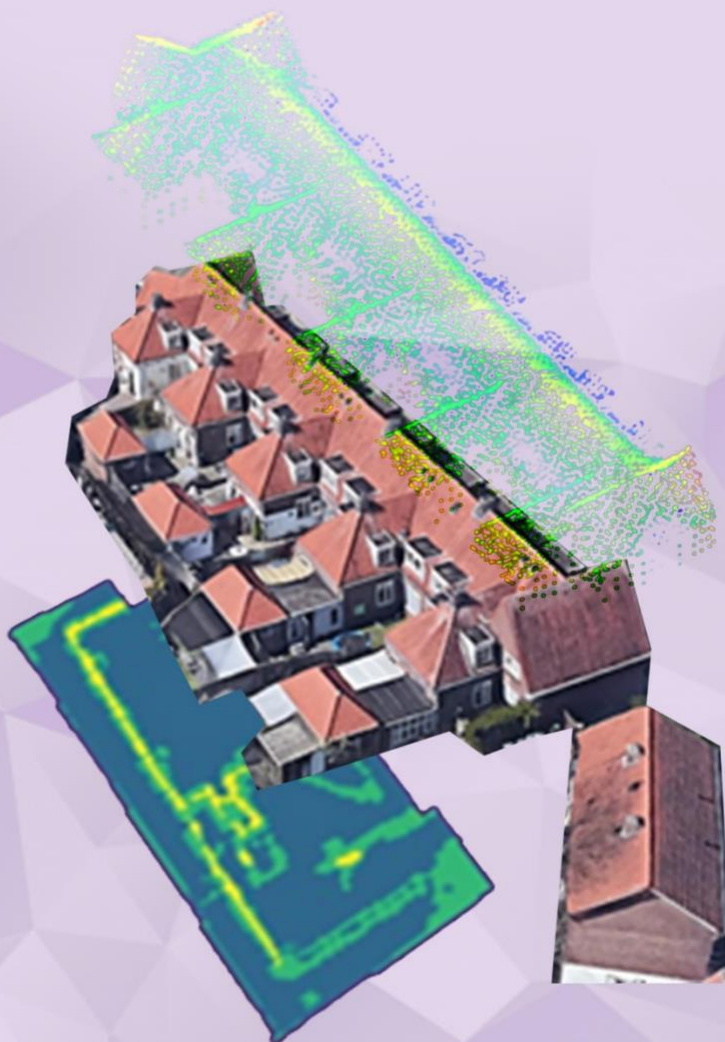


Delft University of Technology
Faculty of Architecture and the Built Environment
MSc Geomatics for the Built Environment

Inferring roof semantics for more accurate solar potential assessment

GEO1101 - Synthesis Project



Team Members

Irène Apra

Carolin Bachert

Camilo Cáceres

Özge Tufan

Ondrej Veselý

Abstract

Led in cooperation with the company Brink, who provides management and consultation services for construction and real estate sectors, this Synthesis Project aims at automatically deriving meaningful information about buildings. More precisely, the focus is to automatically detect roof obstacles - such as dormers, chimneys, and solar panels - to be able to determine the available roof surface for new solar panel installation, and therefore to perform more accurate solar potential analysis. For this purpose, three different methods are developed and implemented to increase the results' accuracy, which are geometry-based, unsupervised, and supervised classification. While *AHN3* point cloud and *3D BAG* Level of Detail (LoD) 2.2 building models are used for the geometry-based classification, the input data of the unsupervised image classification consists of aerial images and *BAG* footprints. Finally, supervised image classification method makes use of the aerial images as well as the *BAG* footprints and a dataset of manually labelled solar panel polygons. The results show that the accuracy of individual methods is not sufficient; therefore, the outputs of all three methods are merged together into one pipeline, with the aim of obtaining one final end product. The latter is the *3D BAG* LoD2.2 building model in *CityJSON* format, enhanced with three new attributes per building: the obstacle area on the roof, the available area for installing solar panels, and a Boolean value showing whether the building has existing solar panels or not. Additionally, an enhanced point cloud for future use is generated, with a new attribute per point indicating its distance to the 3D model and therefore its potential for being an obstacle or not. The assessment of the results with the ground truth illustrates that the algorithm gives promising results; however, the scope of the project can be broadened, and improvements can be made to increase the accuracy as well as the efficiency.

Table of Contents

1	Introduction	1
1.1	Project introduction	1
1.2	Project context	1
1.3	Reading guide	2
2	Theoretical Background	3
2.1	Solar potential	3
2.1.1.1	Solar Radiation	3
2.1.1.2	Aspect and Tilt	3
2.1.1.3	Shadow Casting and Available Area	3
2.2	Obstacle Detection	5
2.3	Image Classification	6
3	Methodology & Implementation	7
3.1	Used Datasets	7
3.1.1	AHN3	7
3.1.2	BAG and 3D BAG	7
3.1.2.1	Dutch National Register of Addresses and Buildings (BAG)	7
3.1.2.2	3D BAG	8
3.1.3	Aerial images (Luchtfotos / PDOK)	9
3.2	Overview of the whole workflow	10
3.3	Part I: Geometry-based classification	11
3.3.1	Methodology	11
3.3.2	Input Data and pre-processing	12
3.3.2.1	AHN3 point cloud	12
3.3.2.2	3D BAG	14
3.3.3	Implementation	14
3.4	Part II: Unsupervised image classification	16
3.4.1	Methodology	16
3.4.2	Input Data and pre-processing	19
3.4.2.1	Key Registers Addresses and Buildings (BAG)	19
3.4.2.2	Aerial image (Luchtfoto)	19

3.4.2.3	BAG and aerial images alignment	20
3.4.3	Implementation.....	21
3.5	Part III: Supervised image classification	23
3.5.1	Methodology	23
3.5.2	Input Data and pre-processing.....	25
3.5.2.1	Solar panel inference.....	25
3.5.2.2	Additional semantics.....	27
3.5.3	Implementation.....	28
3.5.3.1	Solar panel detection	28
3.5.3.2	Other semantics.....	29
3.6	Merging the detection methods	31
4	Discussion of the Results	33
4.1	Obstacle detection.....	33
4.1.1	Validation of the result, comparison to ground truth	33
4.1.1.1	Validation method:.....	33
4.1.1.2	Application example of the validation method to one building:.....	34
4.1.1.3	Overall result, general statistics:	35
4.2	Solar panel inference	37
4.3	Additional semantics	39
4.3.1	Assessment and discussion of the results	40
4.3.1.1	Impact 1 - Rasterization, accuracy loss:.....	40
4.3.1.2	Impact 2 - BAG and aerial image alignment:	40
4.3.1.3	Impact 3 - Sample data:	40
4.3.1.4	Impact 4 – Validation method used:	40
4.3.1.5	Impact 5 – Generalizability of input data:	40
4.3.2	Limitations.....	41
4.3.2.1	Geometry-based classification	41
4.3.2.2	Unsupervised image classification	41
4.3.2.3	Supervised image classification	42
5	Reflection	43
5.1	MoSCoW requirements	43
5.2	Problems encountered & lessons learnt	44
5.3	Connection to previous courses in Geomatics.....	44

6	Conclusions and Future Work	46
6.1	Conclusions.....	46
6.2	Future Work.....	47
7	References	48
8	Appendix I.....	51
8.1	Team members.....	51
8.2	Roles	53
9	Appendix II.....	54
9.1	Additional figures	54

List of Figures

FIGURE 1.	GLOBAL HORIZONTAL IRRADIATION NETHERLANDS	4
FIGURE 2.	EXAMPLE BUILDING WITH ROOF OBSTACLES IN DELFT	5
FIGURE 3.	EXAMPLE BUILDING FROM BAG.....	7
FIGURE 4.	FEATURE EXTRACTION: (1) INPUT DATA. (2) ROOF PLANES DETECTED WITH REGION- GROWING ALGORITHM. (3) DETECTED LINES	8
FIGURE 5.	THE FIVE LODS.....	8
FIGURE 6.	IMPROVED LODS	9
FIGURE 7.	EXAMPLE BUILDING FROM 3D-BAG VIEWER.....	9
FIGURE 8.	OVERVIEW OF THE COMPLETE WORKFLOW.	10
FIGURE 9.	PIPELINE OF THE GEOMETRY-BASED CLASSIFICATION	12
FIGURE 10.	GOOGLE EARTH IMAGE OF THE INPUT BUILDING (LEFT). POINT CLOUD IN PLY FORMAT, VISUALIZED IN CLOUDCOMPARE (RIGHT).....	13
FIGURE 11.	POINT CLOUD COLOURED AS DISTANCE TO 3D MODEL.....	15
FIGURE 12.	GEOJSON OUTPUT.....	16
FIGURE 13.	FINAL CLUSTERISATION WITH 3 GROUPS	17
FIGURE 14.	PIPELINE OF THE UNSUPERVISED IMAGE CLASSIFICATION	18
FIGURE 15.	FOOTPRINT OF A PROPERTY IN THE BAG.	19
FIGURE 16.	SECOND REQUEST TO GET THE COMPLETE BUILDING OF A PROPERTY	19
FIGURE 17.	IMAGE FOR A SINGLE BUILDING TAKEN FROM WMS.....	20
FIGURE 18.	MISALIGNMENT BETWEEN BAG FOOTPRINT AND AERIAL IMAGE.....	20
FIGURE 19.	ALIGNMENT STEP OF BAG FOOTPRINT AND AERIAL IMAGE THROUGH CANNY EDGE DETECTION	21
FIGURE 20.	A SIMPLIFIED VERSION OF THE CONVOLUTION TREE WITH DECONVOLUTION BRANCHES	23
FIGURE 21.	PREDICTED GROUND LEVEL SOLAR RADIATION USING PIX2PIX GAN.....	24
FIGURE 22.	PIPELINE OF THE SUPERVISED IMAGE CLASSIFICATION	24

FIGURE 23. AREA OF 1,5KM ² IN DELFGAUW, IN WHICH SOLAR PANELS WERE LABELLED.....	25
FIGURE 24. NATIONAL PDOK (LEFT) AND MAP.DATA.AMSTERDAM.NL (RIGHT) AERIAL PHOTOGRAPHY.	26
FIGURE 25. EXAMPLES OF SOLAR PANEL DETECTION MODEL TRAINING IMAGE PAIRS.....	26
FIGURE 26. EXAMPLE OF NEW IMAGES GENERATED THROUGH IMAGE AUGMENTATION.....	26
FIGURE 27. EXTRACTED TRAINING TILE EXAMPLES	27
FIGURE 28. GRAPH OF BINARY CROSS-ENTROPY LOSS DURING THE SOLAR PANEL MODEL TRAINING EPOCHS.....	28
FIGURE 29. SOLAR PANEL DETECTION	28
FIGURE 30. RESULTS FROM THE EARLY STAGES OF TRAINING ON LUCHTFOTO	29
FIGURE 31. COMPLETE GRAPH OF THE SOLAR PANEL DETECTION MODEL	30
FIGURE 32. MERGING OF THE DIFFERENT METHODS' RESULTS.	31
FIGURE 33. SOLAR PANEL TEST PIPELINE.....	32
FIGURE 34. FINAL CITYJSON, SEMANTICALLY ENHANCED WITH THREE NEW ATTRIBUTES.....	33
FIGURE 35. COMPARISON BETWEEN THE OBSTACLE DETECTION METHOD WITH MANUAL ALIGNMENT AND THE GROUND TRUTH WITH THE IMAGE OF THE RESPECTIVE BUILDING. ...	35
FIGURE 36. COMPARISON BETWEEN THE OBSTACLE DETECTION METHOD WITH AUTOMATIC ALIGNMENT AND THE GROUND TRUTH WITH THE IMAGE OF THE RESPECTIVE BUILDING. ...	35
FIGURE 37. FULLY AUTOMATED HISTOGRAM ACCURACY.....	36
FIGURE 38. ALIGNMENT GIVEN HISTOGRAM ACCURACY.	36
FIGURE 39. SELECTED SOLAR PANEL INFERENCE RESULTS (PART 1/2).....	37
FIGURE 40. SELECTED SOLAR PANEL INFERENCE RESULTS (PART 2/2).....	38
FIGURE 41. SELECTED SOLAR POTENTIAL INFERENCE RESULTS	39
FIGURE 42. SOLAR PANEL DETECTION MODEL CONFUSED BY THE UNUSUALLY BLUE ROOF COLOUR.....	42
FIGURE 42. RESULTS FOR THE BRINK PROVIDED 50 BUILDING DATA SET (PART 1/2).....	54
FIGURE 43. RESULTS FOR THE BRINK PROVIDED 50 BUILDING DATA SET (PART 2/2).....	55
FIGURE 44. RESULTS OF DETECTING OBSTACLES WITH GEOMETRY-BASED AND IMAGE CLASSIFICATION.	56

