3D city modeling for automated verification of safety compliance rules

Vasileios Alexandridis[4], Felix Dahle[2], Amber Mulder[1], Gabriella Wiersma[5], and Yifang Zhao[3]

[1]a.e.mulder@student.tudelft.nl
[2]f.dahle@student.tudelft.nl
[3]y.zhao-18@student.tudelft.nl
[4]v.alexandridis@student.tudelft.nl
[5]m.g.wiersma@student.tudelft.nl

January 14, 2020

Gemeente Amsterdam

TU Delft 3Dgeoinfo

Contents

List of Figures

List of Tables

List of Acronyms

BGT	Basic Registration of Large-Scale Topography
BAG	Basisregistratie Adressen en Gebouwen
AHN3	Actueel Hoogtebestand Nederland
PDOK	Publieke Dienstverlening Op de Kaart
DT	Delaunay Triangulation
WOZ	Waarde Ontroerende Zaken
NHR	Nationaal Handelsregister
LoD	Level of Detail
ADE	Application Domain Extension
DIM	Dense Image Matching
PCA	Principal Component Analysis

1 Preface

During the MSc Geomatics program at the Delft University of Technology, the students execute a synthesis project. The project aims to investigate a real-world problem and to bring students and teachers/supervisors in cooperation with a company or public agency (e.g. municipalities). The students are challenged to use their existing knowledge in geomatics and to gain further experience in handling of geo-information as to come in touch with the involved stakeholders. The duration of the project is ten weeks and takes place at the end of the first year. The novelty of this project is to establish an automated method to assess buildings in Amsterdam on their suitability for catering services, based on the compliance of their chimneys to minimum height restrictions.

2 Acknowledgments

As a group, we would express our gratitude to the people who have helped us throughout this project. First of all, we would like to thank Mr. Wietse Balster, representing the Municipality of Amsterdam, for providing input data and for the defining the desired output of this project. In addition, we would like to thank our two supervisors: Giorgio Agugiaro and Balázs Dukai for their useful and interesting ideas and general supervision which ensured we were able to complete this project with the most achievable results. Finally, we would like to thank Mathias Lemmens, Edward Verbree and Liangliang Nan for the organization of the Synthesis Project course.

3 Introduction

Although it is challenging to establish a direct connection between odors and health symptoms, odor nuisance is attributed to adverse reactions including headaches, sleep disturbances, inability to concentrate and stress-related issues. In many urban areas, odor nuisance is related to the presence of catering services (e.g. restaurants, clubs, takeaways) nearby or adjacent to housing facilities. In those cases, the local residents are mostly affected [12].

Regulations in the Netherlands on odor control, regarding the commercial kitchens and the catering services, have been established by the *Activiteitenregeling Milieubeheer*, Article 3.103, Clause 1 [15]. This clause states that the maximum tolerated odor nuisance from emissions due to the cooking activities should be based on a strict rule. Specifically, this rule defines that the height of the chimney is required to be at least two meters higher than the highest point of the roof of every building present in a radius of 25 meters.

However, the municipality of Amsterdam is not allowed to apply this regulation on the historical buildings of Amsterdam due to aesthetic reasons. As a result, people willing to set up catering services need to install deodorizing systems which leads to extra costs. Also, the responsible inspection office needs to control the appliance of this regulation by doing in-situ measurements, which costs time and requires more staff.

To overcome this problem, the municipality of Amsterdam is interested in a tool for both spatial programmers and citizens that visualizes all the buildings of the 3D city model of Amsterdam and provides information on their suitability for catering purposes. The suitability requirements are based on the chimney heights and the buildings' heights in the surrounding area. In an attempt to unveil how this tool could perform reliable results, this research aims to detect chimneys from point clouds and in a next step to detect buildings with appropriate chimney heights in the city of Amsterdam.

The report is structured as follows. First a description of the problem is provided, together with the research questions and related scientific work. Hereafter, the methodology for creating the 3D model is provided. Next, the methods applied for the chimney detection together with the methods on how the quality of the algorithm is assessed are described. Hereafter, the results are presented followed by an extended discussion. The report finishes with conclusions and recommendations.

4 Problem description

4.1 Background information

The main underlying problem to this project is that currently it is complicated for the municipality of Amsterdam to execute the chimney regulations for restaurants in the city. To minimize the influence of the smell, chimneys need to comply to a minimum height restriction. This restriction is based on the height of surrounding buildings in a certain radius. If the chimney is not compliant, an expensive filter needs to be installed. Currently the height of the chimneys is measured by inspectors on-site, which results in inconsistent and biased measurements and time consuming practices. Therefore, a faster, automated and possibly scalable method is required.

To fulfill the requirements for the tool of the municipality of Amsterdam, two different sub-tasks are specified:

- Creating a 3D model which can be easily accessed by the customer and enables an easy handling for retrieving the chimney heights and compliance information to the regulations.
- Development of an algorithm that can detect the chimneys with their height automatically from point clouds.

4.2 Study area

Even though the resulting algorithm for object detection and the more refined chimney detection can be applied to any urban area, the focus of this project is on the west side of the centre of the city of Amsterdam. This area is selected to fulfill the needs of the client, who is interested in the Nieuwmarkt. Besides, the area is interesting due to the high amount of catering services and monumental buildings (for which installation of new building structures should be avoided, as it might conflict with regulations), as seen in figure 2. The extent of the study area was fixed by creating a polygon in QGIS. The extent and a visualization of the polygon defining the research area are given in table 4.2 and figure 1.

	X-coordinate	Y-coordinate
UL	119825.483905	488220.711820
LL	119787.402886	485704.644437
UR	122208.267719	488226.151967
LR	122194.667355	485682.883855

Table 1: Coordinates of extent of research area in EPSG:28992. UL = Upper left, LL = Lower left, UR = Upper right, LR = Lower right.

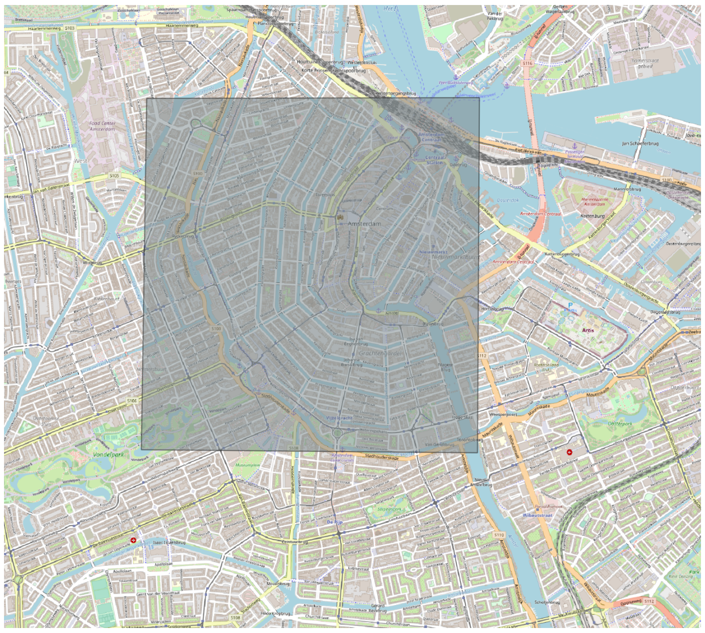


Figure 1: The research area

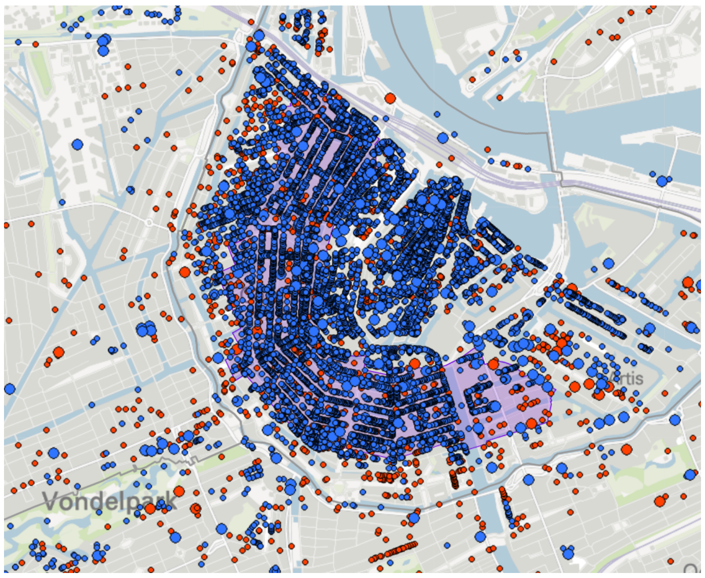


Figure 2: Most national (blue dots) and municipal (red dots) monuments fall within our research area. Retrieved from [3]

4.3 Research questions

When looking at the problem from a scientific point of view the question can be splitted in multiple sub-questions:

- Is it possible to detect small structures on top of roofs with point clouds made from aerial images?
- Is it possible to distinguish between chimneys, other structures on the roof or just outliers from the point cloud?

- If a detection and distinction is possible, what is the quality of the applied methods?

5 Related work

One of the largest challenges to this project is the world-wide lack of scientific knowledge on how to detect small objects on top of roofs. Chimney detection, or in general, detection of small roof structures from airborne point clouds (LIDAR and dense stereo image matching), is hardly researched. Often the quality of the point cloud does not allow for a clear representation of these structures and therefore makes the detection of small objects very challenging. However, research is executed on approaches for classification, segmentation and reconstruction of roof structures in urban areas. Some of these researches have looked into small roof structures. In these researches, small roof structures are generally handled in two different ways:

- Before implementing certain approach, a noise filter is applied to get rid of these structures (i.e. Jochem et al [9] use a threshold in their segmentation to filter out chimneys)
- It is stated that the approach is influenced by the small roof structures - and sometimes they are detected. However, these structures are only side-products of their research with less reliable reconstruction and are not explained in further detail (i.e. Vosselmann [8], where all small roof structures like chimneys and windows are treated as segments)

Most approaches are based on LIDAR point clouds in contrast to point clouds created by stereo image matching - as mostly used in this research. These point clouds are less noisy and therefore more applicable for many algorithms that use the similarity, or the normals, of adjacent points for identifying roof points (i. e. research by Caixia et al. in 2019 [23]). In the research of this project LIDAR data could often not be used as too less points where present to represent a roof object. Nevertheless, the approaches applied by different researches are worth considering as they deliver important notes and ideas for the development of own methods.

In research executed by Vosselmann in 2013 [8] different segmentation methods are applied and post-processing methods are executed for detecting connected points and extracting features. Detection of small features was not the main purpose of the research, even though the results show some of these features. Jochem et al (2009) [9] used the surface normals to detect planar segments in roofs. Their methods allowed to detect some small structures, but these were rejected as disturbance factors. A different approach is selected by Peters et al. (2015) [16]. They use the medial axis transform and shrinking ball algorithm to perform a visibility analysis for point clouds. Some small objects with spare points could be detected.

It can be expected that in the future the quality of the point clouds will increase, resulting in new possibilities for small object detection. Consequently more research should be expected on this topic.

6 Methodology

6.1 3D model

A 3D city model was created for this project with two purposes. Firstly, the model allows to make the results on object and chimney detection interactive and easily usable for the client. Secondly, generation of an enriched 3D city model was a requirement set by the supervisors in order to generate an input model for a course given at the Technical University of Delft. It was decided to first create a 'base' Level of Detail (LoD)1 model, which is later enriched with information from other datasets. The process of generation of the final model can thus be divided into two steps; creation of the 3D city model and the enrichment of the created model.

In the following paragraphs first the data needed for creation of the city model and the model enrichment will be presented. Hereafter, the methodology on creation of the geometry of the model is described, followed by the methods applied to include the semantics in the model.

6.1.1 Data sources

In order to generate the LoD1 3D city model, 2D shapes and locations of the buildings are needed, together with height information that allows for extrusion of the building shapes. For the building shapes, the Basic Registration of Large-Scale Topography (BGT) dataset is used which contains large scale topology information for the Netherlands. For the height information the Actueel Hoogtebestand Nederland (AHN3) dataset is used, which contains classified LIDAR points for the whole of the Netherlands. All this data is downloaded from the Publieke Dienstverlening Op de Kaart (PDOK) portal [1], which is a geo-portal for The Netherlands. The extent of the research project was based on a BGT tile of a part of Amsterdam, as the polygon defining the study area was drawn around the tile, including a small buffer which was necessary for matching height points to buildings later-on in the project. Consequently, no pre-processing was needed for the BGT tile extracted from the geoportal. For the AHN3 dataset four tiles where extracted and clipped based on the polygon representing the research area.

Moreover, the client provided a LoD2 model of the city of Amsterdam. This model contained a better representation of the roof surfaces - potentially useful for detecting points above the roof. It was delivered as a DGN file, a proprietary format supported by Bentley Systems, MicroStation and IGDS CAD programs. Reading and translating the file properly was not possible, leading it to be dismissed. This will be discussed with more detail in the subsection below.

For the enrichment of the 3D model several datasets were considered, including the Waarde Ontroerende Zaken (WOZ), Nationaal Handelsregister (NHR) and Basisregistratie Adressen en Gebouwen (BAG). The WOZ contains information on the actual function of buildings. The National Handelsregister discloses the names of establishments – which could give a hint on the type of catering services housed in a building. However, the WOZ data can only be retrieved for individual buildings and there is a restriction on the number of searches per time period – making it impossible to retrieve data for larger areas. Furthermore, although the NHR was found online, the data could only be retrieved in an anonymised manner, making it unviable for linkage with the 3D model. Searching for alternatives to these datasets led to the data portal of Amsterdam, where data was found containing all restaurant and café names and addresses in the city area [5]. Unfortunately, this data was outdated: the last update was in 2017. Therefore it was decided not to use the data in the final model. Nevertheless, the work-flow for integrating this data in the model to determine the *bldg:usage* as 'restaurant' is present in the FME workspace. Furthermore, data on energy labels was retrieved to be included in the model [4], next to the main source of information; the BAG dataset. The BAG contains all information on the addresses of buildings and their units – street name, street type, floor areas, geometries, year of construction, valid system time of the units (if they still exist or not), functions, among others. The dataset was obtained as a PostGIS dump, created by NLExtract [14] and based on the latest releases of the BAG documents as made available by Kadaster. The file was restored as a PostGIS table, which could then be easily queried using SQL. The release date of the file was 8 april 2019; the most recent version available at the start of the project.

