fclose(fidout); % Close final file
46 fclose(fidin); % Close auxillary file
47 delete(filenameIn) % Deletes auxillary file

```

test_sigma_0_vs_s.m

J

```
1  %%% test_sigma_0_vs_s.m %%%
2
3  % See Section 9.2.1 in the project report
4  % The script tests if sigma_0 and s gives the same value for a large ...
   dataset with random normal distributed errors
5
6  clc
7  clear all
8  %%% Best fit line according to:  $y = ax + b$ 
9  pts = 10000; % Points in the adjustment
10 x = [1:pts]'; % x-coordinates
11 y = normrnd(0,1,[1 pts])'; % 10000 random points with mean = 0 and ...
   std.dev. = 1
12 A = [x ones(size(x))]; % Design matrix
13 x_old = [0 0]'; % Initilize x_old matrix
14 W = eye(numel(y)); % Initilize weight matrix
15 p = 1; % Adjustment with L_1 norm
16 while 1
17 xhat = inv(A'*W*A)*A'*W*y; % Solve normal equations
18 r = A*xhat-y; %Calculate residuals
19 W = diag((ones(size(r))*100*eps + abs(r)).^(p-2)); %Update weight ...
   matrix according to L_1 norm
20 if max(abs(xhat-x_old)) < 0.001; % break loop if difference between ...
   this and the previous iteration is small
21 break
22 end
23 x_old = xhat; % Update x_old
24 end
25
26 s_l_1 = median(abs(r-median(r)))/0.6745 % Calculate robust standard ...
   deviation of unit weight
27
28
29 W = eye(numel(y)); % Initilize weight matrix
30 p = 2; % Adjustment with L_2 norm
31 while 1
32 xhat = inv(A'*W*A)*A'*W*y; % Solve normal equations
```

```

33 r = A*xhat-y; %Calculate residuals
34 W = diag((ones(size(r))*100*eps + abs(r)).^(p-2)); % Update weight ...
    matrix according to L_2 norm
35 if max(abs(xhat-x_old)) < 0.001; % break loop if difference between ...
    this and the previous iteration is small
36 break
37 end
38 x_old = xhat; % Update x_old
39 end
40
41 sigma0_l_2 = sqrt((r'*r)/(length(x)-2)) % Calculate standard ...
    deviation of unit weight

```