List of Tables

TABLE 1. VISUAL ILLUSTRATION OF THE GEOMETRIC ALGORITHM'S STEPS.	15
TABLE 2. STRUCTURE OF THE CSV FILE RESULTING OF THE MERGING OF THE THREE METHODS.	32
TABLE 3. ERROR MATRIX FOR AN EXAMPLE BUILDING, USING A PERFECT ALIGNMENT.....	34
TABLE 4. ERROR MATRICES FOR AN EXAMPLE BUILDING, USING THE AUTOMATIC ALIGNMENT METHOD.	34
TABLE 5. OVERALL RESULTS ACCURACY.	36
TABLE 6. ACCURACY OF THE SOLAR PANEL DETECTION FOR THE 50-BUILDING DATASET PROVIDED BY BRINK.....	38
TABLE 7. MoSCoW METHOD OF REQUIREMENTS AND OBJECTIVES.	43
TABLE 8. CONNECTION BETWEEN THE PROJECT AND GEOMATICS COURSES.....	45

Abbreviations

AHN	<i>Actueel Hoogtebestand Nederland</i> = Current Height Model of the Netherlands
BAG	<i>Basisregistratie Adressen en Gebouwen</i> = Dutch national register for Addresses and Buildings
LoD	Level of Detail
PDOK	<i>Publieke Dienstverlening Op de Kaart</i> = Public Services on the Map
PV	Photovoltaic
WFS	Web Feature Service
WMS	Web Map Services

1 Introduction

1.1 Project introduction

In 2018 renewable energy in the Netherlands makes up 7.4% of the total final energy consumption (TFEC). According to set goals, this share is to be increased to 16% by 2023 and to 27% by 2030. Although the proportion of solar energy in the TFEC is still comparatively small, it shows the highest rate of increase together with wind energy (International Energy Agency, 2020). In 2020 alone, the Dutch solar market grew by 41% compared to the previous year (Dutch New Energy Research, 2021). Together with ongoing governmental incentives for solar energy production, this proves that no decreasing numbers are to be expected in the near future.

To determine if a roof is suitable for mounting PV panels, solar potential analysis and resulting solar maps or tools such as the *Zonatlas* (*Zonatlas*, 15.06.2021) are valuable resources. The solar potential of a rooftop first and foremost depends on the incoming solar radiation, which in turn depends on atmospheric parameters and the geometry of the building. Focusing on the latter, the right aspect and tilt of the roof are fundamental properties. However, the available area and shadow casting also play an important role in accurate predictions. Both of these properties are influenced by roof obstacles such as chimneys, dormers or roof windows (Machete *et al.*, 2018). Accordingly, Romero Rodríguez *et al.* (2017) show that the more detailed the input geometry, the more accurate the solar irradiance estimation.

In that context, the recently launched 3D BAG (*3D BAG*, 2021), which provides 3D building models up until Level of Detail (LoD) 2.2 for the entire Netherlands, holds a big potential. Yet, it describes a simplified model of reality which in some cases fails to correctly or consistently model small-scale obstacles. Furthermore, additional information about roof properties may be of interest to allow for an even better targeted solar potential analysis. Among them is the knowledge about already existing PV panels and the material properties of the roof, which is of interest for installation, maintenance, and thermal effects. However, these kinds of information are difficult to retrieve in a simple solar potential analysis workflow since they ask for various input data which in turn need to be analysed through separate pipelines. Thus, we provide such semantic roof properties as additional attributes by integrating them into existing 3D models, in order to simplify their incorporation in a more comprehensive solar potential analysis. The properties are automatically generated from openly available data in the Netherlands. Three self-contained approaches are developed for this and are eventually merged into a joint workflow. The final result is an enriched 3D model encoded in CityJSON.

1.2 Project context

This work is done in scope of the Synthesis Project of the MSc Geomatics program and is implemented in cooperation with the company Brink. Founded in 1972, Brink provides management and consulting services together with software products and financial advice for the construction, infrastructure, and real estate domain. Their clients vary from public to

governmental and the private sector. An integral part of their work is to automatically gather technical knowledge about proprietors' buildings as they still heavily rely on the work of inspectors to gather such information. Therefore, our project complements and is in line with Brink's already implemented solutions for inferring, for instance, roof segmentation, roof area or gross façade area. From the academic perspective, we are mentored by Ir. Edward Verbree who is specialised in positioning and location technologies (outdoor and indoor), point cloud processing and geodesy. Pim Klaassen and Brenda Olsen from Brink support us from a practical perspective.

1.3 Reading guide

The remaining report is structured as follows. Chapter 2 introduces the underlying theory of our research project and discusses related work. After that, Chapter 3 first describes the used input datasets before going into the details of the methodology and implementation of our algorithm. Chapter 4 subsequently shows and discusses the results together with the limitations of the implementation. Chapter 5 reflects on the overall progress and success of our project and shows the connection to the already completed Geomatics coursework. Finally, in Chapter 6, we draw the conclusion of our research and illustrate possible future work in this field.

2 Theoretical Background

This section introduces and discusses some of the concepts we use in our argumentation and workflow. The first part briefly explains the underlying theory of solar potential with a focus on the parameters needed for a solar potential analysis. The following part deals with how (roof) obstacles are detected in other approaches, and for which use cases this is needed. The last part shortly illustrates two ways of classifying images, which are later used in the implementation.

2.1 Solar potential

In a broad sense solar potential describes the suitability of a surface for the utilisation of solar energy. By far the most common and known use case is the installation of photovoltaic (PV) modules to generate electricity. The potential is usually presented as a solar map in a classic map format or as web-GIS applications (Kanters, Wall and Kjellsson, 2014). The underlying approaches and methodologies to calculate the solar potential vary greatly. Recently, research is conducted to model the solar potential through machine learning approaches (Gassar and Cha, 2021). However, in this part we focus on a GIS-based analysis. When modelling a large geographic extent, a terrain is usually the main input geometry (Suri *et al.*, 2020). For more accurate predictions in an urban environment, 3D geometries (namely point clouds and 3D models) are used to evaluate the potential on roofs and facades (Brito *et al.*, 2019). Whether a surface is suitable or not, mainly depends on the following parameters:

2.1.1.1 Solar Radiation

The incoming solar radiation on a horizontal plane is given in W/m^2 or kWh/m^2 and is composed of direct beam shortwave radiation, sky diffuse, and ground reflected radiation. To calculate the incoming radiation on a tilted plane, this value has to be corrected with the aspect and the slope of the surface (Thorpe, 2018). For calculating the solar potential, different solar radiation models exist which approximate the atmospheric conditions at a given location over the year (Freitas *et al.*, 2015). Figure 1 shows the average Global Horizontal Irradiation in the Netherlands over the time from 1994 to 2018.

2.1.1.2 Aspect and Tilt

The solar radiation, and thus the electricity gain is highest on a surface orthogonal to the incoming sun rays. Therefore, the right aspect and tilt of the surface is crucial for solar potential. Aspect describes the direction the surface is facing in degrees relative to north. The tilt is an angle between the horizontal plane and the surface. In the Netherlands, the ideal aspect is 135° , facing southeast. The ideal tilt is 37° . However, these are optimum values and also surfaces deviating from them can still be suitable (*Dutch PV Portal*, 14.06.2021).

2.1.1.3 Shadow Casting and Available Area

Shadows caused by roof obstacles and surrounding features such as neighbouring buildings and vegetation cause a lower incoming radiation and thus efficiency losses on the energy production through PV panels. The available area is furthermore influenced by roof obstacles such as chimneys or dormers and other distinct building features. While this can be difficult to include

in the calculation due to lacking precision of the input geometries, many approaches regard this by using fixed utilisation factors depending on the regional specifications and statistic evaluations. This factor is described through the total usable roof area divided by the total roof surface area and is usually based on statistical data (Yang *et al.*, 2020), (de Vries *et al.*, 2020). Another method based on image classification, extracts the information of available area for each building individually (Bergamasco and Asinari, 2011).



Figure 1. Global Horizontal Irradiation Netherlands. Source: (Solargis, 14.06.2021).

2.2 Obstacle Detection

Obstacle detection in the broad sense has many application areas, in which the understanding of what constitutes an obstacle also varies. Currently, one popular research field is real time obstacle detection for navigation purposes, driving assistance and autonomous driving (Li *et al.*, 2020), (Yoo *et al.*, 2016), (Prakash, Akhbari and Karam, 2019). However, in scope of this research we focus on roof obstacles such as dormers, chimneys, or roof windows (For an example, see Figure 2).

Some synthesis projects from previous years already dealt with obstacle detection in a smaller scope. One project in 2015 worked on creating a database of currently installed PV panels in a research area, including already registered panels, and detected ones through an algorithm made by Buro Karto. This algorithm uses aerial images and is based on the shape, the contrast and the colour of solar collectors (Aarsen *et al.*, 2015). Another project from 2020 detects and models square chimneys in the city of Amsterdam in order to monitor the compliance with chimney height regulations. For this, points in a point cloud above a modelled roof structure are filtered out and subsequently analysed with a rule-based approach (Alexandridis *et al.*, 2020).

Similar to these projects, most of the research found focuses on a specific type of obstacle only. In that sense, (Satari, 2012) proposes a way to recognise dormers based on Lidar data and an LoD2 model without detailed roof structures through support vector machine, a supervised machine learning model. The approach is based on the different normal direction of dormer points to the main roof points.

The only paper found that detects multiple sorts of roof obstacles is from Bergamasco and Asinari (2011). However, this is just a by-product in the bigger aim of exploring the available roof area through a single band image classification. Overall, it can be concluded that obstacle detection is not a vastly explored topic in research, which encouraged us to develop an innovative solution with this project.



Figure 2. Example building with roof obstacles in Delft. Screenshot from (Google Maps, 15.06.2021).

2.3 Image Classification

Classifying remotely sensed images can be useful for many different applications, for example for generating land cover maps or land use maps (Al-doski, Mensori and Mohd Shafri, 2013). Image classification refers to the grouping of pixels according to their spectral information and assigning them to nominal values/classes. In a general regard, two main methods of image classification that are differentiated are Supervised and Unsupervised image classification (Janssen, 2004).