6.1.2 Geometry

In order to generate the 3D city model, 3dfier was used. This software was developed by the Technical University of Delft and allows for creation of simple LoD1 3D models out of 2D GIS data by lifting polygons to corresponding height values delivered by a point cloud dataset (i.e. LIDAR data). Different lifting methods are specified based on the semantics of the polygons [18]. As input the BGT tile is used together with AHN3 data, which are described above. Code is provided on the Github page of 3dfier which allows for preparation of BGT data. It removes both buildings that no longer exist and CurvePolygons, to generate suitable input files for 3dfier [19]. 3dfier requires a configuration file in which references are provided to input files and parameter values are specified. It was decided to use the default values provided in the standard configuration document where possible and to alter some parameters via trial and error, optimizing the output results. Different versions of the model where stored and information on these versions was reported in a table 6.1.3. The output files where CityJSON files so that they could be visualized using the CityJSON viewer

[1]https://www.pdok.nl/

Version	Information	Problems
1	Default settings	Large spikes in almost all classes
2	filter_outliers	Large spikes in almost all classes
3	filter_outliers and flatten roads	Large spikes in almost all classes
4	Input AHN3 data is cropped to remove all points below zero. Default settings are used.	Large spikes in almost all classes
5	radius_vertex_elevation = 2.0 (instead of 1.0)	Still many spikes, but less.
6	radius_vertex_elevation = 3.0	Still many spikes, but less.
7	radius_vertex_elevation = 6.0	Still some spikes, but a lot less.
8	radius_vertex_elevation = 6.0	No difference with version 7 so filtering outliers is not necessary
9	radius_vertex_elevation = 10.0	Only hand-full of very small spikes left
10	radius_vertex_elevation = 15.0	One big spike pointing upwards and a couple of small spikes.

Table 2: Versions of 3D models generated through 3Dfier.

[21]. The selected final model was generated in CityGML. The documentation to the parameters necessary in the configuration document are provided on the 3dfier github [20].

No objects were present in the research area belonging to the BGT class 'Ondersteunend Waterdeel'. Therefore, the path to this empty file was excluded from the configuration file. The default values on lifting options could be kept unaltered.

The issue that arose when using default values was the appearance of spikes (figure 3). Several attempts where executed to remove these spikes. It was believed that the spikes were either a result of outliers in the LIDAR data, or a consequence of no-data values. Firstly, the explanation based on outliers was examined. In the configuration file, for every BGT class where spikes where present, the parameter 'filter_outliers' was set to true. In addition, there was experimented with cropping the AHN3 data by removing all the points with a z value below zero, to check if the issue was indeed a result of outliers. However, the spikes still remained and therefore it was believed that the problem did not lie with outliers.

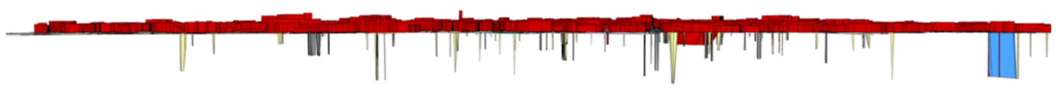
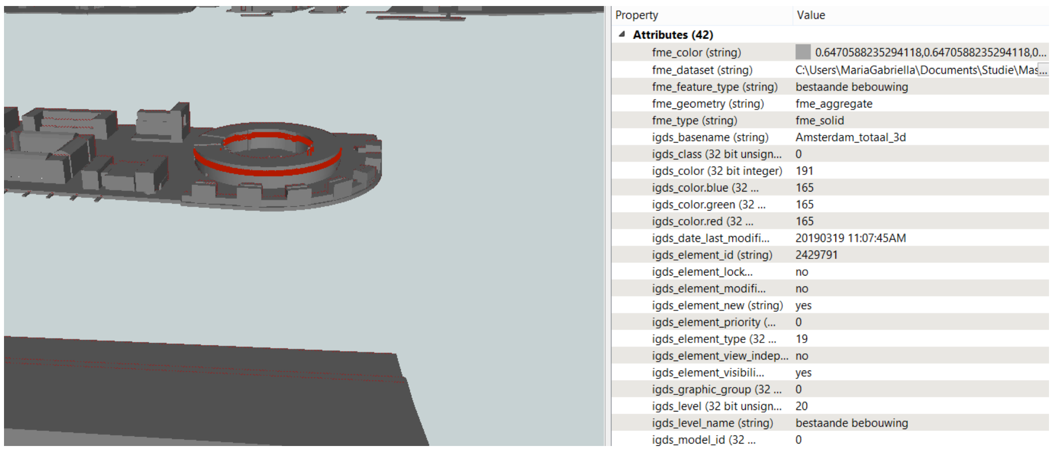


Figure 3: A cross section of 3D model version 2. Spikes are present for multiple classes.

Consequently, the next step was to examine if the spikes where a result of no-data values. A relevant parameter for no-data values is 'radius_vertex_elevation'. This parameter specifies the radius used for the distances between the 3D points and the vertices of polygons, in meters. If this radius is large, more points will be considered when extruding a polygon. However, the larger the radius, the further away of the polygon these points can possibly be and therefore, the smaller the chance that the points used all correspond to the polygon. When the parameter has a small value, less points will be considered, allowing the possibility of no points being considered at all. Consequently, the polygon will be elevated to the default no-data elevation value set for that class. Enlarging the 'radius_vertex_elevation' from the default value of 1 metre to 2, 3, 6, 10 and 15 resulted in less spikes. The reduction of the amount of spikes after enlargement of the 'radius_vertex_elevation' indicated that the spikes could be explained by no-data values. The higher the value, the less spikes were present. In order to balance between the amount of spikes and a justifiable search radius it was decided to select model version 9, with a search radius of 10m, as the final model. It should be noted that a the search radius used for finding points belonging to buildings is a separate value. Therefore the relatively high search radius selected for the other classes does not influence the quality of building extrusion.

After creating the LOD1 model, the DGN file containing the model of Amsterdam was considered as an additional resource for providing LOD2 geometries. This process would have taken place through the FME workbench - as with the semantic enrichment -, allowing proper translation of features to the CityGML format. However, it was not possible to extract the geometries in an ordered manner from the file. Firstly, the layers that could be accessed using the FME readers seemed quite unorganized with regards to their names: some were related to building qualities ('nieuwbouw', 'nieuwe bebouwing', 'bestaande bouw') while others concerned locations ('Ijburg Middeneiland', 'Dreven'). Secondly, there were many different geometries present in these layers: around 130.000 lines, 1000 arcs, 300.000 polygons and 26.000 solids - besides another 330.000 features of different types. And although it is possible to filter these geometries, their properties together with the structure of the file indicate the absence of proper linkage of geometries to their actual buildings. This can be observed in figure 4. It means there is not a way of knowing which surfaces/polygons belong to which buildings - and which are part of the roof. This problem could be a consequence of export methods used for generating the file, or it can be due to limitations of the FME readers. Regardless, these challenges rendered the LOD2 model unfit for use.



Property	Value
Attributes (42)	
fme_color (string)	▨ 0.6470588235294118,0.6470588235294118,0...
fme_dataset (string)	C:\Users\MariaGabriella\Documents\Studie\Mas...
fme_feature_type (string)	bestaande bebouwing
fme_geometry (string)	fme_aggregate
fme_type (string)	fme_solid
igds_basename (string)	Amsterdam_totaal_3d
igds_class (32 bit unsign...	0
igds_color (32 bit integer)	191
igds_color.blue (32 ...	165
igds_color.green (32 ...	165
igds_color.red (32 ...	165
igds_date_last_modifi...	20190319 11:07:45AM
igds_element_id (string)	2429791
igds_element_lock...	no
igds_element_modifi...	no
igds_element_new (string)	yes
igds_element_priority (...	0
igds_element_type (32 ...	19
igds_element_view_indep...	no
igds_element_visibili...	yes
igds_graphic_group (32 ...	0
igds_level (32 bit unsign...	20
igds_level_name (string)	bestaande bebouwing
igds_model_id (32 ...	0

Figure 4: Contents of the DGN file: it was only possible to visualize what seemed to be LOD1 solids. The different parts of a building (as highlighted in red) do not seem to have any attribute linking them to other building parts.

6.1.3 Semantics

After the geometric part of the model was finalized, the focus was set on the semantics and therefore the enrichment of the model. Before discussing the workflow for integrating the BAG data with the existing CityGML model, some limitations of the data should be considered. The most common errors and problematic attributes are discussed in the Kadaster's 'quality dashboard' [10]. Some of these situations deserve special attention, as they occur within the study area or could potentially affect the research results;

- **Unlikely building year:** although the history of Amsterdam dates back to the middle ages and the study area in particular contains many of its older buildings, it is improbable that there are structures present from before the 13th century. When querying for this, the only deviation was that of building years listed as '1005'. According to Gemeente Amsterdam [2] this is actually a special code given to buildings in the historic area of the city, for which the year of construction is unknown and does not require further research. It is important to be aware of this when passing the information on to the CityGML model, as this code must be interpreted properly in the final visualization.

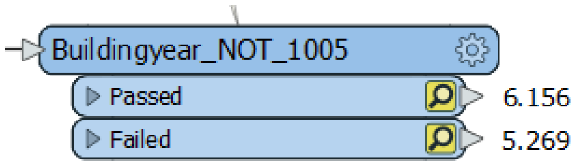


Figure 5: There are over 5000 buildings in the study area with unknown building year - this is logical, as the central area of the city contains most of the older buildings

- **Unlikely floor area:** When the floor area is unknown, the values: *1, 99, 999, 9999, 99999 and 999999* are given instead. As the floor area information is used to compute the storey number/height, it is important that these values are filtered to prevent the propagation of errors. However, when typos are made it becomes more difficult to eliminate the problem. These cases are handled directly in FME: if the calculated storey height (based on the proper building area and the height retrieved from the point cloud) is too low to be realistic, the floor area is most likely wrong – and consequently disregarded in the output model.

  After the semantic enrichment mentioned above, the final step was to link the findings of the object detection and the verification of compliance to the model. This was accomplished by merging the output file of the verification with the building instances in the model, based on their BAG IDs. The file contains: a point for each object, the transformation parameters used to construct the geometry of the object, an attribute for passing/failing the validation test, and the BAG ID of the building each object relates to.

When extracting information from the BAG sub-queries were used for clarity, as there were many linkages present between different tables and attributes that needed to be understood beforehand. These intermediate sub-queries were stored as materialized views, and eventually exported as CSV. This format can be easily read into FME, which was used for the integration of the data with the CityGML model. The model itself already contained a link between the BGT features and BAG id's, making the process more straightforward – it was stored in a generic attribute 'identificatieBAGPND', as specified by the dutch information model IMGeo. Two steps deserve further consideration:

- **Different address representations:** the house suffix is composed of two columns, namely 'huisletter' and 'huistoevoeging'. These columns are used interchangeably: sometimes the object contains both letters and suffix, other times only one of the columns is used. The distinction between both is not always clear, but the suffix allows alphanumeric characters, which makes it more suitable for different situations. For all the objects in Amsterdam, about 34% makes use of house suffixes, while only 17% uses house letters. In the CityGML model, both will be considered as 'ThoroughfareNumberSuffix' – as the maxOccurs of this element is 'unbounded', both attributes can be modelled at the same time.

Postal code	**House number**	**House letter**	**House suffix**
1012WG	277	B	
1015DW	42		H
1016RA	1	A	2
1017SG	24		

Table 3: Addresses can have different elements, house letters and suffixes are used interchangeably

- **Handling wrong values:** when the year of construction is listed as the code '1005', bldg:yearOfConstruction is left empty, as to avoid confusion in the final model. Similarly for *bldg:storeysAboveGround* and *bldg:storeyHeightAboveGround*, the attributes are left empty if the value of 'floorArea' reflects a default code or if the area seems disproportional to the building dimensions. This last case can be checked by comparing the calculated storey height to a realistic threshold – for example, storey height smaller than 2 metres should not be considered in the output. A threshold for maximum storey height should be defined based on the functions of the buildings.

As mentioned before, three other sources were integrated in the model besides the BAG: a dataset with information on restaurant addresses in Amsterdam [5], energy label/quality data from the municipality [4], and a file containing all objects/chimneys that were detected - together with relevant indicators. The information contained in the first two datasets was merged with the features in the model by using the postal codes. The restaurant dataset was used to specify the usage of the building – in contrast to the 'function' that is extracted from the BAG. In this way, the 'usage' element can be used to filter out the attributes/geometries of buildings that are most relevant: those containing catering services. Of course, the dataset's low update frequency makes its use less ideal and reliable. Still, this type of information should be considered. A small fragment of the elements specified for a building entity are shown below, in Listing 1.

Listing 1: All elements found under bldg:Building

```
<core:externalReference>...</core:externalReference>
<bldg:function>winkelfunctie</bldg:function>
<bldg:function>woonfunctie</bldg:function>
<bldg:function>kantoorfunctie</bldg:function>
<bldg:usage>restaurant</bldg:usage>
<bldg:measuredHeight uom="m">22.08</bldg:measuredHeight>
<bldg:storeysAboveGround>4</bldg:storeysAboveGround>
<bldg:storeyHeightsAboveGround uom="m">5.52
</bldg:storeyHeightsAboveGround>
<bldg:lod0FootPrint>...</bldg:lod0FootPrint>
<bldg:lod0RoofEdge>...</bldg:lod0RoofEdge>
<bldg:lod1Solid>...</bldg:lod1Solid>
<bldg:address>...</bldg:address>
<energy:usageZone>...</energy:usageZone>
```

The dataset concerning energy certification, was incorporated through the energy Application Domain Extension (ADE) and specified for each *BuildingUnit* within a Building. Although not directly related to our application, it is not difficult to image that energy data might be of use for analyses regarding other 'above roof' structures – such as solar panels, and air vents. This dataset also contained house numbers and house suffix for each energy certificate documents. However, the house suffix column showed inconsistencies compared to those in the BAG (i.e. hyphenation, case-sensitive letters, and latin numerals), so this information was not used for the merging. As can be seen in Listing 2, the 'usageZoneType' was left empty. This had to be done because the available values were often ambiguous. Some units within BAG buildings have multiple functions listed (and thus, multiple usage zone types). Because of this, it is not possible to confirm to which type of usage zone the building unit belongs. This is a limitation of the data available, so no further actions could be taken. As a 'usageZoneType' value is mandatory, the energy extension could not be validated properly.