Each pixel value can be described through a feature vector. All the feature vectors plotted in one graph build the feature space, similar to a scatter plot. In this feature space, specific features are clustered together in a certain area. These clusters relate to one class which represents one object or type (e.g., tree, water). To assign a pixel to a value, these classes first have to be found or defined. This happens in the training process. In this process lies the difference between supervised and unsupervised image classification. In supervised image classification, there is a ground truth available with which the clusters are built during the training process. In unsupervised image classification there is no such thing available, and the clusters have to be automatically found through a clustering algorithm. As a last step, the pixels of a given image are assigned to the previously defined classes according to their feature vectors. For this process different classification algorithms exist (Janssen, 2004).

3 Methodology & Implementation

This section starts with a brief overview of the input data used in the project, which is followed by a thorough explanation of the methodology. The methodology is divided into three distinctive parts, and each part is discussed with their theoretical aspects as well as how they are implemented. Finally, the steps taken to bring these parts together to be able to obtain the end product is discussed.

3.1 Used Datasets

In order to develop a generic and reusable method, the datasets used are openly available for the Netherlands. These include raw datasets such as point cloud data and aerial imagery but also processed datasets such as the Key Registers of Addresses and Buildings (BAG).

3.1.1 AHN3


The Actueel Hoogtebestand Nederland (AHN) is the digital surface model of the Netherlands which includes approximately eight height measurements per square meter, and the AHN3 is the latest version of this dataset with height measurements from the years 2014 to 2019 (AHN.NL, 2021). Both the grid and the point cloud versions of AHN3 are openly available in GeoTIFF and LAZ formats, respectively, which can be easily and quickly downloaded since the whole dataset is divided into tiles to ensure fast access and retrieval (PDOK, 2021).

3.1.2 BAG and 3D BAG

3.1.2.1 Dutch National Register of Addresses and Buildings (BAG)

All the building information of the Netherlands is stored in the BAG dataset (*Basisregistraties Adressen en Gebouwen*). This information is kept up-to-date and maintained by the municipalities who gather the information in a National Facility (BAGLV). The latter is managed and made publicly available by the Land registry Office (*Kadaster*) (Overheid, 2021). The BAG dataset contains each property's footprint with some attributes, including the buildings' status, the year of construction and the area. Each property is uniquely identified through its *identificatie* (identification ID) (Figure 3), which is used in our implementation to keep track of the buildings through different approaches.

Rotterdamseweg 248 Delft



Pand	
ID	0503100000033690
Status	Pand in gebruik
Bouwjaar	2005
Geconstateerd	Nee
Beigdatum	22-09-2005
Documentdatum	22-09-2005
Documentnummer	BWT/20121433
Mutatiedatum	12-07-2011

Figure 3. Example building from BAG. Source: BAG viewer, 16.06.2021

3.1.2.2 3D BAG

3D BAG is a dataset that consists of 3D building models of the Netherlands. It is automatically generated using the building footprints' geometry from the BAG and their height information from the AHN point cloud dataset. The algorithm consists of three main steps (Arroyo Ohori, Ledoux and Peters, 2021a):

1. Feature extraction: The first step is to detect the roof planes from the point cloud (AHN3) with a region-growing algorithm, and to obtain the boundary lines and the intersection lines (Figure 4) to be able to create a planar partition of the footprint.

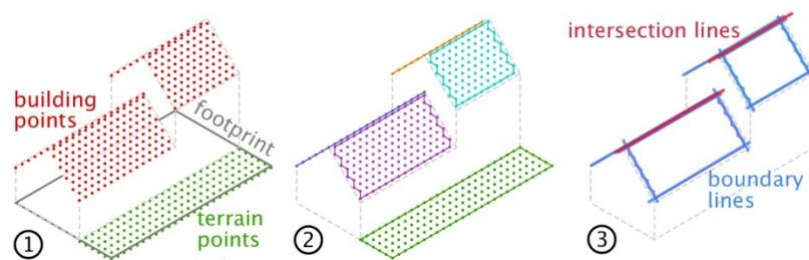


Figure 4. Feature extraction: (1) Input data. (2) Roof planes detected with region-growing algorithm. (3) Detected lines. Source: Arroyo Ohori, Ledoux and Peters, 2021a.

2. Construction of the roof-partition: This step consists of subdividing the footprint to create a planar partition, which is represented with a DCEL data structure.

3. Extrusion: The final step turns the planar partition of the footprint into a 3D building mesh by considering the topological relationships stored in DCEL, and the floor, roof, and wall faces are created.

Currently, it is possible to obtain the 3D models in three different levels of detail: LoD1.2, LoD1.3, and LoD2.2 (3D BAG, 2021). While LoD1 refers to a simple block model where the footprints of buildings are extruded to a given height, LoD2 gives more detailed information by adding the generalized roof shape instead of a horizontal roof (Figure 5) (Arroyo Ohori, Ledoux and Peters, 2021b).



Figure 5. The five LoDs. Source: Arroyo Ohori, Ledoux and Peters, 2021b.

Moreover, the five LoDs have been improved and given more detail at TU Delft to prevent ambiguous situations about the use of these generic LoDs. The improved LoDs, including the three options available in 3D BAG, and their differences in terms of 3D geometry can be seen in Figure 6.

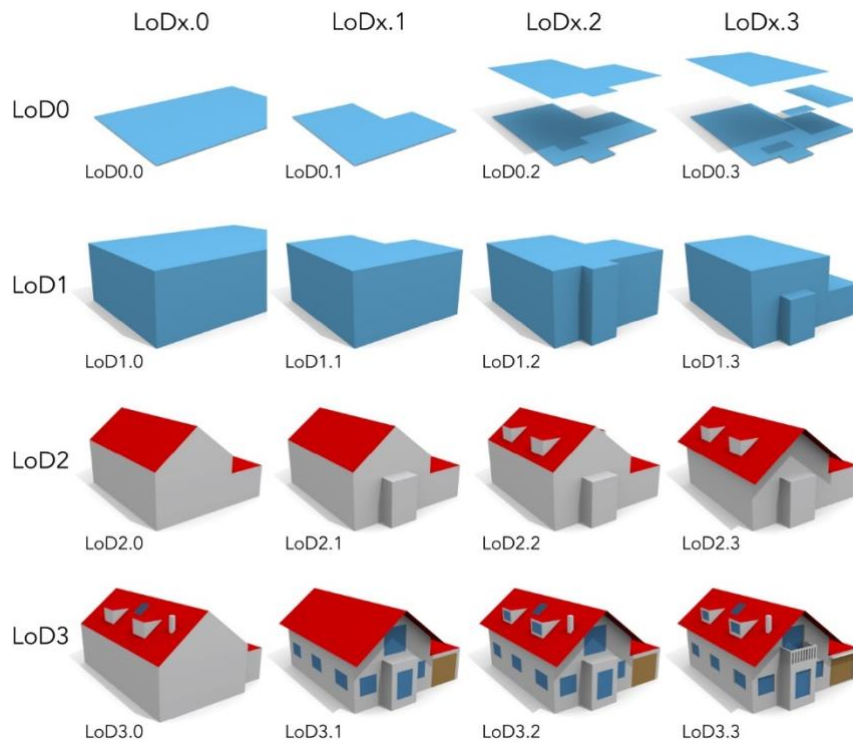


Figure 6. Improved LoDs. Source: Arroyo Ogori, Ledoux and Peters, 2021b.

Finally, it is possible to visualize and obtain attribute information using the 3D BAG Viewer (Figure 7) and/or the provided web services while there are also options to download specific tiles in CityJSON, OBJ and GPKG formats (3D BAG, 16.06.2021).

Attribute	Value
Tile number	5910
identificatie	NL.IMBAG.Pand.0503100000032914
h_maaiveld	0.002
h_dak_70p	21.499681
dak_type	slanted
pw_bron	ahn3
pw_datum	2013-12-01
val3dity_codes	[302]

Figure 7. Example building from 3D-BAG viewer. Source: 3D BAG, 16.06.2021.

3.1.3 Aerial images (Luchtfoto - PDOK)

The orthophoto for the whole Netherlands is a set of smaller orthophotos made available on PDOK as *Luchtfoto* (PDOK, 16.06.2021). The aerial images used for our implementation are taken in summer 2020 and have a nationwide resolution of 25cm.

3.2 Overview of the whole workflow

Figure 8 shows the workflow diagram that includes the steps of all three methods as well as how they are merged to obtain the end product. All codebase for the project is stored in a *GitHub* repository (https://github.com/iapra/Synthesis_project).

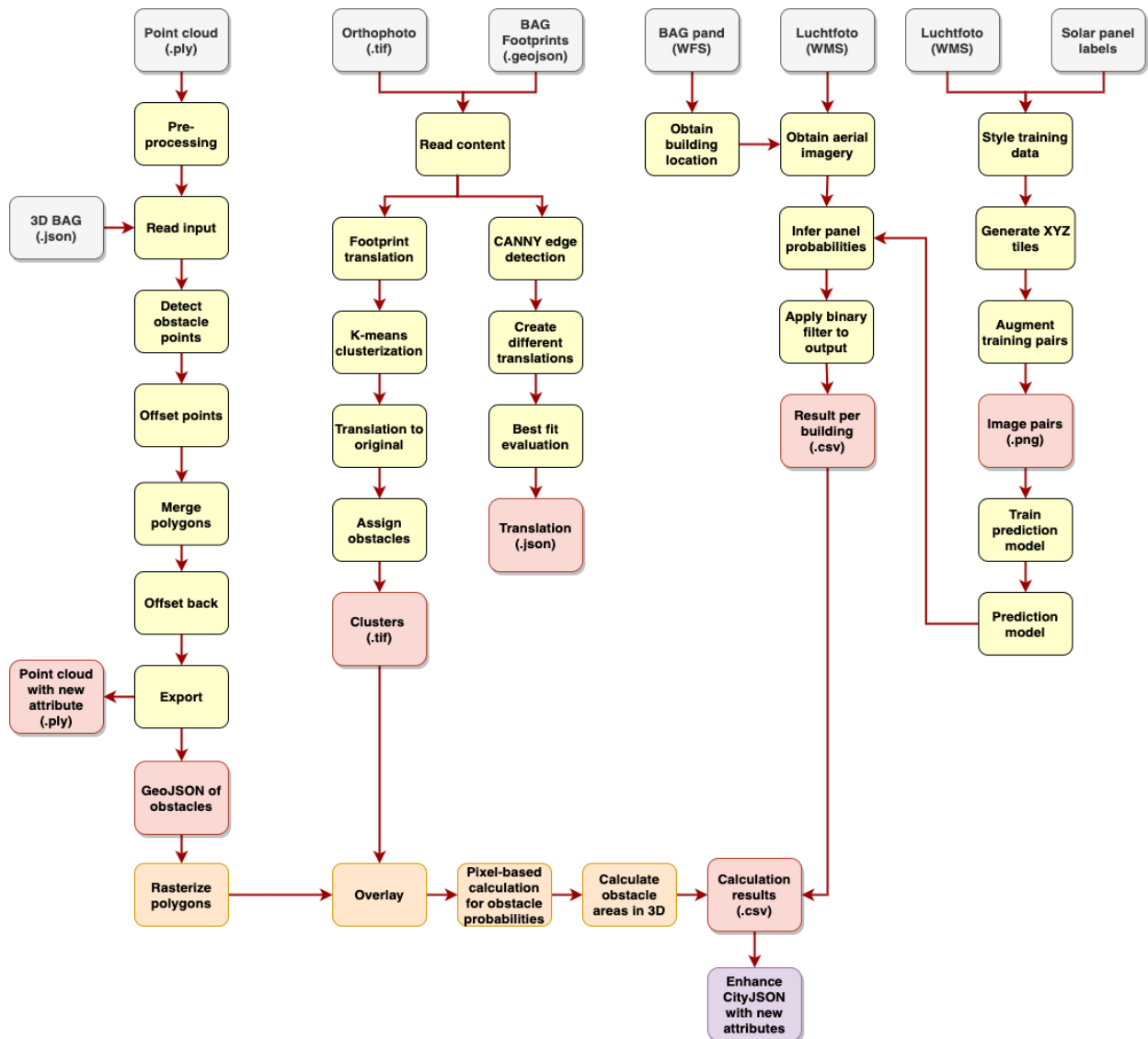


Figure 8. Overview of the complete workflow.