Lastly, the output file from the object detection was imported into the model. Due to the possibly large volume of objects and the simplified (rectangular) representation used for visualization, it was decided to initially model them as implicit geometries. However, due to software problems regarding the support of this type of object, the final *BuildingInstallations* were modelled explicitly. The results will be shown in the next section.

Listing 2: All elements found under energy:usageZone

```
<energy:usageZone>
    <energy:UsageZone gml:id="Uzone_73334">
      <energy:usageZoneType codeSpace=""/>
      <energy:contains>
        <energy:BuildingUnit gml:id="Unit_75853">
          <energy:floorArea>
            <energy:FloorArea>
              <energy:type>grossFloorArea</energy:type>
              <energy:value uom="m^2">116</energy:value>
            </energy:FloorArea>
          </energy:floorArea>
          <energy:energyPerformanceCertification>
            <energy:EnergyPerformanceCertification>
              <energy:rating>D</energy:rating>
              <energy:name/>
              <energy:certificationId>746461896</energy:certificationId>
            </energy:EnergyPerformanceCertification>
          </energy:energyPerformanceCertification>
        </energy:BuildingUnit>
      </energy:contains>
    </energy:UsageZone>
</energy:usageZone>
```

6.2 Object detection

6.2.1 Data sources

There are four data sources used for object detection; a point cloud generated from dense image matching (DIM), their respective orthophotos, the AHN3 point cloud and the BAG.

DIM aims at computing a depth value for every pixel of an image. Thus, all points in the DIM point cloud are regularly arranged along x- and y- axes like pixels in the image. However, there are matching problems in texture-less, occluded, overshadowed, and depth discontinuous areas, which affect the accuracy of height and planar values of the corresponding points [17]. The DIM point cloud, provided by the municipality of Amsterdam, contains 3D information of buildings and is used as the main source for roof detection and segmentation. The orthophotos underlying the DIM point cloud are used for extraction of ground truth data to be able to execute quality checks. Using the respective orthophotos ensures that all deviations occurring in the point cloud should in theory be visible in the images. To

As the AHN3 point cloud has a better height accuracy than the DIM point cloud, AHN3 data was used for reconstructing 3D roof surfaces as planes (see Figure 6). These planes were used for detection of objects above the roof. The building boundaries are represented by the BAG dataset.BAG is selected instead of BGT as the orthophotos are taken from a top-view. BGT defines ground level geometry as the boundary of a building where it touches the ground, while BAG defines the 2D boundary of a building by providing the footprint of the roof.

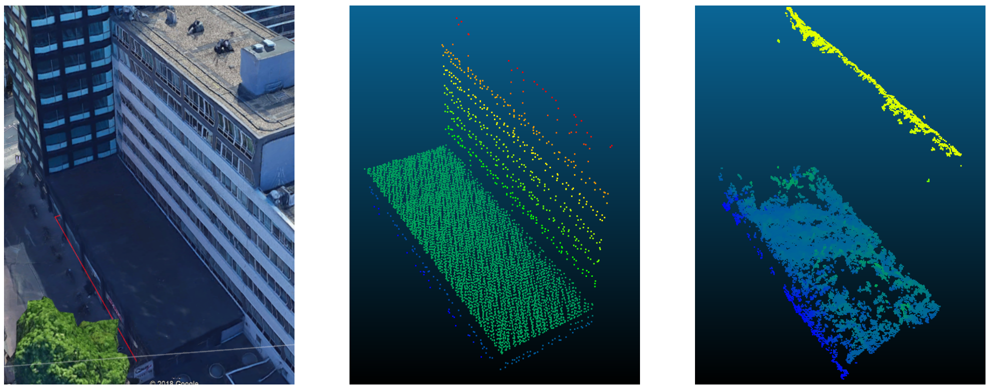


Figure 6: Quality comparison between AHN3 and DIM (point clouds are rendered by height)

Another important factor is the density difference between the two point cloud datasets. Samples were taken from the DIM and the AHN3 point clouds to compute the planar point density and average distance between points. The DIM point cloud is around two times more dense than the

AHN3 point cloud. When looking closer, the DIM point cloud's point density per squared meter for the study area can be as high as 110 points, compared to 10 points for the AHN3. The limited amount of points restricts the possibility to detect very small objects, as can be explained by the Nyquist–Shannon sampling theorem [7]. According to this theorem, only objects with dimensions twice as big as the sampling distance (lambda) will be reconstructed fully. The reason for this is that objects smaller than the sampling distance may be completely missed at times, as illustrated in 7. An object twice as big will have at least one point to represent it. However, it should be noted that this is not enough to properly classify points as objects/outliers - so very small objects will still suffer from these limitations.

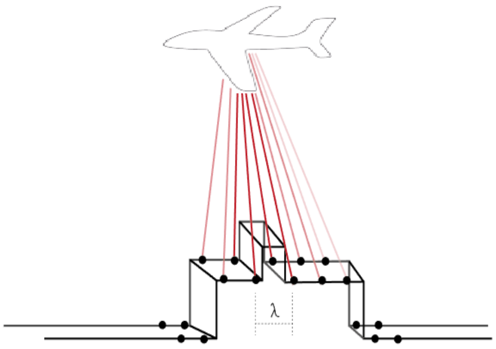


Figure 7: Illustration of relation between sampling distance (lambda) and object dimensions

The results from the table above indicate that the lidar point cloud is less appropriate to be used for the detection of small objects. With a distance among points of 24cm, only objects around 50cm would be detected. Nonetheless, the lower point density means there might not be enough points available to actually distinguish an object and its shape. Since the DIM point cloud was created from aerial photos with the ground sample distance (GSD) being 10cm, the distance among consecutive-nearby points in the DIM point cloud is ideally 10cm. However, the sampled point clouds from the table above indicate less optimistic results: an average distance between points of 18cm. Therefore, there is a chance that small objects-structures with dimensions less than 36cm have not been captured by the aerial photos and are not present in the point cloud. Still, the results from the DIM indicate that this point cloud could lead to better results.

In real-life, many gaps are present in the DIM point cloud (due to occlusions, shadowing), which enlarges the required minimum object size needed to be certain that the object is detected. This limitation has been acknowledged, but no alterations to the applied methodology could be implemented.

	Area size (km2)	Number of points	Point density (points/m2)	average planar distance (m)
DIM	0.25 x 0.25	5,559,162	88.95	0.11
AHN3	0.25 x 0.25	1,001,292	16.02	0.25

Table 4: statistics for point cloud quality.

6.2.2 BAG footprint point cloud separation

Detection of all square chimneys in an area can be realized by firstly detecting chimneys on a single building and applying the methods to all the buildings in the research area. Therefore, it was decided as a first step to split the DIM point cloud per building in the BAG dataset and to use the splitted point clouds for the chimney detection algorithm. This splitting was executed in FME (Figure 8).

The input data consisted of 110 DIM point cloud tiles and 15455 buildings/footprints from the BAG. First, the transformer "Clipper" was used to split all the DIM files into separated files per building. This tool traverses all the input point clouds and splits them one by one, according to the input footprints. Since some footprints are at the junction of several DIM tiles, there might be several separated files (2, 3, or 4) referring to the same footprint, which need to be merged into one. This was realised by the transformer "PointCloudCombiner", which organizes point clouds into groups by the attribute "pandid" and outputs for each group its own point cloud.

Due to the inconsistency in the extents of Dense Image Matching (DIM) and BAG, only 14392 separated files were generated. 1061 of the point clouds completely fall outside of the extent of DIM. Some footprints (173) were exactly at the boundary of the DIM, hence the corresponding point clouds only contained a part of the buildings.

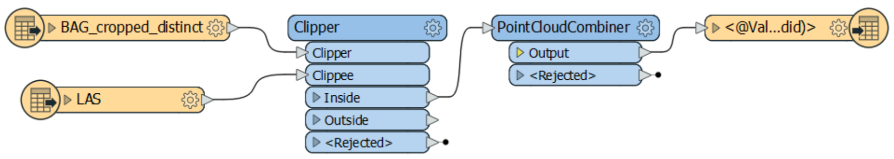


Figure 8: Separation workflow in FME

6.2.3 Segmentation and filtering

The next step comprises the segmentation of the DIM point cloud based on a region growing algorithm for detecting planes. As chimneys and other roof structures stick out, they disturb the planarity of the roof and are not part of the same segment.

Each segmentation starts with the selection of a randomly picked point. From this point the k-nearest neighbours are selected with a sample size of 20. These points are used for a plane estimation using RANSAC [22]. Afterwards points are detected that fit to the plane based on a threshold. For this paper the threshold is set to 0.4m, by trial and error (points that are up to 40 cm over or under the plane are considered as part of the plane). This threshold is set as it compensate to the low quality (bumpiness) of the input point cloud but it still allows for the detection of roof structures. If the number of points of a plane is increased by at least 30%, the plane is re-estimated. A potential plane must have a least 150 points to be considered as plane. With this approach the planes are segmented and structures not belonging to the roof are set to unclassified (Figure 9).

Afterwards all points with a segment ID were removed so that only the unclassified points are left. These points are reclassified based on their euclidean distance to each other. If points were closer than 50 cm they were considered to belong to the same class. All classes with less than 25 points were then removed as well, as they are outliers or leftovers from edges of the roof and are not needed for identification (see Figure 10).

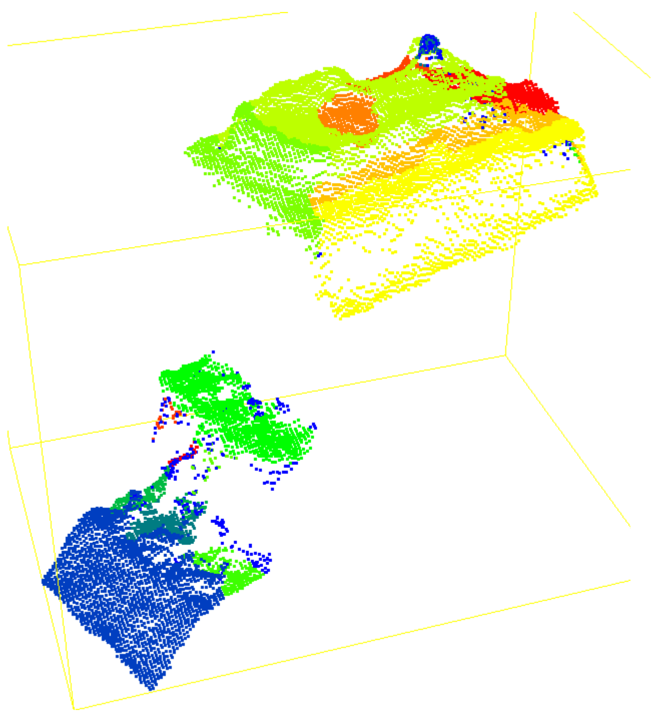


Figure 9: Segmented point cloud. The unsegmented parts are displayed in blue (except the big structure at the bottom left).

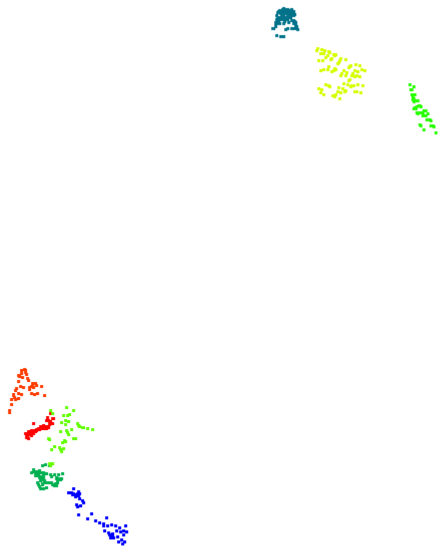


Figure 10: The filtered and clustered point cloud

6.2.4 Roof reconstruction and wall detection

The next step included the detection and removal of roofs and walls from the point clouds. The basic idea is that during the segmentation, some planar segments containing more than 150 points were detected. It is assumed that these segments represented roofs or walls, as objects on top of the roof will not be represented by such a large amount of points. Therefore, based on the segmentation results, roof detection was carried out.

Firstly, the wall segments were removed from the point cloud. For all the segments which contained more than 150 points, the normals were calculated. The length of the estimated normal vector is 1, if the angle between this vector and horizontal plane is smaller than 20 degrees, the corresponding segment is considered as too steep it is likely that it represents a wall. Therefore, these segments were removed.

With the remaining segments and with AHN3 data, roof detection and reconstruction was executed. The reconstructed roofs were used later on for removing objects that are located under the roofs. Whenever possible the AHN3 point cloud was used as input for the roof reconstruction, as AHN3 can better represent flat surfaces than the DIM point cloud. The AHN3 point cloud was segmented. Instead of using segments which contained more than 150 points for roof reconstruction, segments where selected that contained more than 100 points. This was done as the AHN3 is less dense than the DIM point cloud. If a building did not exist in the AHN3 point cloud, it was decided to use the DIM point cloud for roof detection. In the following paragraphs the roof detection procedure will be described in more detail.

The goal of the roof detection was to reconstruct roof surfaces from point clouds per building (BAG footprint). One of the most common reconstruction methods is Poisson reconstruction introduced in Kazhdan et al.[13], which generates smooth surface models by extracting ISO-surfaces from a vector field. However, the output model of this approach consists of many triangles. For this research, this is not a desired output as it complicates the performance of the next step; identification of structures above roof. The model of the roofs should be as concise as possible, i.e., containing several planar surfaces represented by a set of vertices. Nan et al. [11] have proposed a polygonal surface reconstruction method which aims at a manifold and watertight polygonal surface model from a point cloud. But it requires the model to be scanned completely, while point clouds from DIM have limited information on the facades and the bottom surfaces. Considering the characteristics of a DIM point cloud, it was decided only to reconstruct roof surfaces and neglect facades and small structures on the roof. Because, in practice, reconstruction of the facades is not necessary as only roofs are going to be used for the filtering of objects above them.

First an infinite plane was estimated for a segment. Hereafter, this plane was clipped to the extent of the segment. Principal components analysis (Principal Component Analysis (PCA)) was used for the estimation of the plane. As for the extent for clipping, firstly a convex hull was generated. However, some roofs were found in the point clouds which could not be described properly by its convex hull. In example T-shaped roofs will become triangles in the output model. Therefore, the concave hull was used to describe the shape of the roof.