3.3 Part I: Geometry-based classification

3.3.1 Methodology

The first part of our algorithm is based on the buildings' geometrical representations and developed on 3D datasets openly available for the Netherlands: the AHN3 point cloud dataset and the 3D BAG model. The aim is to couple and compare these two in order to identify their differences. Indeed, the potential geometries' mismatches would correspond to the relief of some installations present on the roofs (chimneys, solar panels, windows...), which are referred to as "roof obstacles".

On the one hand, the 3D BAG dataset is used for several applications such as energy analysis and various simulations which results from raw data processing by simplifying some buildings (3D BAG, 16.06.2021). The original AHN3 point cloud dataset, used as input, is actually closer to the reality since emanating from the acquisition phase. As a result of the 3D BAG algorithm, not all buildings' details are depicted in the output CityJSON model, which might lead to a wrong interpretation from the data user. For instance, some roofs could be considered free of obstacles and available for solar panels' installation, although they are not. On the other hand, point cloud data is a precise and raw dataset, but with the disadvantage to be unstructured, discrete, and heavy. For this reason, we combine the structured 3D data model obtained by 3D BAG and compare it to the original raw point cloud data to underline the omitted installations present on the rooftops.

More precisely, the distance between the points and the 3D model's roof mesh is evaluated: if it is bigger than a given threshold, the point is considered an obstacle. Some additional experimentally defined parameters allow us to refine the detection: the points belonging to vertical surfaces are discarded (based on the orientation of their normal) as well as the isolated ones. In a next step, we aim at combining points belonging to the same obstacles and represent their surface in 2D. Since clustering methods do not give promising results, we use another approach: the points are offset with a size based on the precision of the AHN3 dataset (about 40 cm) and the overlapping shapes are merged together. Thanks to this method, long obstacles such as pipes or bodyguards, are correctly represented as one feature. After reversing the offset to obtain more realistic surfaces, the obstacle polygons are stored in 2D and used as input for the merging of the two first methods' results.

The following flowchart gives a detailed overview of the just described process of the geometry-based classification.

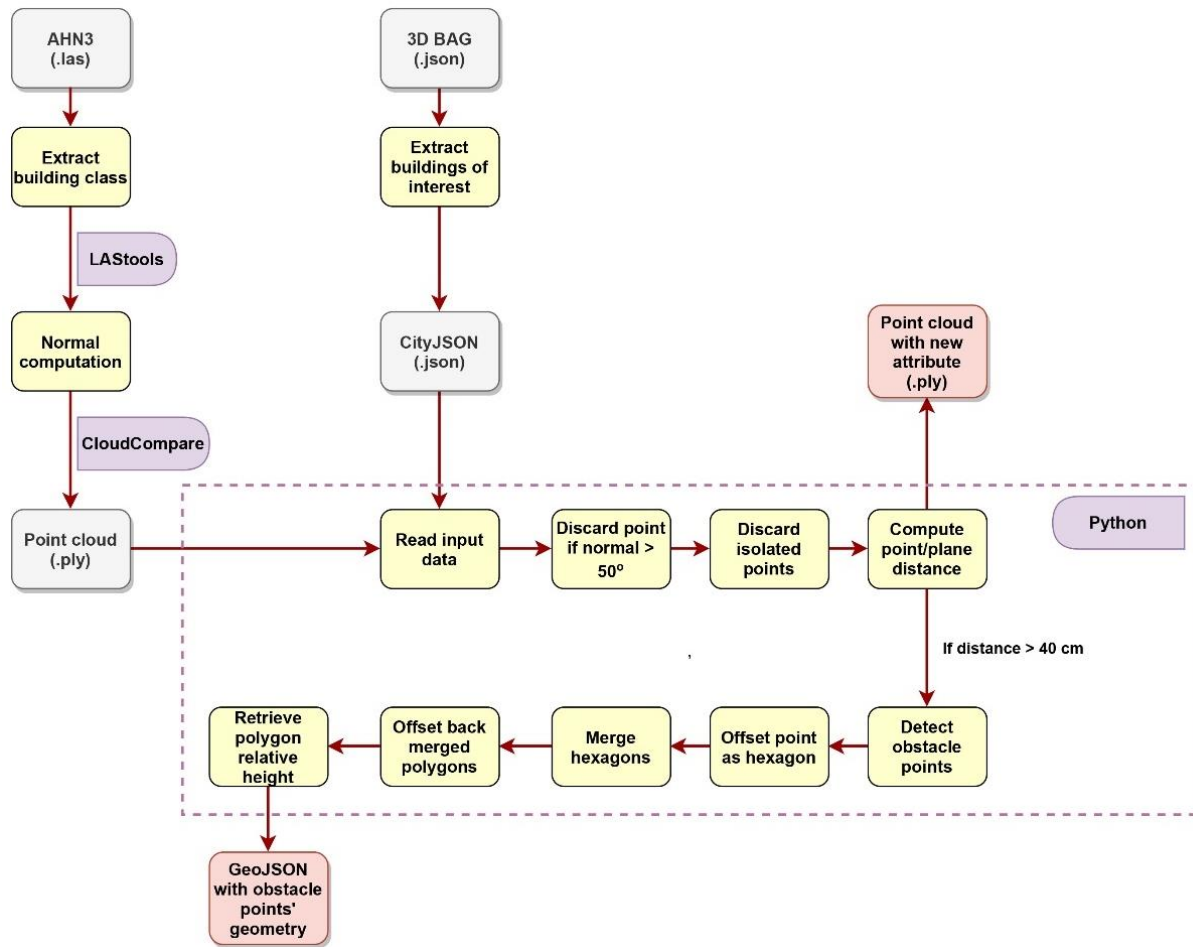


Figure 9. Pipeline of the geometry-based classification

3.3.2 Input Data and pre-processing

For this part, the input data consists of the point cloud extracted from AHN3 and the corresponding CityJSON buildings obtained from the 3D BAG. Both are pre-processed as follows.

3.3.2.1 AHN3 point cloud

First of all, the point cloud tiles covering the area of interest are downloaded from PDOK (*PDOK*, 16.062021). LAStools software is used to filter the points belonging to the “building” class while the rest of the points are discarded. The file is then converted from LAS to PLY format, since we prefer this format for its flexibility with the “composition of the point record” (Arroyo Ohori, Ledoux and Peters, 2020). Optionally, only the points corresponding to the desired buildings are kept, while discarding the others. This step is not necessary but reduces the computation time later on. However, this requires an advanced algorithm that Brink already has implemented. The results are stored in a database that can, among others, be queried using the buildings’ *identificatie* number as indicated in BAG registries. Thanks to this data, we could integrate our work to the existing pipeline of the company since they provide us with this pre-processed point cloud containing only the AHN3 buildings’ points we are interested in (Figure 10).

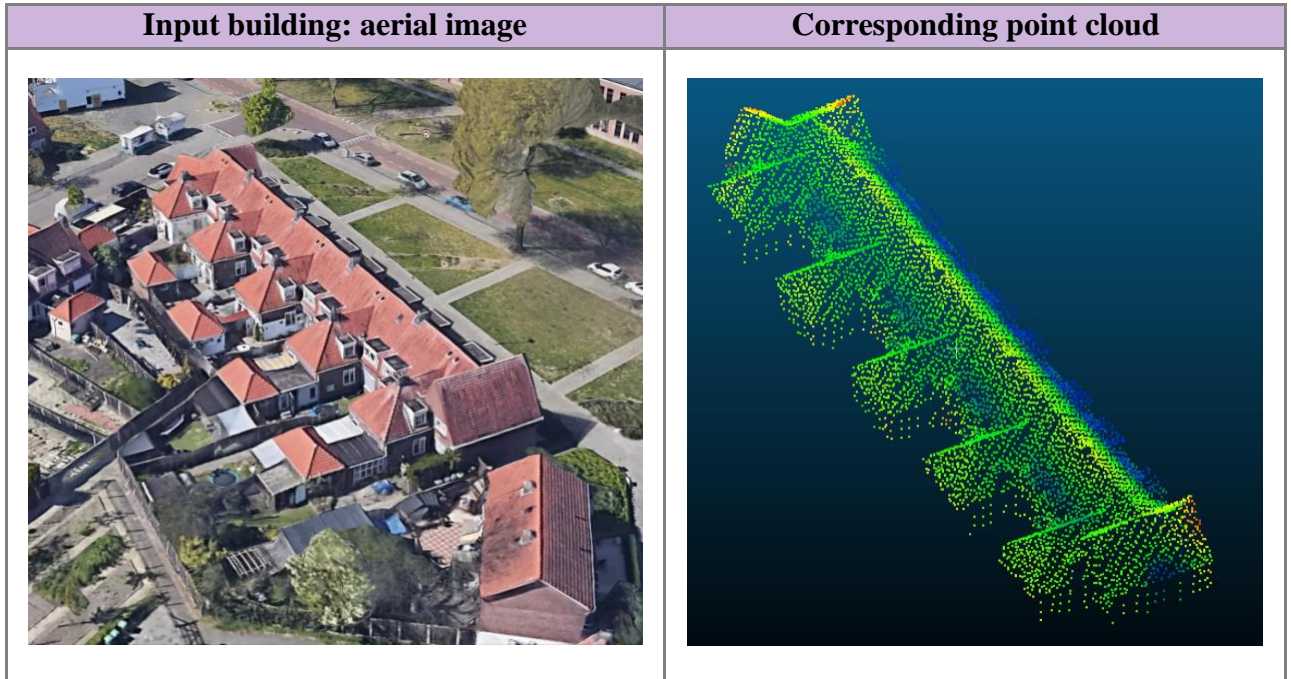


Figure 10. Google Earth image of the input building (left). Point cloud in PLY format, visualized in CloudCompare (right).

Secondly, by using the open-source software CloudCompare the points' normals are computed, added to the point and saved to the point cloud in PLY format. CloudCompare uses a C++ implementation is not worth converting to python in the scope of this project, considering the fast and good results obtained with the software. To compute the normal vectors for each point, an experimentally determined user-defined neighbourhood radius of 5 is chosen.

Finally, the point cloud input for the geometry-based classification should contain the x,y,z coordinates but also each point's normal vector values (nx, ny, nz) and be stored in PLY format. An exemplary PLY file with the input requirements is illustrated below.

```
ply
format ascii 1.0
comment Created by CloudCompare v2.12 alpha
comment Created 6/11/2021 2:24 PM
element vertex 8728
property double x
property double y
property double z
property float nx
property float ny
property float nz
end_header
159132.822998 384390.138977 22.615999 -0.527856 -0.192085 0.827328
159132.887024 384389.862976 22.841000 -0.455005 -0.217742 0.863458
159132.953979 384389.934998 22.853001 -0.473723 -0.236100 0.848436
159132.968018 384395.702026 21.065001 0.979742 0.015181 -0.199685
159133.018005 384389.512024 23.268000 -0.440382 -0.243648 0.864118
159133.026978 384395.611023 22.749001 0.861529 -0.169360 -0.478627
159133.148987 384395.541016 22.889999 0.855873 -0.179481 -0.485044
159133.169983 384389.505981 23.347000 -0.351370 -0.268346 0.896956
159133.190979 384390.239014 22.844999 -0.486268 -0.243901 0.839080
...
```

Example PLY file

3.3.2.2 3D BAG

Since our algorithm benefits from the buildings' attributes, specifically their *identificatie (ID)*, the 3D model of the buildings is downloaded in CityJSON format. The 3D BAG dataset is divided into tiles to ensure fast retrieval of data; therefore, the tiles including the buildings of interest are downloaded from the 3D BAG Viewer. At this point, there are two options to use these tiles as input data. Firstly, the downloaded tiles can be merged into one single CityJSON file that contains all the input buildings. Secondly, this single CityJSON file can be further processed and filtered to keep only the input buildings to decrease the file size. Both, the merge and filter operations can be performed with *cjio*, which is a Python CLI for the processing of CityJSON files (Ledoux, 2019). In order to speed up the algorithm, we decide to use the second option: we first merge all the tiles in a single file, then filter it by the *CityObject* IDs to keep only the relevant buildings.

3.3.3 Implementation

As the whole pipeline, this part is written in python, using *Shapely* library for most geometrical processes. In the *main* file, the input PLY and JSON files are first read to store their content in arrays. Only the roof planes in the relevant LoD of the CityJSON file are extracted. Then, the processing file *roof_obstacles* is applied, containing successive steps.

First of all, the detection of obstacle points is achieved by computing the distance between the mesh and the points. For this, we loop through the triangular mesh and consider only a subset of points situated above it. The points below the mesh are discarded as well as the one belonging to a vertical surface, that is, if the angle between the vertical axis and its normal is greater than 50 degrees. For the remaining points, when its distance to the mesh is greater than 40 cm, it is considered as an obstacle. This parameter is experimentally determined and gives the best results. Additionally, isolated points are discarded, using a radius search method to find the number of neighbouring points.

Next, the points are considered in 2D. Since we did not obtain satisfying results by clustering the points into obstacles, we form the obstacle shapes by offsetting the points as polygons and merging them together. For practical matters and since it is a close equivalent to circles, the points are offset as hexagons. The overlapping ones are merged, constituting together one obstacle. An offset back is performed to obtain more realistic surfaces (Table 1).

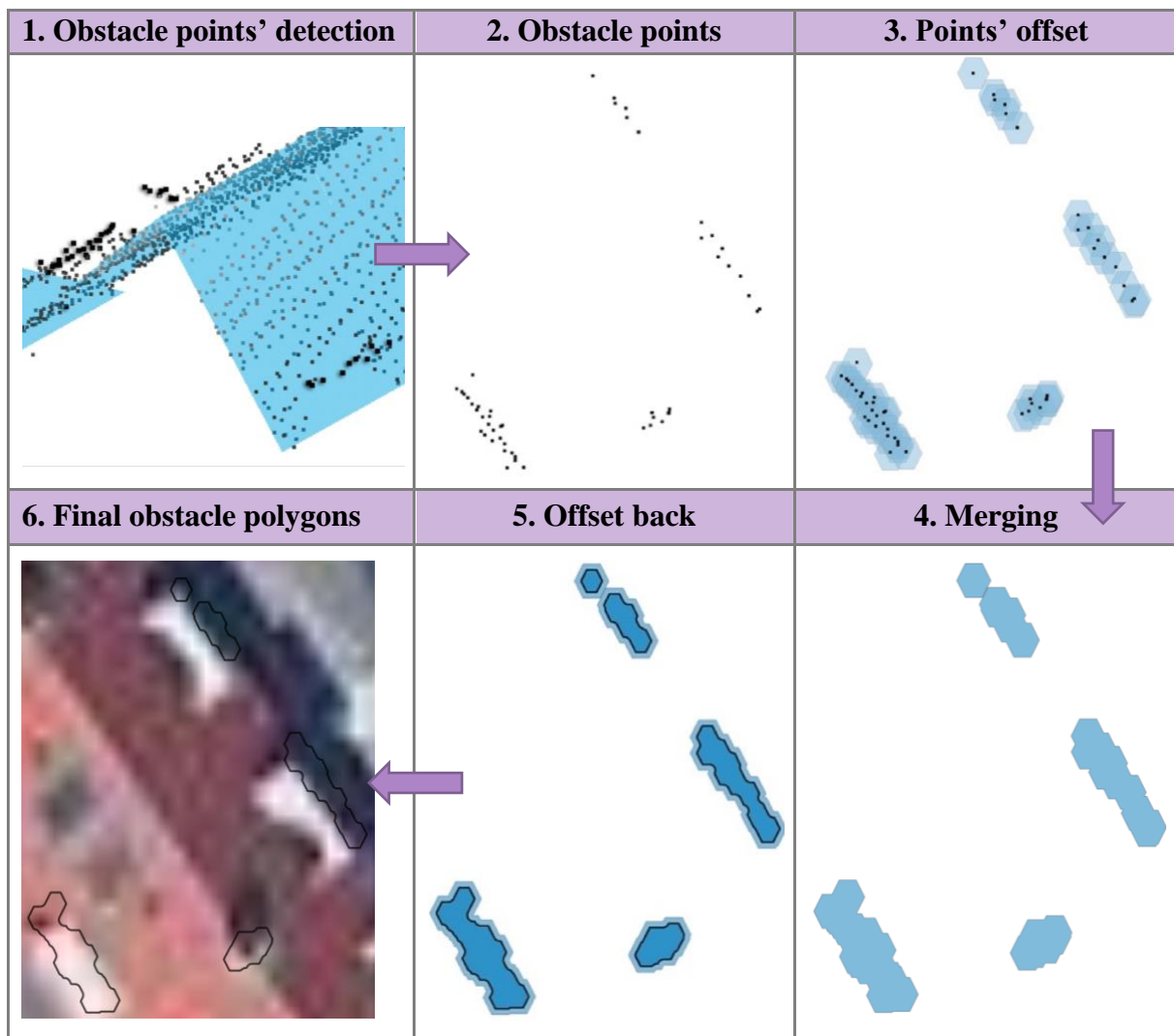


Table 1. Visual illustration of the geometric algorithm's steps.

Finally, the results are exported. First of all, a point cloud in PLY format is stored, containing a new attribute corresponding to the distance between the point and the 3D-BAG model. This value can efficiently be used by a future user in order to retrieve the obstacle points in their raw format. Since the distance is stored and not a Boolean attribute, the users can define themselves what should be the distance in order for the point to be considered as an obstacle (Figure 11).

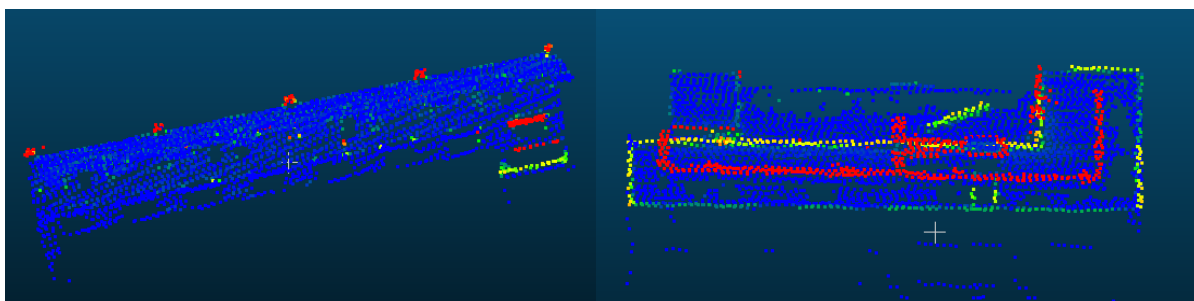


Figure 11. The colours are function of the distance of the point to its corresponding 3D-BAG plane. The blue points have a value of zero and are surely no obstacles. The red points are the furthest ones. The chimneys are for instance clearly distinguished. (Visualisation on CloudCompare).

Last but not least, the polygons are exported in GeoJSON format, which is the input of the next step in the pipeline. Each polygon keeps track of the building it belongs to through its *identificatie* value (Figure 12).

CityObject	Identificatie
6523169	0772100000295738
6523169	0772100000295738
6423576	0772100000302910
6423576	0772100000302910
6423576	0772100000302910
6423576	0772100000302910
2730039	0772100000297597
2730039	0772100000297597
...	...



Figure 12. GeoJSON output, storing the obstacle polygons' geometry, the buildings' *identificatie* and their *CityObject* id from the CityJSON format.

3.4 Part II: Unsupervised image classification

3.4.1 Methodology

The second part of the project is to detect the obstacles of a roof through an unsupervised image classification. Herewith, two open datasets from the Netherlands are used: the aerial images and the BAG footprints. The aim of this approach is to classify the roofs in two parts, the obstacle-part and the main roof part, based on the RGB pixel value of the aerial image which is clipped with the BAG footprints.

However, the aerial images taken by the *Kadaster* are not true orthophotos, but slightly oblique. Thus, the buildings are displayed tilted, covering not only the roof but also parts of the façade (Figure 18, red lines). Since the objective of the classification is to classify the roof, an alignment of the footprints with the corresponding roofs on the aerial images is required beforehand. Otherwise, the façades would distort the results. This pre-processing step allows us to retrieve the translation value to apply on each BAG footprint and is described in detail in the corresponding section.

Once the BAG footprint is translated, a clip between the footprint and the aerial image is performed to have only the pixels of the roof. Since the image has three values for each pixel, red, blue, and green, the classification is performed with a 3-dimensional feature space of the image. Three clusters are detected using: the obstacle area, the main roof area and the pixels out of the property building (Figure 13).

Then, the assignment of the building groups semantics is made. The out bounds area is easy to identify since it corresponds to the one containing the most pixels, due to the buffer previously

performed. For the remaining two classes, it is assumed that the second largest area will be the main roof and the third the obstacles' area.

Finally, the clustered image is translated back to its original position to be further processed and combined with the geometry-based classification.

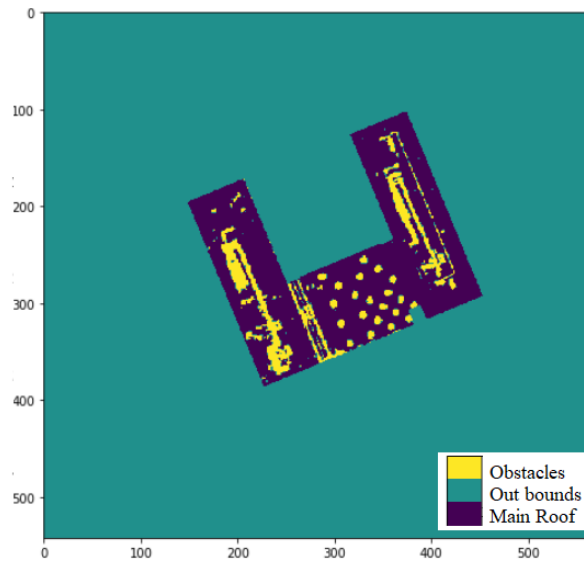


Figure 13. Final clusterisation with 3 groups. Out bounds, main roof and obstacles.