The concave hull of a set of 2D points was computed in three steps:

1) **Building the Delaunay trianglation (DT)**: There is an existing python library "scipy" that can be used to generate DT. Since the coordinates of DIM are quite big, the library didn't function well at first: many points were unreasonably missing and only a few of them were left to form the DT. Thus, the input points were then centralized (i.e. shifted to the centroid). Hereafter, the points were all included in the DT.

2) **Creating the alpha shape**: An alpha shape is the DT with big triangles removed of which the radius of the circumcircle is bigger than the threshold 'alpha'. For AHN3, of which the average distance between each pair of points is approximately 0.4 meters, the threshold was set to 0.7 meters. This threshold was selected by executing several experiments. With respect to the DIM point cloud, considering its horizontal resolution is 10 centimeters, the alpha parameter was set to 0.2 meters. These parameters ensure that the basic triangles (i.e., an equilateral right triangle with the radius about 7 centimeters) will be kept as well as relatively larger triangles which make the alpha shape more smooth (see Figure 11), i.e., less vertices for representing the boundary. It will be further explained in step 3).

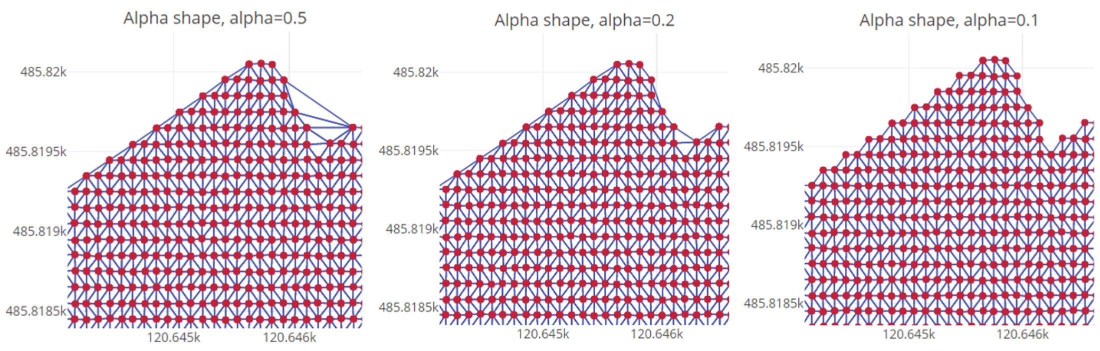


Figure 11: Alpha shapes with different parameters

3) **Extracting boundary**: The alpha shape is simply a set of triangles. Boundary edges are only stored once among these triangles, while other edges are stored twice, as they are shared between two triangles. To select all the boundary edges, this knowledge was used, as all the duplicate edges were removed and only boundary edges were retained. However, it should be noted that these boundary edges might include inner rings and separated rings (see Figure 12).

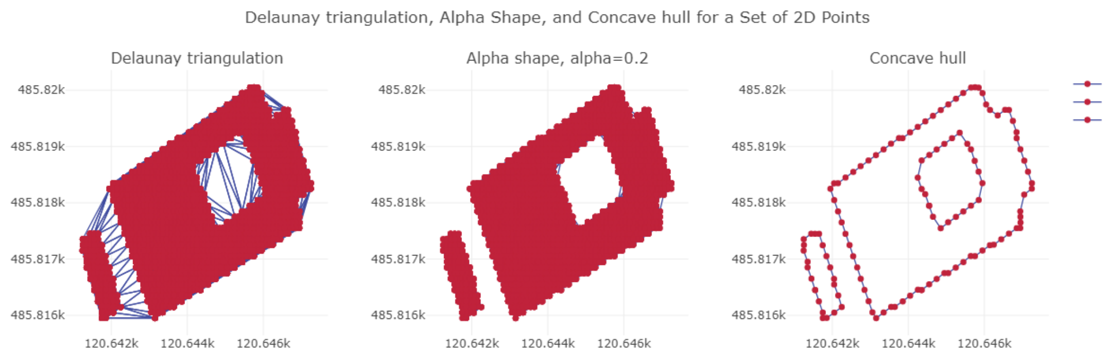


Figure 12: Potential inner rings and separate rings

The boundary edges are not ordered and need to be sorted to form a loop so it can be stored as a valid geometry. It was done by randomly selecting an edge and finding the next one to be connected. Once no next edge was found, the existing loop was returned. As is shown in the figures above, the number of vertices of a boundary can be large due to the short distance between points, so that boundaries can be simplified. The criterion for simplification was that the angle between adjacent edges should be no more than 160 degrees. If the angle exceeded the threshold, the corresponding vertex was removed and the two adjacent vertices were connected (figure 13).

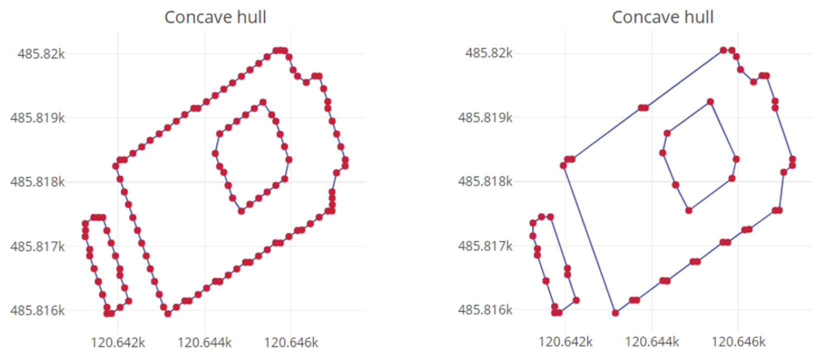


Figure 13: Simplification of the concave hull

For the convenience of the computation, only the longest outer ring was used among all returned rings. Inner rings and separated rings were removed.

6.2.5 Identification of structures above roof

As mentioned in the previous section, roof surfaces were constructed based on the point clouds. For every BAG ID (i.e. polygon with buildings), one obj file was exported containing all the possible roof surfaces, which are created by fitting planes through the segmented points. Moreover, the segmented point cloud, which had been cropped in separate ply files based on the BAG footprints, was “cleaned” from points on the walls and roofs.

In this part, the roof surfaces and the cleaned point cloud corresponding to every BAG footprint were used as input datasets. The reconstructed roof surfaces were used to detect all the points/segments that were located above them.

The algorithm takes as input values two different datasets: the roof surfaces (obj files) and the segmented point cloud (ply) for every BAG ID footprint. It retrieves the coordinates of the faces that exist in each obj file and stores them in a list. Then, it stores the coordinates and the segment IDs of the segmented point clouds.

The next step was to create 2D polygons of the faces (roofs surfaces) and to check and store all 2D points of the point cloud that are contained in the corresponding face (using Shapely library). By applying this method, the algorithm filters out the points corresponding to a specific face in 2D space and then searches which of them are above or below the roof surface in 3D space.

Afterwards, a plane is fitted on each roof surface in order to retrieve its centroid coordinates, its normal vector (i.e (A,B,C)) and the D parameter of the plane equation (i.e '$D = -Ax - By - Cz$'). All the parameters of the planes were stored in a dictionary.

All the segmented points, that corresponded to a specific roof surface, are projected on the roof surface and a new z value is calculated for each point. The height difference between the original 3D point and its projected point is stored as an extra attribute of this point. This will be used latter on for chimney detection described in section 6.2.6.

In the end, the algorithm returns all the points that are above the roof surfaces in a ply file. This file contains for every point; (x, y, z) coordinates, the segment ID, the height difference and the face ID of their corresponding roof surface.

The above procedure is executed for all the BAG footprints of the study area.

In Figure 14, the flowchart presents the order of the aforementioned steps. In figures 16 and 17, the segmented points are visualized before and after the implementation of the algorithm. The algorithm detects the actual points above the roofs and stores them in an output ply file.

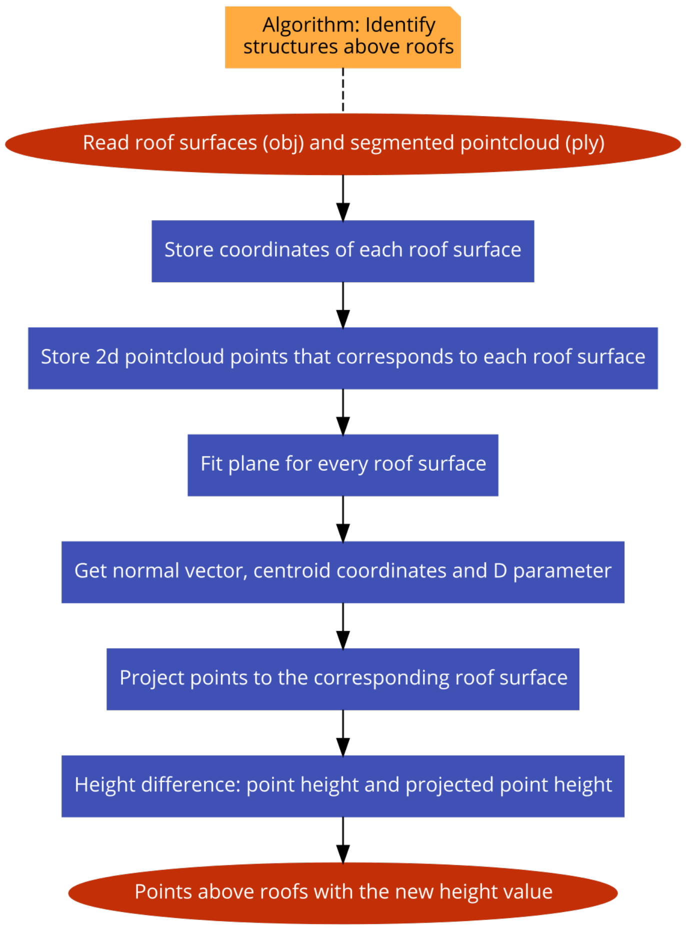

Algorithm: Identify structures above roofs
Read roof surfaces (obj) and segmented pointcloud (ply)
Store coordinates of each roof surface
Store 2d pointcloud points that corresponds to each roof surface
Fit plane for every roof surface
Get normal vector, centroid coordinates and D parameter
Project points to the corresponding roof surface
Height difference: point height and projected point height
Points above roofs with the new height value

Figure 14: Flowchart above roofs structures

In figure 15, the idea of the algorithm is presented. The segmented points that possibly define a small structure on the roof are projected on the corresponding plane.

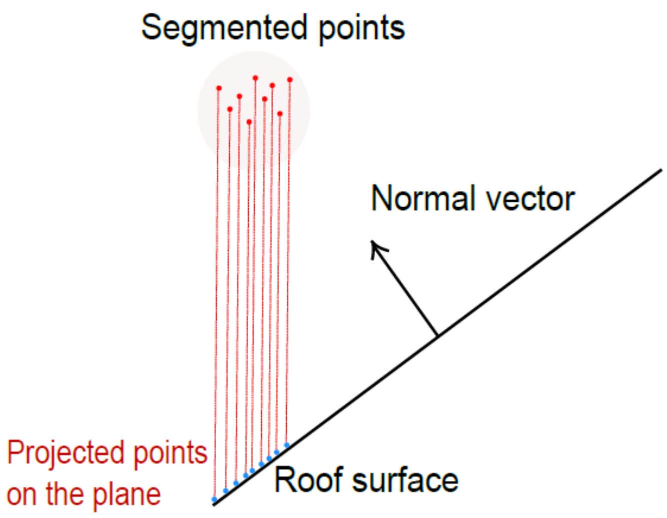


Figure 15: Projected points on the roof plane

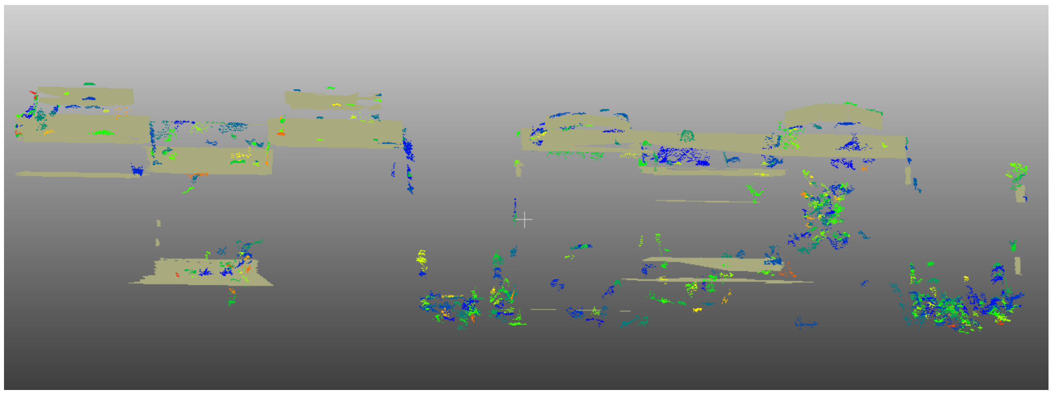


Figure 16: Before: Roof surfaces and segmented points

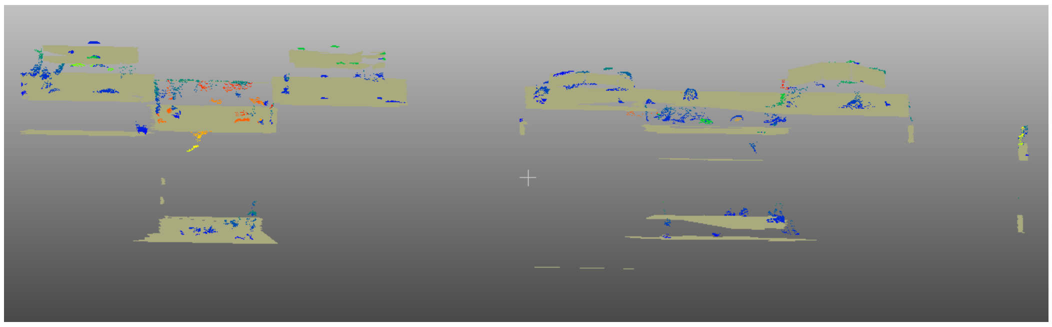


Figure 17: After: Roof surfaces and points above them

In some cases, the roofs surfaces overlapped each other. This happens as big flat structures above the roofs are segmented and reconstructed as separate roof surfaces. Consequently, some segments corresponded to multiple roof surfaces. In order to select only the points that are above the roof as a whole, the surface with the highest elevation was selected to be the corresponding surface to a segment, on which the points were be projected.