The following chart provides an overview of the pipeline of the image-based classification part.

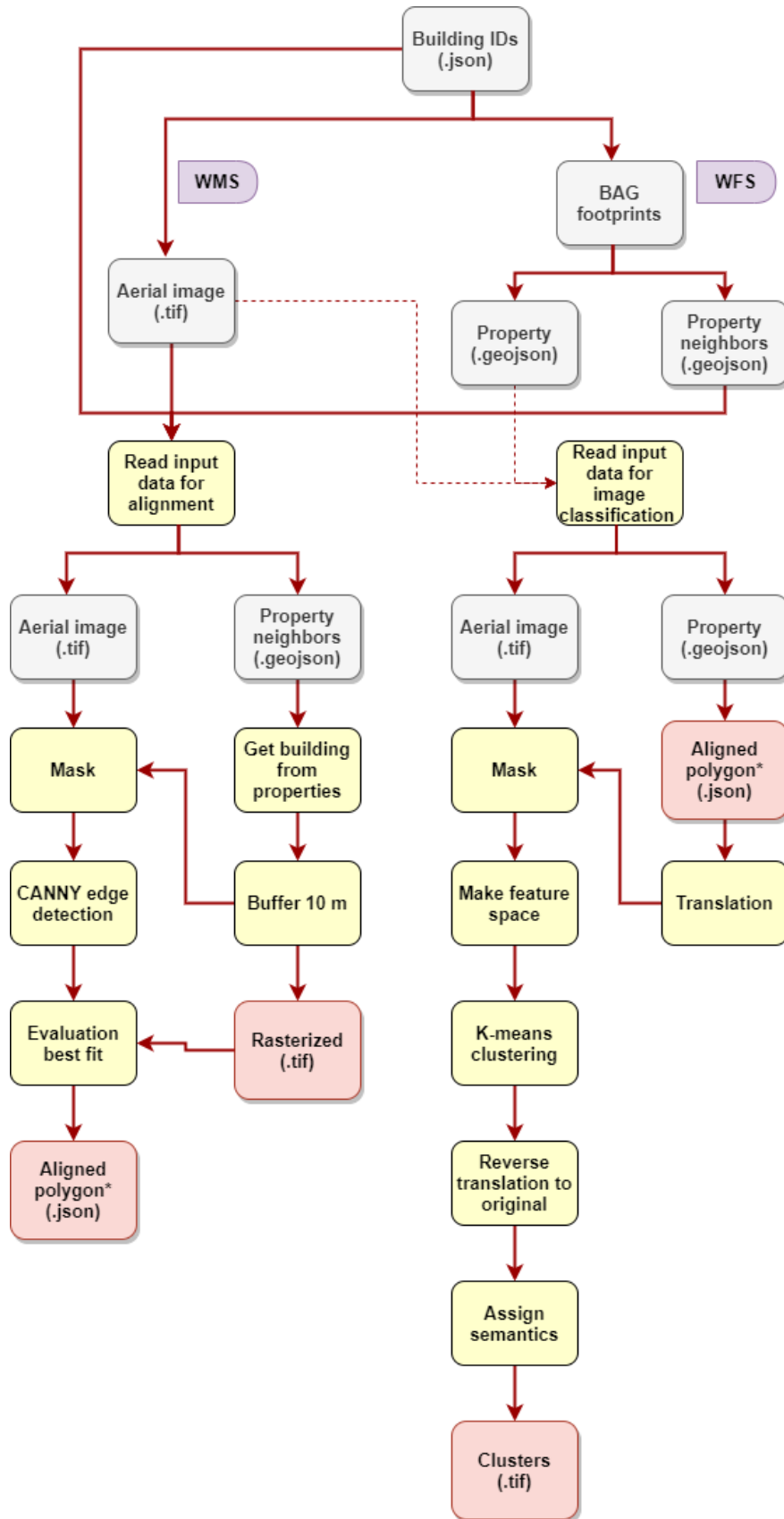


Figure 14. Pipeline of the unsupervised image classification

3.4.2 Input Data and pre-processing

3.4.2.1 Key Registers Addresses and Buildings (BAG)

One of the set requirements for this work is to use open datasets only. For the unsupervised image classification, two datasets are used, both available through *PDOK* web services (*PDOK*, 16.06.2021). The data is retrieved by the Web Feature Service (WFS) of the BAG. This service has the filter encoding to query the buildings through their *identificatie*, allowing us to retrieve only the buildings of interest.

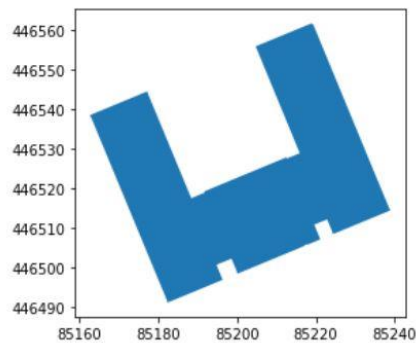


Figure 15. Footprint of a property in the BAG.

For the alignment purpose, we need to visualise clear edges for the edge detection algorithm to work efficiently. For this, since one property can be a part of a bigger building block (Figure 16), the bounding box of each of these buildings is stored. Then, another WFS request is sent, but with a bounding box filter to obtain all the neighbouring properties (Figure 16).

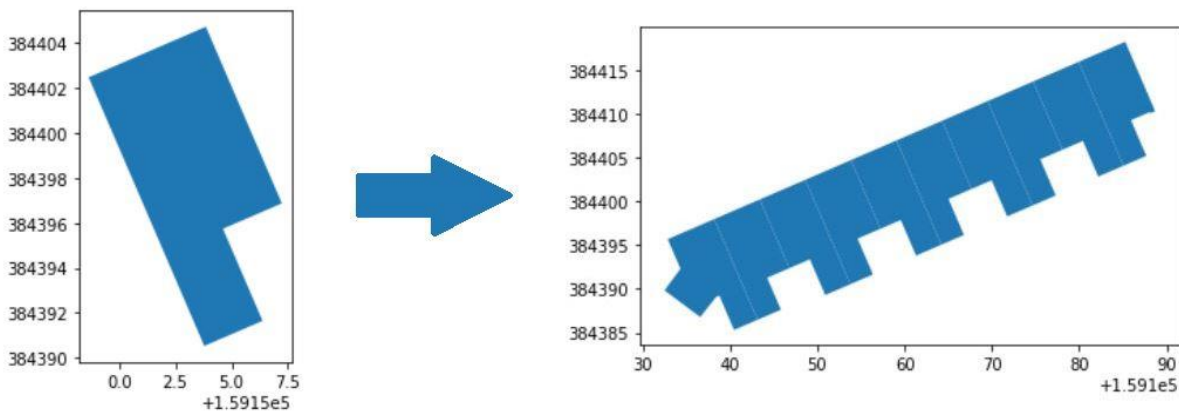


Figure 16. Second request to get the complete building of a property.

3.4.2.2 Aerial image (Luchtfoto)

The aerial image is retrieved through a Web Map Service (WMS).

Only the extent of the image around the desired buildings is needed. For this sub-selection, the *identificatie* of the BAG buildings is used. Their bounding box taken from the BAG's WFS is used, following a request with this bounding box plus a 25-meter buffer. As a result, we obtain a centric image of each building - with a resolution of 25cm - in which the clusterisation is computed.



Figure 17. Image for a single building taken from WMS.

3.4.2.3 BAG and aerial images alignment

As already mentioned, the BAG footprints are not perfectly aligned with the roof surface. Therefore, the BAG footprint is translated in such a way that it covers only the roof pixels of the image (Figure 18, green lines). Consequently, the first step is to prepare the input data for the alignment. Since the input data is a property that can be part of a bigger buildings' block, the neighbouring properties are firstly retrieved, with a bounding box augmented by a buffer. All the buildings recovered within this bounding box are then dissolved to obtain a common polygon out of properties having shared borders (Figure 18, left). Finally, a spatial join is performed to query only the building of the property of interest.



Figure 18. Misalignment between BAG footprint and Aerial Image. A property that compounds a building by itself (left), a group of properties that compound a single building (right).

The second step is to compute the alignment that best fits the building. For this, a 10 meter buffer is performed to the BAG footprint. Then, the image is clipped using this buffered footprint. This allows to focus and limit the following step to the area of the building of interest. Next, the main edges of the images are detected by Canny edge detection algorithm (Figure 19) and different footprints of the building are made with translations steps of 25 cm (resolution of the aerial images) in both x and y direction with a maximum translation of three meters. Thus, a total of 1861 polygons with different translations are made for each building. Finally, each polygon is rasterized with an 8-connectivity to get more understandable lines in a raster.

To evaluate the fitting of each alignment polygon, a map algebra calculation is performed between the Canny edge image and the rasterized alignment polygon. The calculation consists of the intersection pixels between the images. At the end, the alignment with most intersected pixels is chosen as the right alignment and the values of the translation are stored (Figure 19).

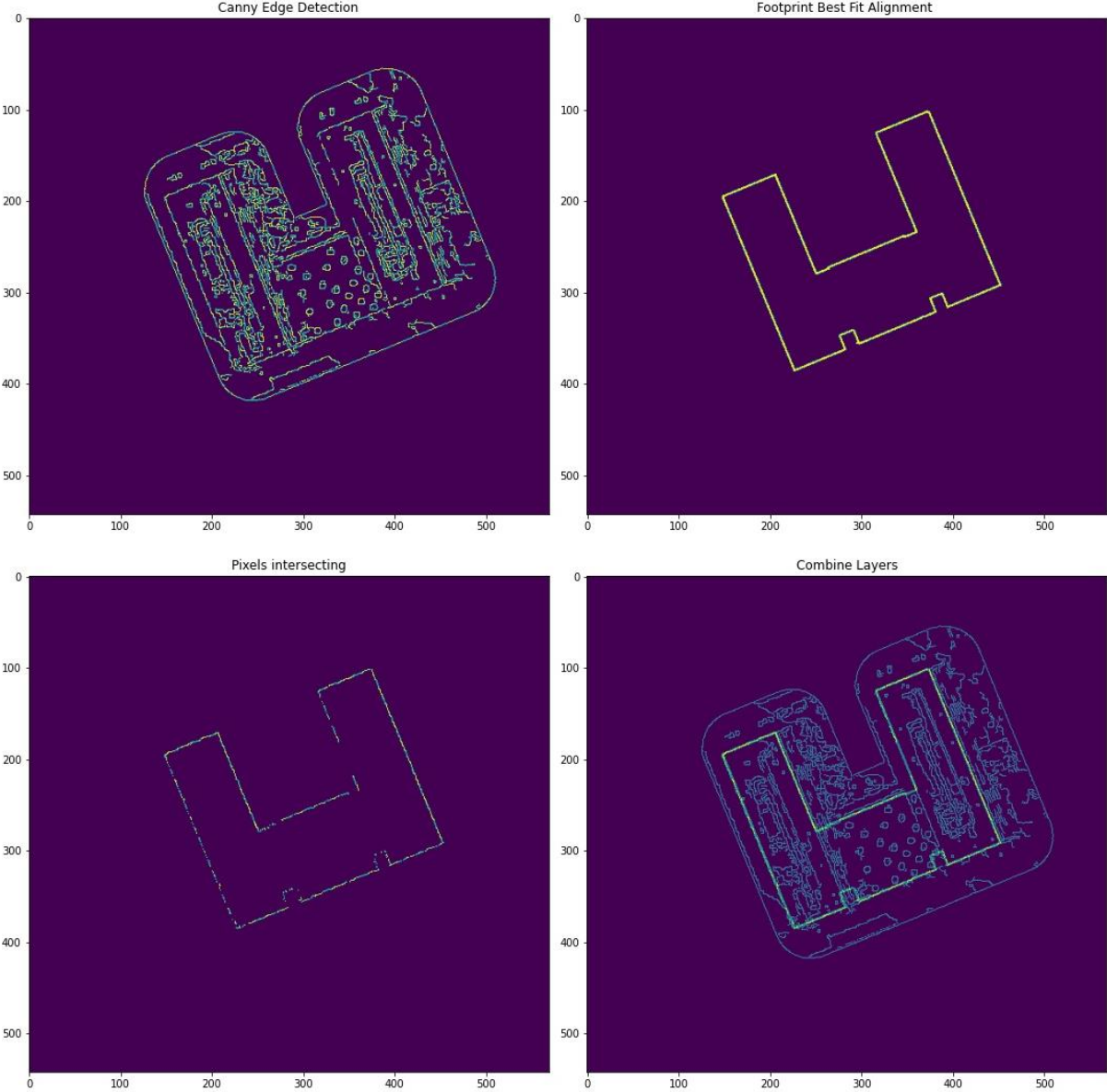


Figure 19. Alignment step of BAG footprint and aerial image through Canny edge detection