Figures 18, 19, 20, 21 visualize the specific case of overlapping surfaces and how the algorithm detects only the required ones. In figures 19 and 21, the color scale indicates the height difference of each point from its corresponding roof surface. The red color displays the highest point, whereas the blue color shows the lowest point.

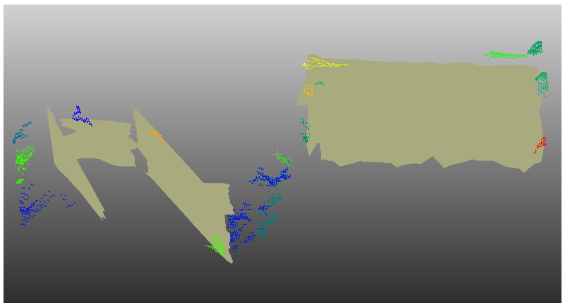


Figure 18: Segmented points above 2 overlapping roof surfaces

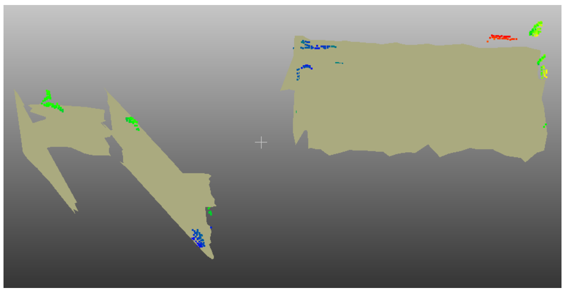


Figure 19: Only points above roofs

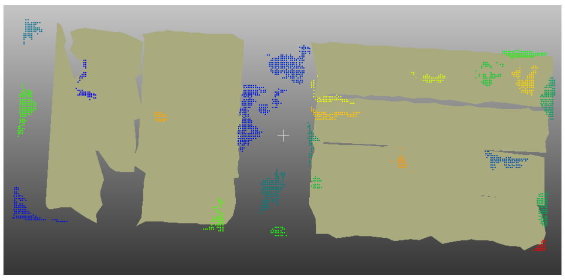


Figure 20: Segmented points above 2 overlapping roof surfaces

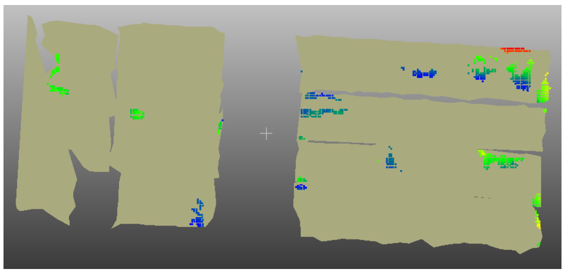


Figure 21: Only points above roofs

6.2.6 Square chimney detection

Chimneys exist in many different shapes and sizes. For this project it was decided to focus only on square brick chimneys, as it is believed this type of chimney is most easily distinguished from other objects. All available segments over the roof are filtered out using different criteria, with the goal to have only possible square chimneys left over. In order to distinguish between other objects and square chimney, the minimum and maximum X, Y, Z position of all points of a segment were determined and the span for each dimension was calculated. Furthermore the number of points per segment and the distance of the points to the roof (not the same as Z, which is the absolute height) was considered.

A chimney is defined as a square-like structure with dimensions of maximum 1m and a maximum height of 2m. Even though this will exclude some very large chimneys, it ensured that the majority of false segments could be excluded. The following criteria were used for selecting square chimneys out of all segments:

- For every dimension, it is checked whether the span is larger than 1 meter. For x and y direction this criterion can remove objects that are too large for being square chimneys (for example solar panels or lifted parts of the roof). In z direction it is able to discard oblique segments (for example, leftovers of oblique roofs).
- The span of x is divided by the span of y to get a ratio between x and y. If the ratio is lower than 0.9 or higher than 1.1, it is considered not to be a chimney. With this approach long

and thin structures can be detected, as the result will be very high or very low. A result of 1 represents a complete square segment. As the span is determined with the bounding box this approach cannot detect oblique long and thin segments (as seen in figure 22), as the ratio of their span of x and y is close to 1. An additional filtering step is needed.

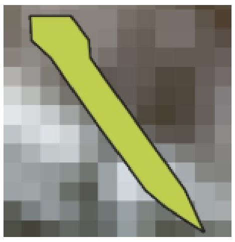


Figure 22: Oblique feature

- To solve the problems of oblique segments the correlation between x and y is calculated. Every segment with a correlation higher than 0.9 or smaller than -0.9 is excluded, as in this case the points are distributed similar to figure 22.
- The distance of the segment's centroid to the roof is calculated. Every segment with a distance bigger than 2 meter from the roof is removed. This method allows to eliminate the segments which correspond to trees located over the roofs.

All the thresholds used for the criteria were set carefully using trial and error. It is believed that the selected thresholds are reasonable, however, it is realized that small differences in the thresholds can lead to very different outcomes.

6.2.7 Compliance analysis

In order to check whether a building complies with the set chimney regulations, the height of its chimney(s) and the highest point of its surrounding buildings are needed. As the filtering of square chimneys out of all objects detected was not successful (see results), it was decided to implement the compliance check for all the objects detected on top of the roofs. The client can then check all detected objects for a building, which objects are compliant to the regulations, and whether they are predicted to be a chimney.

The object's height can be acquired from the chimney detection part (section 6.2.6), by finding the highest point of a segment. The heights of a building can be derived from roof detection (section 6.2.4) by finding the highest vertex of its roof polygons. For the purpose of convenience, the heights of all buildings have been calculated and saved in a JSON file (dictionary format) beforehand, which contains BAG IDs (key) and building heights (value).

The safety compliance analysis for one object can be divided into two parts as illustrated below, and then this procedure can be applied for all the detected objects.

1) **Finding buildings inside the 25-meter circle of which the center is the centroid of the object** (see Figure 23). This was done by calculating the distance between the center and all the surrounding buildings. If the distance is less than 25 meters, then it is considered inside the radius. Also, buildings that are not completely inside the radius are included. This means that, although the highest point of one building may be outside the circle, it is still taken into account. This is a limitation of the approach. All the buildings that were considered to be inside the circle were stored in a list of BAG IDs.

2) **Comparing the chimney height to the building height.** The BAG IDs of the buildings present in the search radius of 25 meters were used as keys to get the corresponding height values in the pre-made JSON file. If any of these heights is bigger than the chimney height

minus 2, this chimney is considered non-compliant with the regulations. Otherwise, it complies.

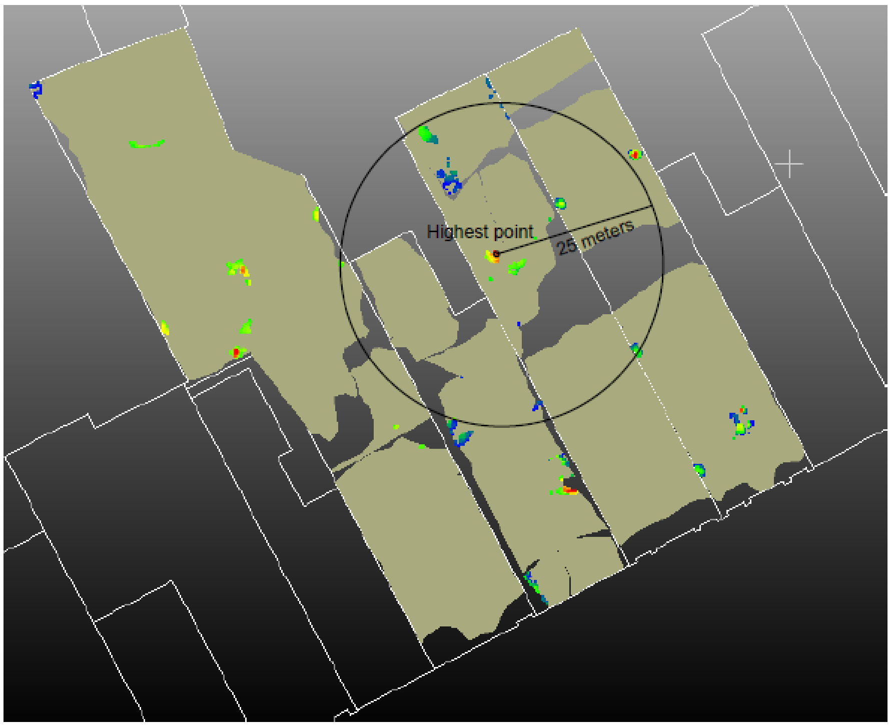


Figure 23: Highest segment point in a 25 meters radius

6.2.8 Quality assessment

In order to check the quality of the methods for object-chimney detection, results need to be compared to ground truth data. Both the results generated by the algorithm detecting objects above the roof, as the algorithm detecting chimneys were assessed. This was done in three steps:

1. Collection of ground truth data
2. Conversion of 3D point segments to 2D polygons
3. Comparison between ground truth and the converted 2D polygons

Ground truth data

As no ground truth data exist that contain shapes and locations of objects and square brick chimneys on top of roofs in Amsterdam, the research team had to generate ground truth data manually. It was aimed to collect all the objects on top of the roof for at least 5% of all the (BAG) buildings in the research area. The aerial imagery underlying the DIM point cloud was clipped to the research area and hereafter divided into 110 tiles.

Five tiles where selected randomly and divided among the research team members (fig. 24). Every researcher manually drew all the required objects on top of all the buildings in the tile, which were visible in the aerial imagery. For every object a polygon was created in a shapefile layer using QGIS. To ensure consistency in the ground truth data collection methodology, guidelines where specified on how to generate the data, especially focusing on what kind of objects to record, and what kind not.

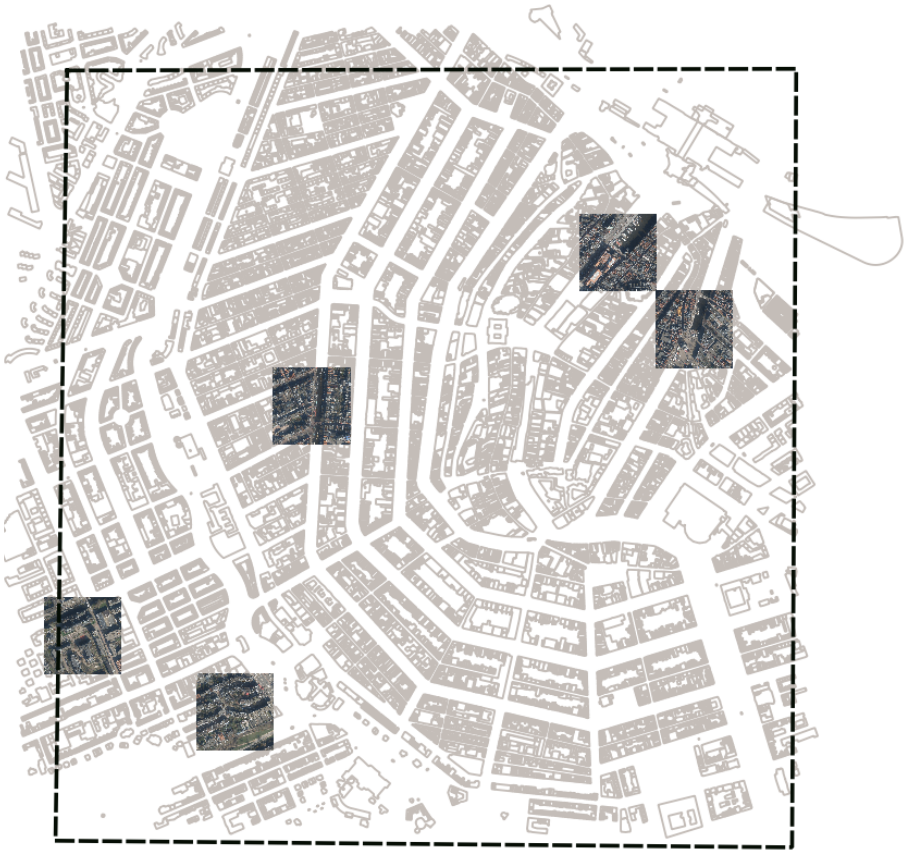


Figure 24: Overview of the study area

The main goal was to draw all objects that are sticking out and are not part of the roof itself, leading to the following guidelines:

- Capture every object that is on top of a roof and;
  - Has a shadow, or;
  - Has a very coarse texture, or;
  - Sticks out on 3D viewer of Google Earth

- Do not capture
  - Windows
  - Solar panels
  - Dormers

Next to a unique ID, two attributes were added for each object; 'brick_chim' and 'polyarea'. The first attribute could contain the value 'y' or 'n', which indicated if the object is a square brick chimney ('y') or not ('n'). The value of this attribute was selected by using both Google Street View and the 3D capabilities of Google Earth. An example of a square brick chimney on an image, in Google Street View and in Google Earth is presented in figure 25.

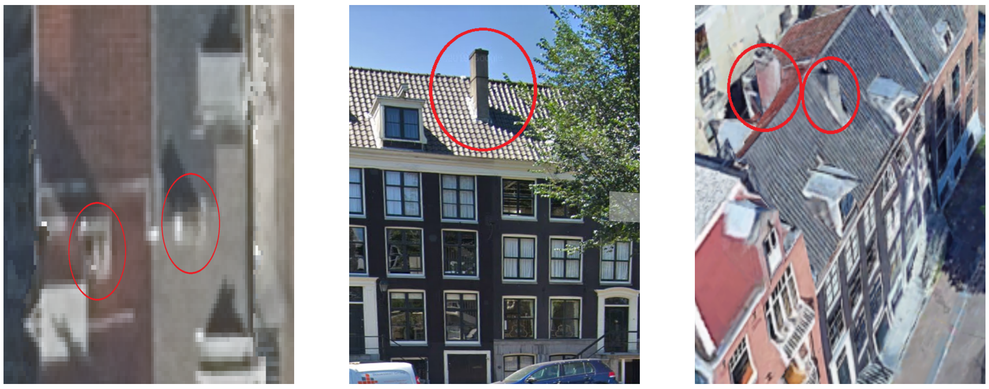


Figure 25: The same square brick chimneys represented in an image, Google Street View and Google Earth (from left to right)

The second attribute contains the polygon area. All the polygons with area less than 0.2 square meters (m2) have not been considered in the whole process, as they correspond to small structures that can not be detected in the point cloud.