3.4.3 Implementation

First, the images and footprints for each property are retrieved. For this purpose, a *python* file is made with three functions. One, for retrieving the BAG footprint through a filter encoding by *identificatie* from the WFS. The second, also for retrieving the BAG footprint, but now through a bounding box from the *identificatie* to get the neighbouring properties. The last one

is for retrieving the aerial image of the property through a WMS request with a bounding box. The footprints are stored in a *GeoDataFrame* from *Geopandas* in a gejson format, whereas the images are stored as *GeoTIF* and processed in *rasterio*.

Secondly, for the alignment of the BAG footprint, a *python* file is implemented using the building's footprints and the images. Next, a buffer on the footprint is made and clipped with the image using the *mask* module from *rasterio*. On this clipped image and with using the *OpenCV* library, the edge detection is performed by means of the Canny algorithm. Different alignments are generated and stored into a *GeoDataFrame* and then rasterized into a grid. Finally, the best fitting alignment is selected, and the values of the alignment are stored in a *json* file with its property *identificatie*.

Furthermore, the image classification is also implemented in *python* and three inputs are used: the BAG footprint of the property, the image centred on the property and the alignment values previously calculated. Firstly, the BAG polygon is translated to fit the image. Then, the image is clipped with the footprint and the values outside the footprint are changed to *no data*. Then, three arrays are collected from the image. These arrays correspond to green, red, and blue values, respectively. The following step makes the 3D feature space from these arrays by flattening them into one. Having only one array, the library *Scikit-learn* is used to perform a k-means clusterisation with three clusters as input parameter. The k-means algorithm is used because it has to detect the three following classes: obstacle area, main roof area and pixels out of the property building.

Since the main roof's part is usually of different colour than the obstacles - such as chimneys, solar panels, windows, and dormers - and since the number of clusters is known, this algorithm is preferred. Indeed, it creates groups by solving an optimisation problem, minimizing the total pixels' Euclidean distance to their centroids, which are iteratively generated (Arroyo Ohori, Ledoux and Peters, 2020). In addition, the time computation is faster than other methods.

Finally, the result is reshaped to the same shape of the original image and translated back into the original footprint and saved as a *GeoTIF* using *rasterio*.

3.5 Part III: Supervised image classification

3.5.1 Methodology

The third part of the project focuses on exploration of use of supervised machine learning methods to identify additional roof features, especially solar panels and infer additional semantics from openly available data such as aerial photography or raster DSM.

The main principle of supervised machine learning, and the way it distinguishes from unsupervised methods, is that model is first provided with set of desired input – output pairs. The model then iteratively optimizes its parameters to find a best fit for a function that will map the provided inputs to outputs as accurately as possible. The assumption is that the mapping function will generalize to new input – output pairs the model has not encountered within training data.

In the last decade, neural networks have been gaining in popularity and basically dominated the field of supervised machine learning, especially when it comes to image-based learning methods. In our research we experimented with simple CNN (convolutional neural network) based models, but also more advanced architectures like GAN (generative adversarial networks), which combine multiple CNNs into one model.

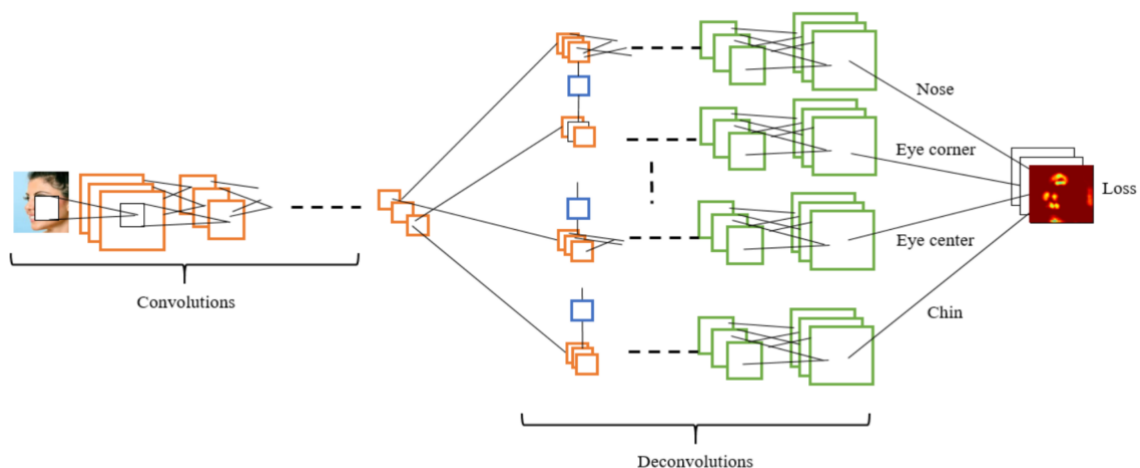


Figure 20. A simplified version of the convolution tree with deconvolution branches. Source: (Kumar and Chellappa, 2017).

CNNs use so called convolutional layers to compress the input matrix (image) into a denser representation by multiplying the matrix with a weighted kernel filter. This allows for the network to learn to summarize the input data into a map of detected features, which can get progressively more abstract with each additional convolution (i.e., pixel – edge – shape – object). The resulting feature map can then be upscaled back to the original input resolution using so called deconvolutional layers. This allows for e.g., providing a per-pixel map of features detected in the original input image (see Figure 21). This makes CNNs well suited for our goal of detecting solar panels in aerial imagery (see Figure 22 for our proposed pipeline).

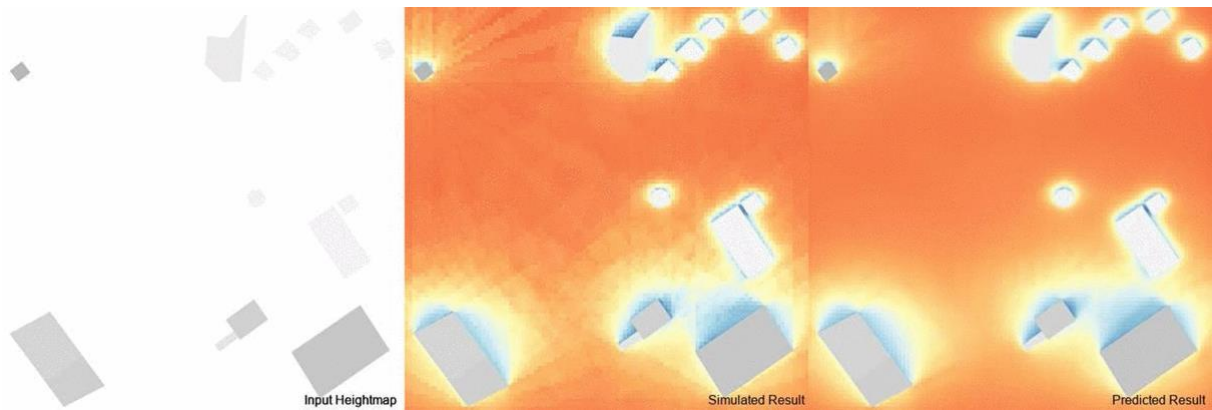


Figure 21. Predicted ground level solar radiation using Pix2Pix GAN. Source: (Chronis et al., 2020).

The more advanced architecture of GAN utilizes two competing CNNs to further improve the quality of CNN driven image generation and image to image translation. In GAN, one network fulfils the role of a generator, translating input into synthetic outputs as close as similar as possible to the training data. The other network fulfils the role of the discriminator, estimating whether the input it is given is the real data, or a synthetic output of the generator network. GANs have been recently used to predict solar radiation or wind speeds at ground levels from LOD1 DSM models (Chronis *et al.*, 2020) and seemed well suited for our goal of exploring solar potential and inferring additional more related roof attributes.

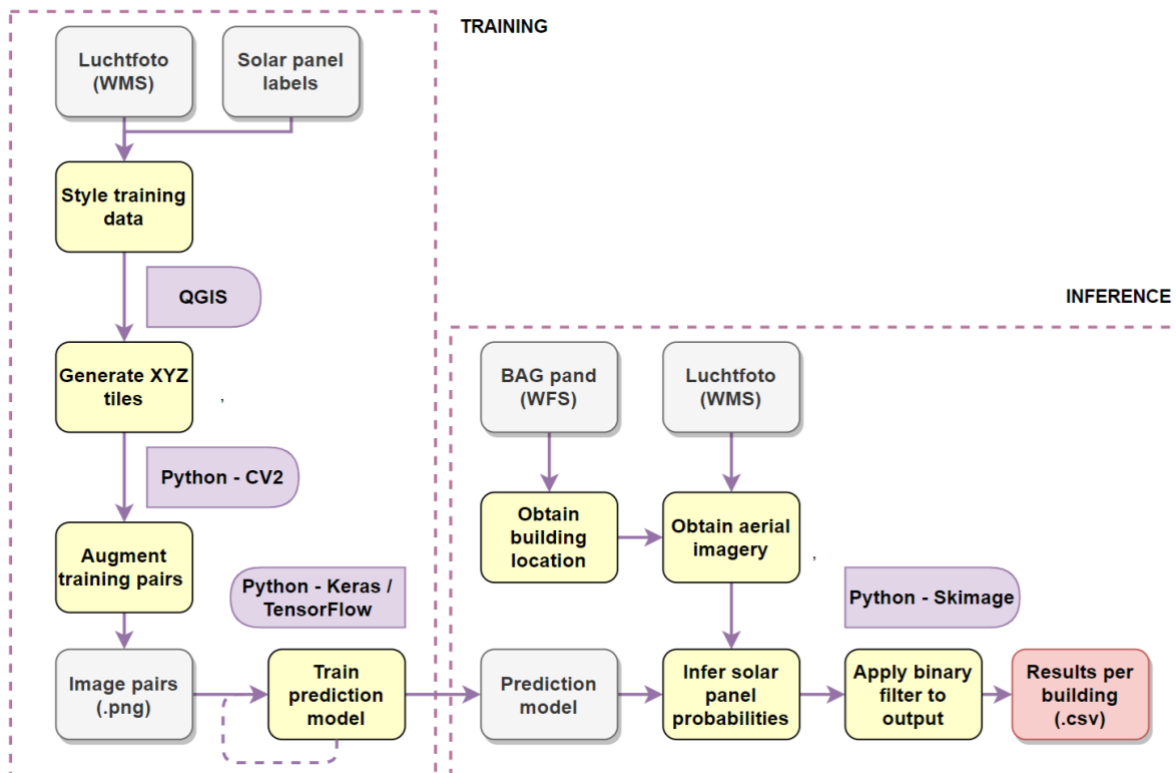


Figure 22. Pipeline of the supervised image classification

3.5.2 Input Data and pre-processing

3.5.2.1 Solar panel inference

For our primary goal of training a model capable of inferring solar panel placement from aerial photography, PDOK Luchtfoto dataset with nation-wide coverage and resolution of 25 cm per pixel is an obvious choice for the input data. Although multiple datasets concerning solar panels are available in Netherlands, they only include aggregate statistics (*Polder PV, 2021*). We failed to find any open datasets with individual panels labelled. Thus, we resorted to labelling the solar panels manually, focusing on an area of Delfgauw (Figure 23). Over 600 polygons are labelled, demarking all the visible solar panels in the area. Each polygon almost always contains a larger cluster of panels, meaning thousands of individual panels are labelled in total. Additionally, some panels outside the area are marked to be used as validation data.