After the ground truth for the five tiles was collected, the content of all the files was checked by one individual for consistency and completeness. Hereafter, the shapefiles were merged and every object received an unique ID.

In figure 16, the ground truth data are presented with green color whereas the polygons of BAG buildings with light grey color.

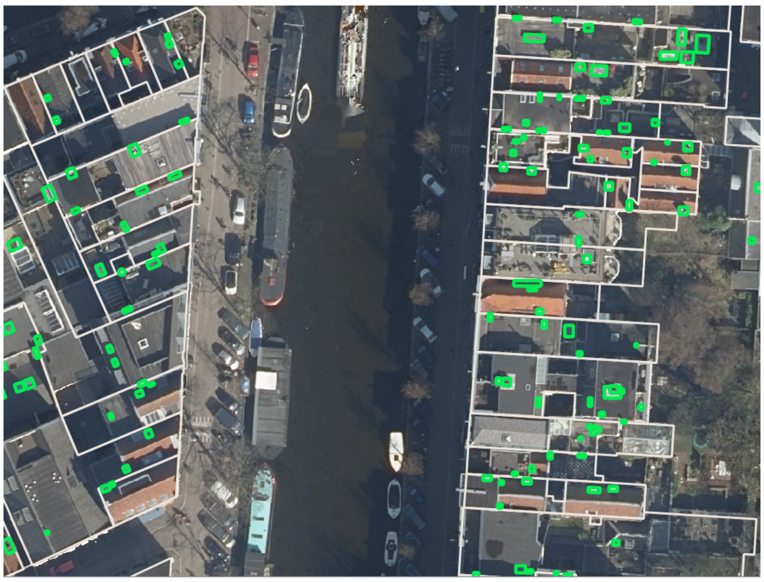


Figure 26: Sample of ground truth data

3D point segments to 2D polygons
To be able to compare the results of both the object detection algorithm and afterwards the chimney detection algorithm, the object detected in the extent of the five tiles were converted from groups of 3D point segments to 2D polygons. Specifically, the Z coordinate was removed from all the points. Hereafter two approached were tested.

In the first approach, the convex hull was generated for each segment - each detected deviation above the roof (see fig 27). In the second approach, the concave hull was generated for each segment (see fig 28). These two different approaches were tested, as the concave hull based on the alpha shape technique (Delaunay Triangulation (DT)) is affected by the distribution of points in a segment. That's why the triangulated points with triangle sides bigger than a specific threshold value (0.2m) were not used (i.e. points that were far away).

For both approaches, the generated polygons were stored in a shapefile.

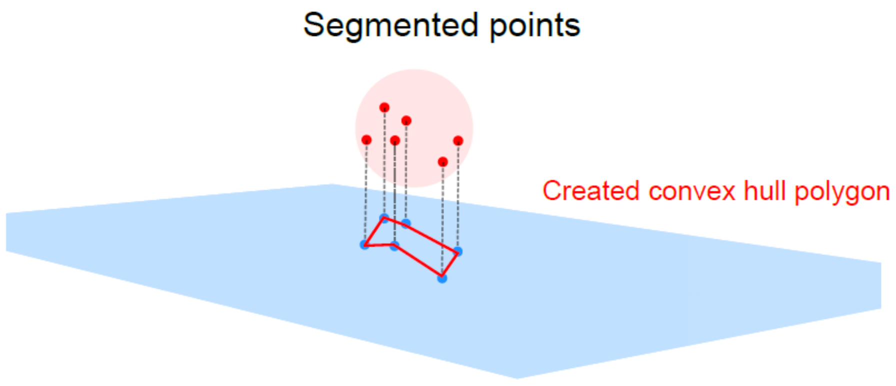


Figure 27: Example of a created convex hull for a detected segment.

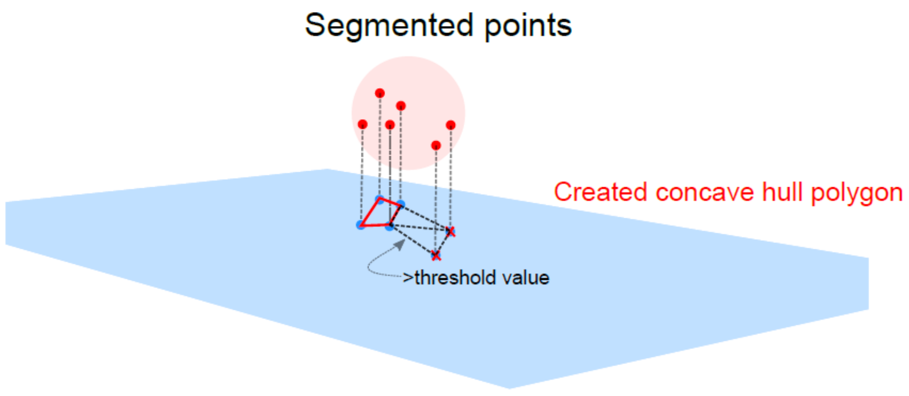


Figure 28: Example of a created concave hull for a detected segment.

Quality check
The last step of the quality assessment comprises the comparison between the ground truth data and the 2D polygons representing the detected objects and square brick chimneys by the algorithm. Test statistics were calculated for all the detected objects and for the matching objects (where a match is detected between a ground truth object and a algorithm object). This comparison is done for both the concave polygons and the convex polygons. The calculated test statistics are presented in table 5 and table 6.

Objects above roof	
Test statistic	Description
Objects detected	Percentage of ground truth objects that are detected by the algorithm (Based on a 10% overlap; 10% of area of ground truth is overlapping with corresponding detected object and a maximum area ratio of 2.5)
Mean overlap	For the detected ground truth objects, the average percentage of the area of the ground truth object that overlaps with the matched detected object
True positives	Number of occasions where algorithm has detected an object at a location where there is also an object in ground truth
False positives	Number of occasions where algorithm detected an object while there is no object at that location in the ground truth
False negatives	Number of occasions where algorithm did not detect an object at a location where there is an object in the ground truth

Table 5: Test statistics calculated for the detected objects above the roofs

Description of test statistics calculated for all the matching objects	
Test statistic	Description
Confusion matrix	Presents the true positives, true negatives, false positives and false negatives
Overall accuracy	Total number of well classified segments/polygons divided by all the total amount of matches
Kappa coefficient	The overall accuracy with the chance component removed

Table 6: Description of test statistics calculated for the matched objects

The 'Objects detected' test statistic represented in table 5, depends on two parameters: a threshold on the minimum percentage of overlap required to identify the ground truth object as 'detected', and the maximum ratio that is allowed between the area of the ground truth polygon and the area of the 2D polygon representing a detected segment.

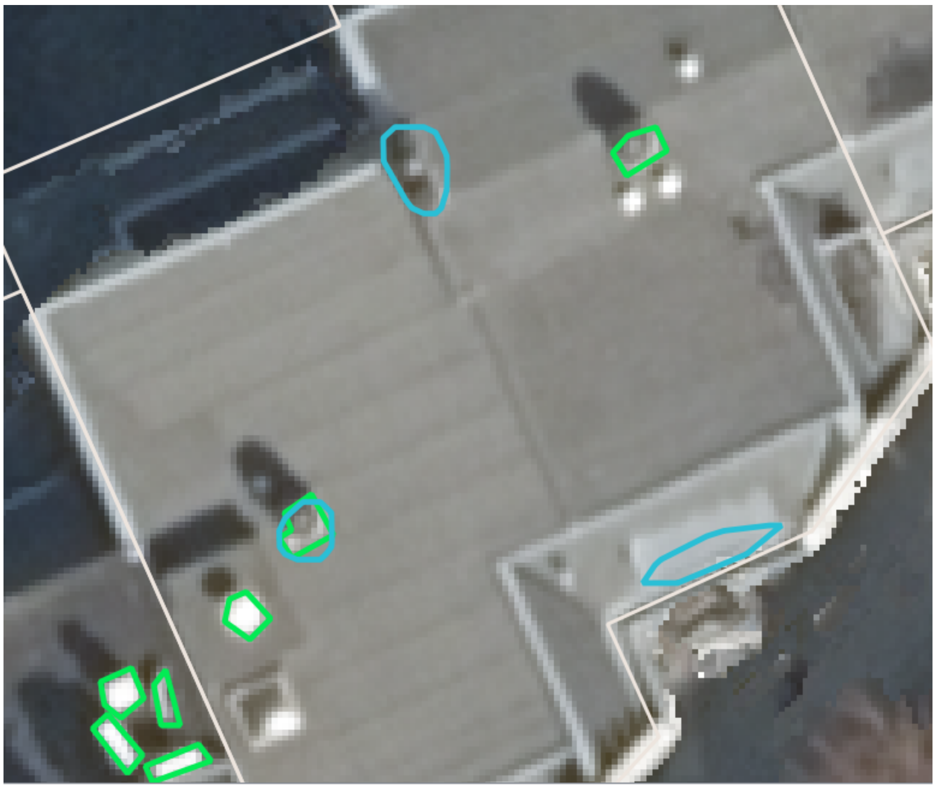


Figure 29: Case 1: Overlap between ground truth (green) and detected structures (blue)

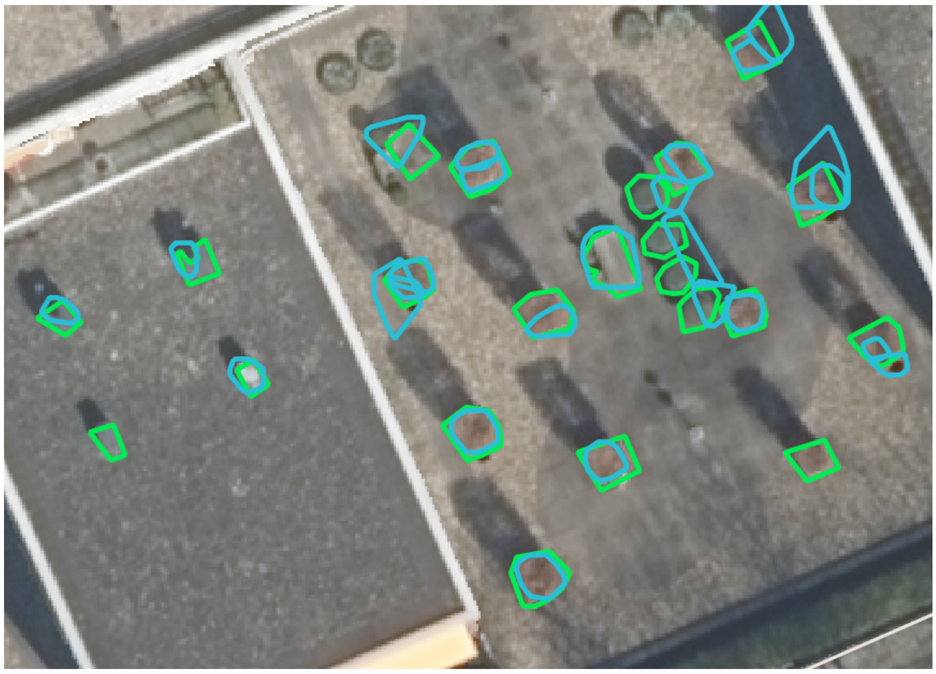


Figure 30: Case 2: Overlap between ground truth (green) and detected structures (blue)

As seen in Figures 29 and 30, the ground truth data (green color) are overlapping with the detected structures (light blue color). The 10% overlap was used as the minimum required overlap in order to detect as much as possible overlapping polygons. Since the ground truth data were digitized manually (i.e. not well-drawn shapes) and the detected structures-polygons were created from the segmented points (i.e. not good shapes), it was important to not pay attention to the exact boundaries of the polygons, but to their minimum overlapping area.

6.3 Visualization

The envisioned tool for the verification of the regulation should allow the client to select a building or chimney in the 3D model. Upon selection, all relevant properties of the objects should be displayed. This includes the time of the last update and whether the object complies to the regulations according to the predictions made by the algorithm. As the object detection did not yield satisfactory results, it should be remembered the data used for the visualization is conceptual - further research is needed to produce more reliable results. The software involved in the visualization of the output will be briefly discussed in the results.

7 Results

7.1 Convex versus concave

In figure 31, the polygons of the ground truth data are displayed in green, while the convex hull and the concave hull is displayed in red and light blue respectively. It is believed that the convex hull provides a better 2D representation than the concave hull. The shape and the size of concave hull polygons cover less area than that of the convex hull whilst this larger area often follows the shape of ground truth data better. It was decided to use the convex hull approach in order to maximize the possibility to overlap larger area of the corresponding ground truth polygon.

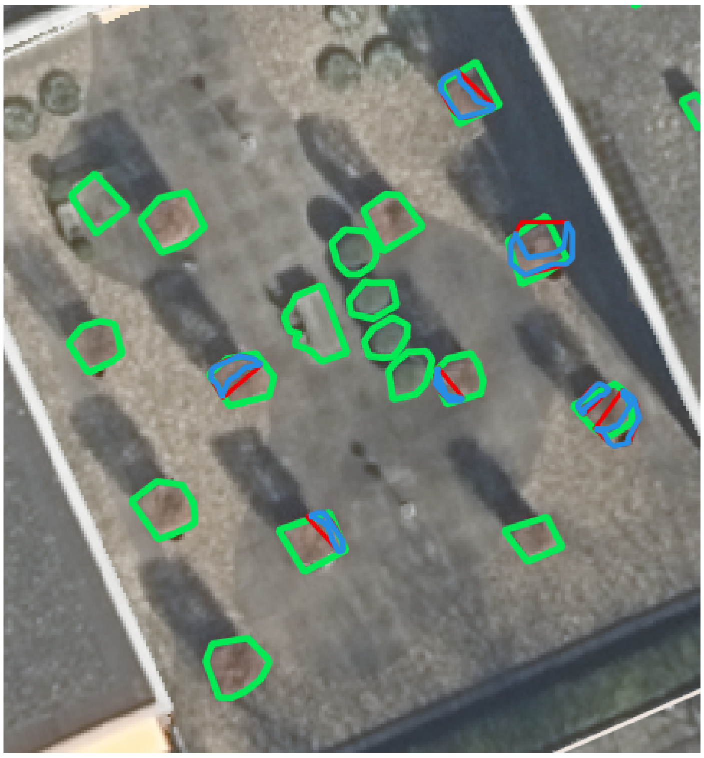


Figure 31: Overlap between concave (red) and convex hull (blue) approaches presented on top of the ground truth (green)

7.1.1 Objects above roof detection

In table 7 below, the test statistics calculated for the results of the object detection algorithm are presented. The ground truth dataset consists of 3108 objects. In the whole extent of the ground truth, the algorithm has detected 1185 objects. Approximately 5% (=154 objects) of the ground truth objects have been detected by the algorithm (true positives), based on a 10% overlap of the convex polygons representing the detected object with the ground truth and a maximum area ratio of 2.5. The mean overlap of these matching objects is approximately 64%. The false positives indicate that the algorithm has detected 1031 objects at locations where no objects were present in the ground truth.

Objects above roof	
Test statistic	Value
Objects detected	4.95%
Mean overlap	63.71%
True positives	154
False positives	1031
False negatives	2954

Table 7: Test statistics calculated for the detected objects above the roofs

When visualizing the resulting detected objects by the algorithm, it could be noted that overall the algorithm performs better at buildings with flat roofs when compared to more complex roof structures. As seen in figures 32 and 33, the

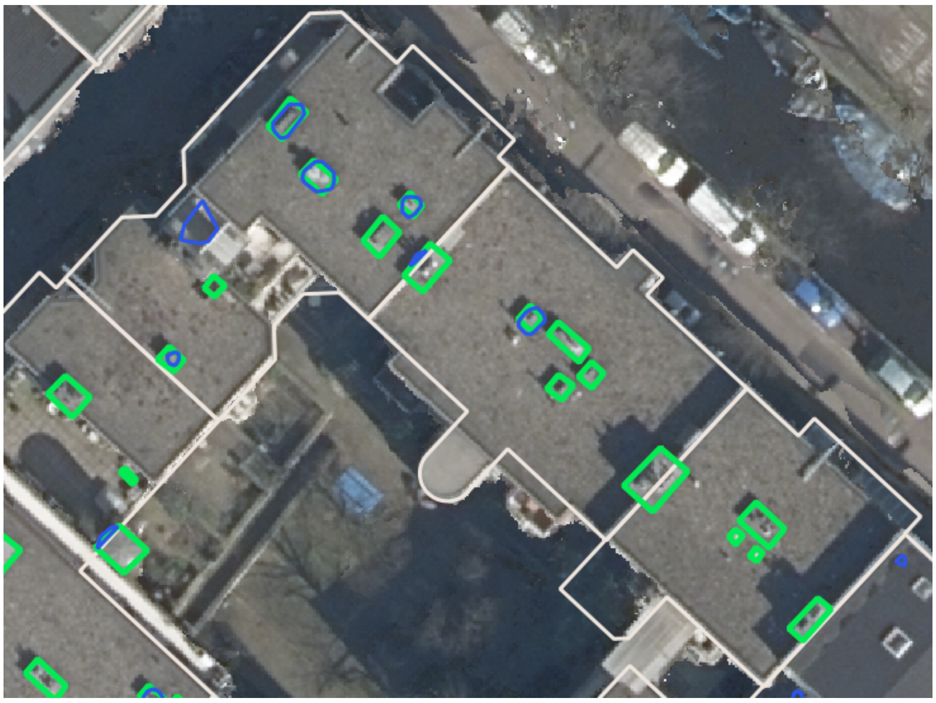


Figure 32: Flat roofs. Green = ground truth, blue = Convex hull

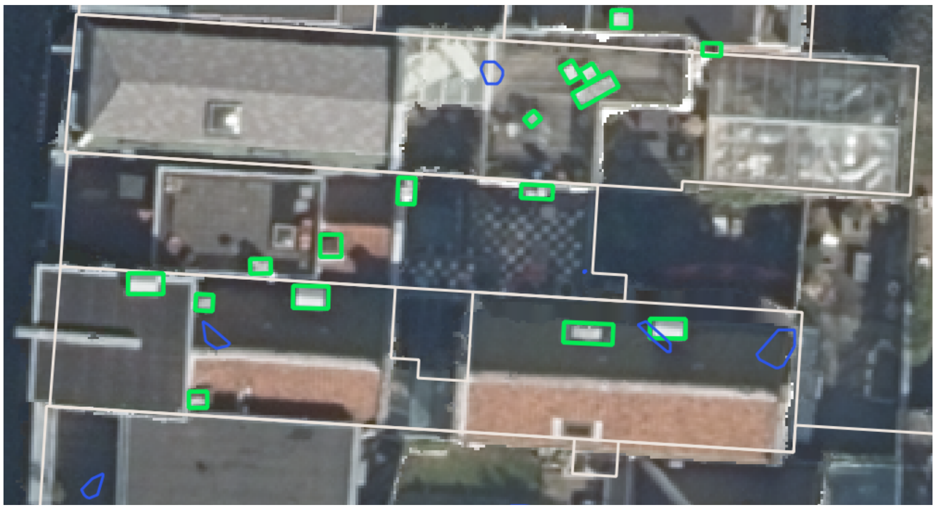


Figure 33: Non flat roofs. Green = ground truth, blue = convex hull.

7.1.2 Detection of chimneys

The confusion matrix and the calculated test statistics for the matching objects (the true positives in table 7), are given in table 8 and table 9.

For the matching objects detected, the true positives show that the algorithm was able to detect 17 square brick chimneys which are also chimneys in the ground truth. This is 5% percent of all the square brick chimneys present in the ground truth and 50% of all ground truth chimney objects for which a match has been detected by the algorithm.

The algorithm has detected 833 chimneys in the whole extent of the ground truth. Therefore, 816 chimneys where detected by the algorithm at locations where there is no chimney or object in the ground truth data.

		DO	DO	All
		no	yes	all
GT	no	40	80	120
GT	yes	17	17	34
All	all	57	97	154

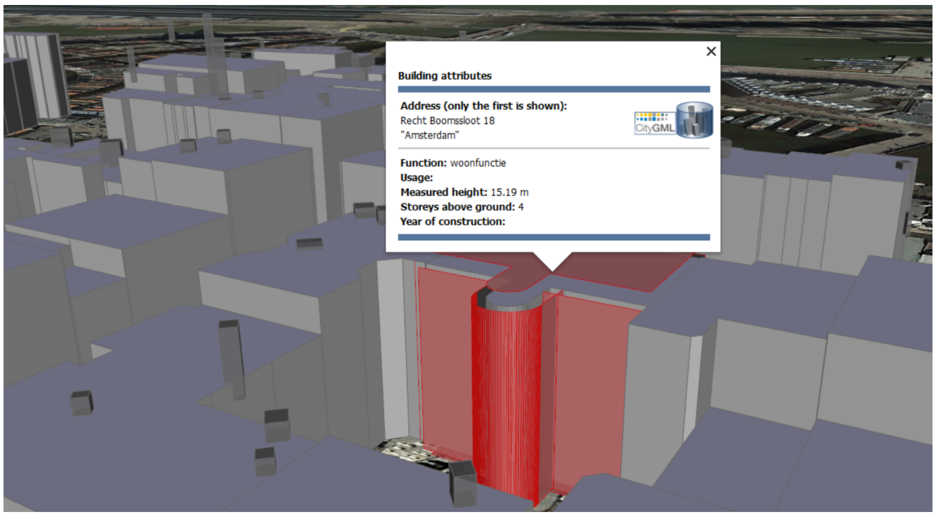
Table 8: The confusion matrix for the matching objects. GT = Ground truth, DO = Detected objects

Test statistic	Value
Overall accuracy	37.01%
Kappa coefficient	-0.10

Table 9: Test statistics calculated for the detected square brick chimneys

7.1.3 3D City Model

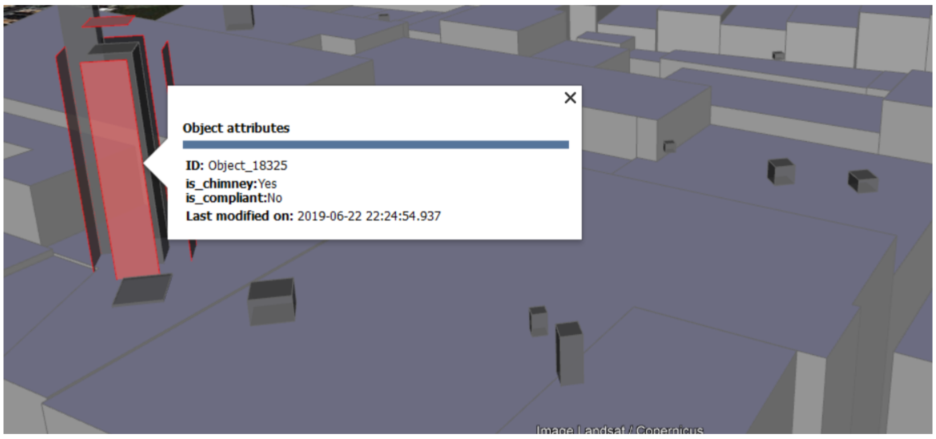
For visualization purposes the CityGML model can be converted to KML/COLLADA/glTF format, through the 3DCityDB Importer/Exporter. These conversions prepare the model to be used in a variety of applications including Google Earth, Cesium and GIS. Figure 35 shows an example of the city model exported in KML. Such applications are user-friendly, in that they allow the user to easily distinguish between the different spatial features. In this way, only the relevant pieces of information are retrieved - through information balloons. These have been created through HTML templates. The structure of the CityGML file could allow one to hover/click on the individual building installations to view if they do/do not comply to the regulations. Of course, this is a hypothetical 'template'. If the object detection had produced any reliable results, then the formulation of the object attributes would probably be of another type (for example, using percentiles to define object height and estimating the compliance with this).


Building attributes

Address (only the first is shown):
Recht Boomssloot 18
"Amsterdam"

Function: woonfunctie
Usage:
Measured height: 15.19 m
Storeys above ground: 4
Year of construction:

Figure 34: Buildings modelled in Google Earth, with attributes shown and hovering enabled


Object attributes

ID: Object_18325
is_chimney: Yes
is_compliant: No
Last modified on: 2019-06-22 22:24:54.937

Figure 35: Objects on the roof modelled in Google Earth, and possibly relevant attributes

8 Discussion

The calculated test statistics for all the objects detected by the algorithm indicate that the algorithm is able to detect only 5% of the objects in the ground truth dataset. In addition, it should be noted that the false positives and the false negatives are high for both all the objects detected by the algorithm and the algorithm objects for which a match is found with the ground truth.

The overall accuracy reaches 37%, however, the Kappa coefficient indicates that the distinction between square brick chimneys and other roof objects of the matched objects, provided by the algorithm, is of lower quality than if random classification were to be applied. This suggests that there is currently no added value of applying the object and chimney detection algorithm.

After execution of the algorithm it can be concluded that it is possible to detect some small structures over the roof by using a point cloud made from aerial images and a LIDAR point cloud. However, next to real structures on the roof also many false structures are detected. A strong limitation to this research, which likely contributes to these outcomes, is the quality of the input point clouds. The roof surfaces in the DIM point cloud are bumpy. In order to cope with these bumps it was decided to use a threshold of 40 cm above and below the generated roof plane, to eliminate points belonging to roofs from the DIM point cloud. Consequently, object detection is limited to objects that are at least 40 cm high. This problem could be solved by using a point cloud of higher quality, such as LIDAR point clouds like the AHN3. However, using solely the AHN3 for object detection was not possible as the density of the point cloud is quite low. Consequently objects are being represented in the point cloud by only a few points.

Another important limitation for this project is the methodology applied for ground truth data generation. No data is available on the location and/or the type of chimneys in Amsterdam, therefore the data needed to be created manually. In addition the data was collected by different researchers. Even though it is believed that the selected approaches were of the highest quality achievable, the methods applied have some large drawbacks. Even on a high quality aerial image it is very difficult to recognize chimneys or distinguish them from other roof structures. Even with Google Earth and Google street view used as an additional aid, a complete identification of objects above roofs - and especially chimneys -, is not possible. A lack in quality of the ground truth data strongly influences the outcomes of the quality check of the generated algorithm. For example, the algorithm has detected many more square chimneys than were present in the ground truth data. Therefore, it is possible that during the manual approach of ground truth data collection many square chimneys have been missed. For future research in this field, higher quality ground truth data with the location of objects and chimneys is essential to improve the reliability of the quality check of the algorithm.

In addition, the way the quality check of the detected objects is executed also influences the results. For example, it was decided to use the convex hull to represent detected segments in 2D as it was believed that it provided a better representation of an object than the concave hull. However, if one would argue the concave hull would be more suitable, the percentage of overlap between polygons will change, leading to a different amount of matches being detected. In further research the influence of the 2D representation of objects on the quality assessment of the algorithm needs to be further examined.

The results are also influenced in the case of the filtering of objects over the roof and the distinction of square chimneys from other roof objects. A change of the distinction criteria will directly affect the number of objects over the roof that are kept, as well as the number of chimneys. Less rigid numbers, for example a span x-y-ratio with lower than 0.9 or higher than 1.1, leads to more objects left after the filtering and also to more possible square chimneys. On the other hand, more outliers (part of the roofs) from the segmentation will be kept and the number of false values would increase. Furthermore, the threshold which is set to eliminate tree points; a distance of maximum 2 meters from the centroid of a segment to the roof, has a strong limitation. This threshold has the disadvantage that very large chimneys can not be detected, as they will be eliminated.

Furthermore, when focussing on the classification errors of the algorithm, the distinction between chimneys and other objects on the roof is very challenging. Chimneys can have many different shapes and can differ much in size. Chimneys represented as points are hard to distinguish from other objects, such as air conditioning objects. Again, this problem could be solved with a higher quality point cloud, as more details would be depicted and some unique attributes of the roof objects would be visible.

Even if the detection was successful, another challenge would still have to be tackled: linking the detected objects to the proper building units. Although the 3D model was enriched with information regarding building units, it is currently not possible to determine the exact location of these units within each building geometry. Hence, even if a certain object/chimney is modelled as a building installation belonging to a certain BAG building, it is not possible to distinguish to which unit it actually belongs. The problem becomes particularly apparent in the context of large mixed-use buildings: there is no way of telling whether a non-compliant chimney belongs to a restaurant located alongside the front facade or to a commercial unit at the far end of the building. Moreover, the BAG geometries that were used throughout the process also present problems. In some cases, the geometries represent clusters of buildings – this was observed for some rowhouses, which are a quite common building type in the centre of Amsterdam. This further decreases the chances of matching chimneys to the proper premises. More detailed data is needed to develop the model in a way that allows chimney installations to be verified against regulations.

Finally, interoperability is an important concept for generating an usable and scalable output. In other words, the result of the verification of regulations should make use of Open standards that enforce interoperability. In the case of geospatial data, the Open Geospatial Consortium (OGC) is one of the main organizations behind these standards. OGC standards used in this project include the international encoding standards KML and CityGML, besides the community standard LAS [6]. However, the use of open standards does not guarantee optimal output. An example are the addresses in the 3D model: the documentation of the software used to process the CityGML file [1] only allows parsing a number of fixed templates. It was not possible to create multiple *xAL:Premises* within one *xal:Thoroughfare*. Because of this, each address registered to a building had to be written to a separate *xAL:AddressDetails*. This caused much redundancy, as the same postal code, street name and house number had to be repeated many times. The result is a much larger and verbose file. Another examples occurred when trying the visualize the roof surfaces in obj file format, through the software CloudCompare. The shape of the concave hulls could not be properly displayed, which could lead to wrong interpretations. However, this limitation was acknowledged timely and the software Mapple was ultimately used for visual validations.

This is a consequence of very generic and flexible standards: they enable many different representations and operations, but can become very complex to handle. Applications involving these standards may therefore choose to limit their scope and only focus on a subset of the specification. For future work, the capabilities and limitations of different software should be thoroughly reviewed before generating any output.

9 Conclusions and recommendations

The objective of this project was to explore an automated approach to handling (and presenting) compliance of chimneys to regulations. To this end, a number of research questions were formulated. Firstly, the research focused on the detection of structures on top of roofs. This problem needed to be tackled before moving on to distinguishing the actual chimneys from all other structures. The results of this object detection are determined by the initial segmentation of the building's points and the roof reconstruction. When visualizing the 2D projection of the segmented points (as polygons) on top of the original aerial images, it seems possible to distinguish some objects.

Even though the algorithm was able to detect 5% of the objects present in the ground truth, the high amount of false positives indicated inconsistencies in either the ground truth or the algorithm itself. Changes in the parameters of the codes used for detection and revision of the ground truth did not lead to better results. Thus, this questions the possibility to make further statements on the identification of chimneys. However, an attempt was made to filter out square chimneys from the detected objects. Even though some square brick chimneys where correctly detected, most of the chimneys in the ground truth were not.

In conclusion, it can be stated that identifying small objects and square chimneys on the roof, using a point cloud generated from aerial images, is currently not possible with the applied methodology. With current input data quality there is not a reliable method to do so. Whereas point clouds created by stereo image matching usually have many points to represent small objects, their accuracy is limited. Even flat roofs are very bumpy and it is hard to distinguish between an object or simply false values. LIDAR point clouds instead are very accurate in their height but the distances between the points are too big and small objects are not represented very well or not at all. With better quality for the input data a big jump in the quality of the detection is to be expected. These accuracy limitations also obstructs the possibility to distinguish square chimneys from other roof objects.

Regarding the 3D city model, the basic idea was to link the outcome of the detection algorithm (compliance to regulations) to actual buildings and visualize this in an interactive tool. However, the detection was not completely successful and the modelling itself presents some limitations. Therefore, further developments are needed to create a complete and usable tool.

Due to the time and manpower limitations of this project, only one approach for identifying chimneys could be examined. The following approaches are suggested to be investigated in further research:

- The aerial images of the point cloud could be used for feature detection in images using machine learning approaches. Even though chimneys can differ greatly in color and shape it could be a good decision supporter for the existing method.
- The color information of each point in the point cloud could be used as an additional information source for distinguishing chimneys from other roof structure/vegetation over the roof.
- High quality infrared images captured during the winter can contain valuable information that allows for localization of chimneys on top of roofs, as an operational chimney emits heat.

Overall it can be concluded that if the municipality of Amsterdam is interested in automated chimney detection and validating to regulations, they need to invest in better quality input data. LIDAR point cloud data should be collected for the city, which contains more points per square metre and high quality ground truth data should be generated. It is believed that when improved input data is used as inputs for the applied methods of this research, the output will be of a higher quality and therefore of a higher value.

References

[1] 3DCityDB. *Documentation.* URL: https://www.3dcitydb.org/3dcitydb/documentation/. (accessed: 19.06.2019).

[2] Gemeente Amsterdam. *Gemeente Amsterdam.* URL: https://www.amsterdam.nl/stelselpedia. (accessed: 19.06.2019).

[3] Gemeente Amsterdam. *Monumenten.* URL: https://maps.amsterdam.nl/monumenten/. (accessed: 19.06.2019).

[4] Open data Amsterdam. *Energielabels in Amsterdam.* URL: https://data.amsterdam.nl/datasets/rV-5t8Wzy6TWzA/. (accessed: 19.06.2019).

[5] Open data Amsterdam. *Eten Drinken.* URL: https://data.amsterdam.nl/datasets/y5I1tIFyd9e8aA/. (accessed: 19.06.2019).

[6] Open Geospatial Consortium. *OGC Standards and Supporting Documents.* URL: https://www.opengeospatial.org/standards. (accessed: 19.06.2019).

[7] Academic Dictionaries and Encyclopedias. *Nyquist–Shannon sampling theorem.* URL: https://enacademic.com/dic.nsf/enwiki/23700. (accessed: 19.06.2019).

[8] Vosselmann Georg. *Point cloud segementation for urban scene classification.* Oct. 2013.

[9] Rutzinger Martin Pfeifer Norbert Jochem Andreas Höfle Bernhard. *Automatic Roof Plane Detection and Analysis in Airborne Lidar Point Clouds for Solar Potential Assessment.* July 2009.

[10] Kadaster. *KadasterBAG.* URL: https://imbag.github.io/praktijkhandleiding/. (accessed: 19.06.2019).

[11] Peter Wonka Liangliang Nan. *PolyFit: Polygonal Surface Reconstruction from Point Clouds.* (accessed: 14.06.2019).

[12] Michael A. McGinley and St. Croix Sensory. *The 'Gray Line' Between Odor Nuisance and Health Effects.* URL: http://www.fivesenses.com/Documents/Library/23%20%20Gray%20Line%20Nusance%20Health.pdf. (accessed: 14.06.2019).

[13] Hugues Hoppe Michael Kazhdan Matthew Bolitho. *Poisson Surface Reconstruction.* (accessed: 14.06.2019).

[14] NLextract. *Data klaar voor gebruik.* URL: https://nlextract.nl/downloads/. (accessed: 19.06.2019).

[15] Overheid. *Activiteitenregeling milieubeheer.* URL: https://wetten.overheid.nl/BWBR0022830/2019-04-01. (accessed: 14.06.2019).

[16] Biljecki Filip Peters Ravi Ledoux Hugo. *Visibility analysis in a point cloud on the medial axis transform.* Nov. 2015.

[17] C. Ressl. *Assessing the accuracy of dense image matching.* URL: https://www.isprs.org/tc2-symposium2018/images/ISPRS-Keynote_Ressl.pdf. (accessed: 14.06.2019).

[18] 3D geoinformation group TU Delft. *3dfier.* URL: https://github.com/tudelft3d/3dfier. (accessed: 18.06.2019).

[19] 3D geoinformation group TU Delft. *3dfier.* URL: https://github.com/tudelft3d/3dfier/tree/master/resources/BGT_prepare. (accessed: 18.06.2019).

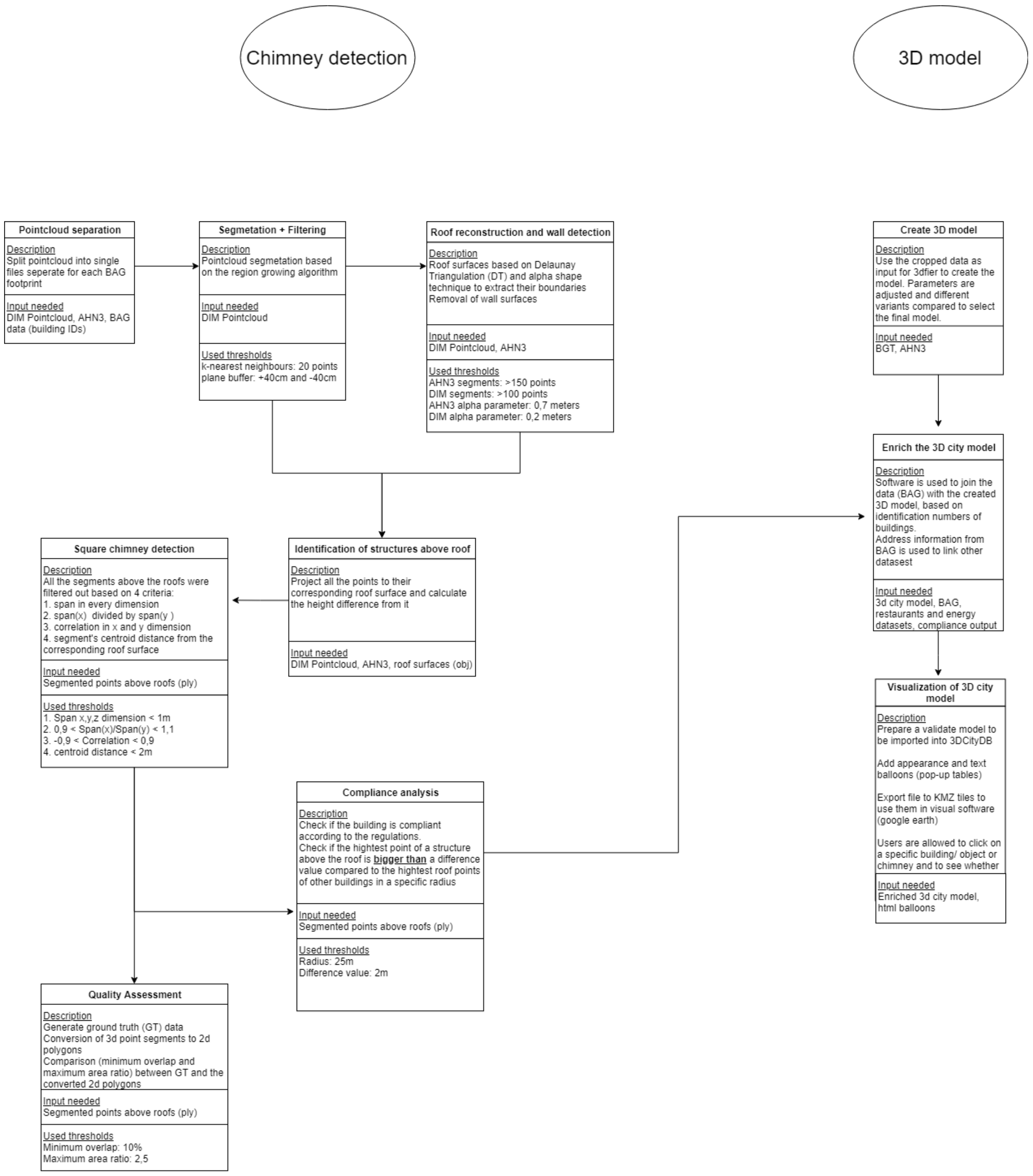
[20] 3D geoinformation group TU Delft. *3dfier.* URL: https://github.com/tudelft3d/3dfier/blob/master/resources/config_files/myconfig_README.yml. (accessed: 18.06.2019).

[21] 3D geoinformation group TU Delft. *CityJson.* URL: https://viewer.cityjson.org/. (accessed: 18.06.2019).

[22] Förstner Wolfgang Yang Michael Ying. *Plane Detection in Point Cloud Data.* Feb. 2010.

[23] Liu Caixia Zhang Yaling Zhao Ruibin Pang Mingyong. *Robust Normal Estimation for 3D LiDAR Point Clouds in Urban Environments.* Mar. 2019.

A Appendix



Pointcloud separation
Description
Split pointcloud into single files seperate for each BAG footprint
Input needed
DIM Pointcloud, AHN3, BAG data (building IDs)

Segmetation + Filtering
Description
Pointcloud segmetation based on the region growing algorithm
Input needed
DIM Pointcloud
Used thresholds
k-nearest neighbours: 20 points
plane buffer: +40cm and -40cm

Roof reconstruction and wall detection
Description
Roof surfaces based on Delaunay Triangulation (DT) and alpha shape technique to extract their boundaries
Removal of wall surfaces
Input needed
DIM Pointcloud, AHN3
Used thresholds
AHN3 segments: >150 points
DIM segments: >100 points
AHN3 alpha parameter: 0,7 meters
DIM alpha parameter: 0,2 meters

Identification of structures above roof
Description
Project all the points to their corresponding roof surface and calculate the height difference from it
Input needed
DIM Pointcloud, AHN3, roof surfaces (obj)

Square chimney detection
Description
All the segments above the roofs were filtered out based on 4 criteria:
1. span in every dimension
2. span(x) divided by span(y)
3. correlation in x and y dimension
4. segment's centroid distance from the corresponding roof surface
Input needed
Segmented points above roofs (ply)
Used thresholds
1. Span x,y,z dimension < 1m
2. 0,9 < Span(x)/Span(y) < 1,1
3. -0,9 < Correlation < 0,9
4. centroid distance < 2m

Compliance analysis
Description
Check if the building is compliant according to the regulations.
Check if the hightest point of a structure above the roof is **bigger than** a difference value compared to the hightest roof points of other buildings in a specific radius
Input needed
Segmented points above roofs (ply)
Used thresholds
Radius: 25m
Difference value: 2m

Quality Assessment
Description
Generate ground truth (GT) data
Conversion of 3d point segments to 2d polygons
Comparison (minimum overlap and maximum area ratio) between GT and the converted 2d polygons
Input needed
Segmented points above roofs (ply)
Used thresholds
Minimum overlap: 10%
Maximum area ratio: 2,5

Create 3D model
Description
Use the cropped data as input for 3dfier to create the model. Parameters are adjusted and different variants compared to select the final model.
Input needed
BGT, AHN3

Enrich the 3D city model
Description
Software is used to join the data (BAG) with the created 3D model, based on identification numbers of buildings.
Address information from BAG is used to link other datasest
Input needed
3d city model, BAG, restaurants and energy datasets, compliance output

Visualization of 3D city model
Description
Prepare a validate model to be imported into 3DCityDB

Add appearance and text balloons (pop-up tables)

Export file to KMZ tiles to use them in visual software (google earth)

Users are allowed to click on a specific building/ object or chimney and to see whether
Input needed
Enriched 3d city model, html balloons

Figure 36: Methodology