Figure 23. Area of 1,5km² in Delfgauw, in which solar panels were labelled.

Additionally, to test whether higher resolution aerial imagery influences the results and whether the resolution of the aerial imagery distributed through PDOK would be sufficient, we additionally use a higher resolution aerial imagery for the city of Amsterdam (Figure 24). In there, 488 polygons are marked, containing roof mounted solar panels in the area around the station Amsterdam Amstel. Although our primary focus is on datasets with nation-wide coverage, we wanted to test what is theoretically possible with better input data.



Figure 24. National PDOK (left) and map.data.amsterdam.nl (right) aerial photography.

The aerial imagery layer and solar panel polygons are collected and styled in QGIS, from where they are exported as a collection of XYZ tiles. For training the final solar detection model, the tiles are exported at double the resolution (512x512) from which multiple crops and other transformation are generated. This step - called data augmentation, allows us to generate more unique training data for the network while working with a limited dataset.



Figure 25. Examples of training image pairs for solar panel detection model.



Figure 26. Example of new images generated through image augmentation.

3.5.2.2 Additional semantics

For further experiments, we are interested in other roof related open data sets in the Netherlands. Besides the aerial imagery in visible light and infrared spectrum and the AHN3 point cloud, respectively raster DSM, two other interesting data sets are found.

Amsterdam’s Solar panels and heritage – roof inventory (*Zonnepanelen en erfgoed - dakeninventarisatie*) is a data set specifying feasibility of solar panel retrofitting for majority of the buildings in a wider centrum of Amsterdam. Feasibility is assessed based on the age and heritage protection status of the building.

Zonatlas (*Zonatlas*, 2021) is a multi-layered dataset, focusing on feasibility of solar production from a roof geometry perspective. The solar potential is computed from the DSM model derived from AHN3 and combined with estimate of available roof area and building size.

Both data sets are of interest for testing, whether these additional semantics can be inferred from aerial imagery or DSM model by the use on neural networks. Similar to the solar panel training dataset, the training images are generated as XYZ tile sets created from different layers. Using XYZ tiles makes it possible to generate repeatable and consistent tiling for each dataset (see Figure 27), allowing us to experiment by mix and matching input and output datasets.

In total we generated 1800 (100 pre-augmentation) image pairs with a resolution of 256x256 at zoom level 19 for the solar detection model training and 3016 image tiles (each tile in various variants) at a 256x256 resolution at zoom level 18 for the roof semantics inference experiments.

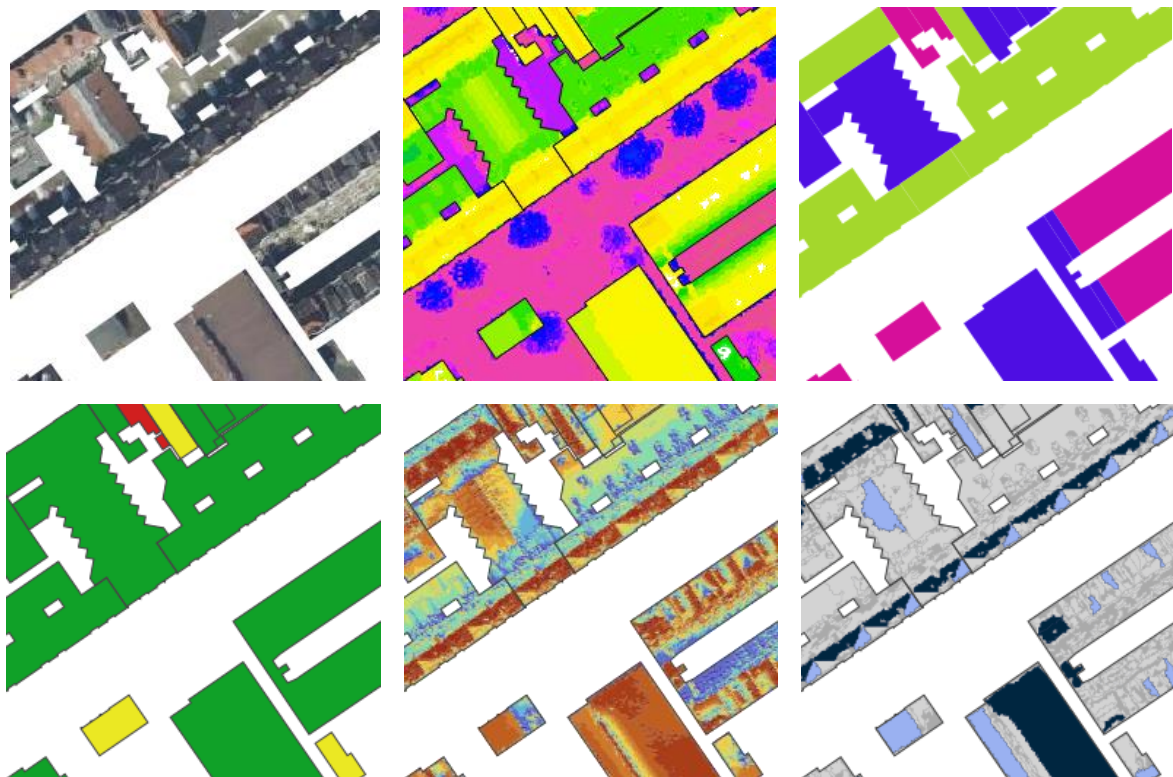


Figure 27. Extracted training tile examples (top row left to right: PDOK Luchtfoto, AHN3 DSM, Zonnepanelen en erfgoed - dakeninventarisatie, bottom row left to right: Zonatlas - opwekking, Zonatlas - instraling, Zonatlas - Geschied)

3.5.3 Implementation

3.5.3.1 Solar panel detection

We use simple CNN with only 3 convolutional layers for the solar panel detection model. We even omit deconvolutional layers, which means the networks output comes does not keep the resolution of the input and is cropped instead, which we have to compensate for in postprocessing. On the other hand, the simplicity of the network reduces the number of parameters and speeds up the training times significantly (see Figure 31 for the model graph).

The whole solar panel detection model is implemented in Keras – a deep learning API for Python build on top of very popular Tensorflow machine learning library. Tensorflow allows for training the network on a GPU (with CUDA drivers) which speeds up the training process dramatically and allows for quick iteration times. Training on ca. 1200 images for 200 epochs takes around 1 hour on a consumer level laptop equipped with a GTX 1050Ti graphics cad.

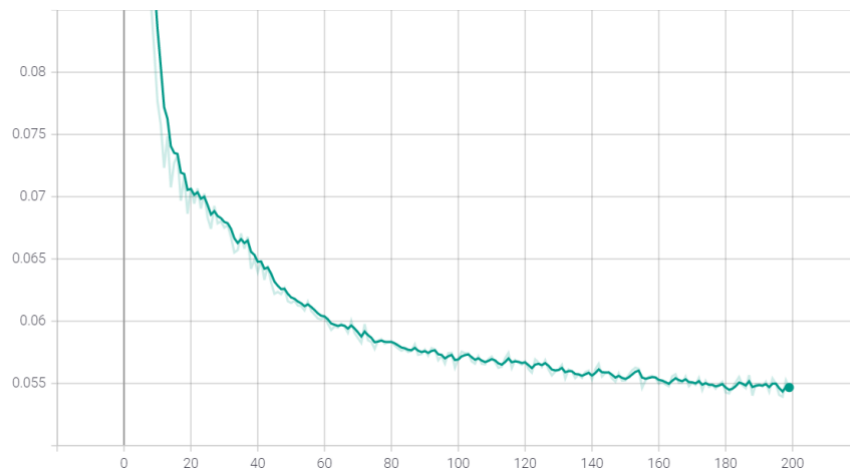


Figure 28. Graph of binary cross-entropy loss during the solar panel model training epochs.

The output from the network is a per pixel map of probabilities ranging from 0 to 1 for a solar panel being located at a given pixel. Since the probability range is continuous while we prefer a fully binary result, we implement a chain of gaussian blur filters and thresholding operations to identify the blobs / areas of high solar panel probability and filter out the noise (see Figure 29).

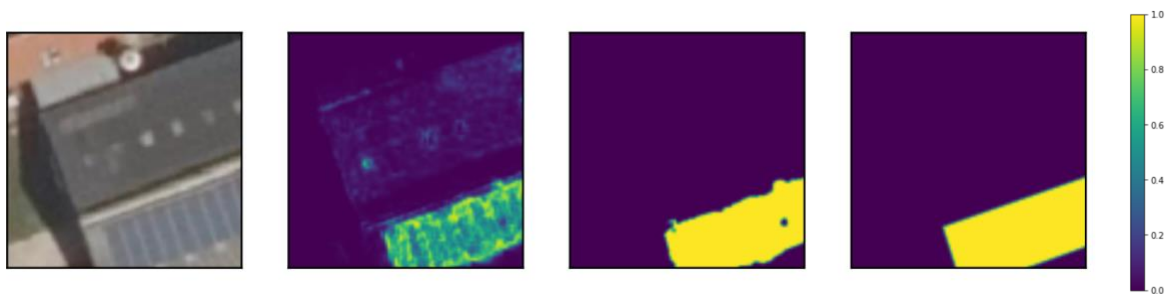


Figure 29. Solar panel detection (left to right: input image, prediction, filtered result, ground truth).

To conclude, we implement a simple API, which is based on the provided building identification number fetched aerial imagery over WMS from PDOK, inferred solar panel probability from it and based on the results restrained to the buildings footprint obtained from BAG returned and information on whether a building is equipped with solar panel system (Figure 31).

3.5.3.2 Other semantics

For inferring the other semantics, we use a picture-to-picture translation using a GAN model (both input and output are RGB images, not binary labels). The specific implementation is the *pytorch-CycleGAN-and-pix2pix* (Isola *et al.*, 2018), very popular GAN project implemented in PyTorch. Since *pytorch-CycleGAN-and-pix2pix* is a fairly complex project including multiple network models, we stuck with the default parameters, using the *UNET_256* generator with the 256x256 input and output image pairs.

For these 256x256 image pairs, the generator model consisted of 54,414,000 and the discriminator model of 2.769,000 parameters, making the model much more complex than the simple solar panel detection CNN from previous exercise. As a result, the training time for an image set of around 2500 image pairs and 200 epochs is almost a full day, limiting the number of iterations we could try within the projects timeframe.

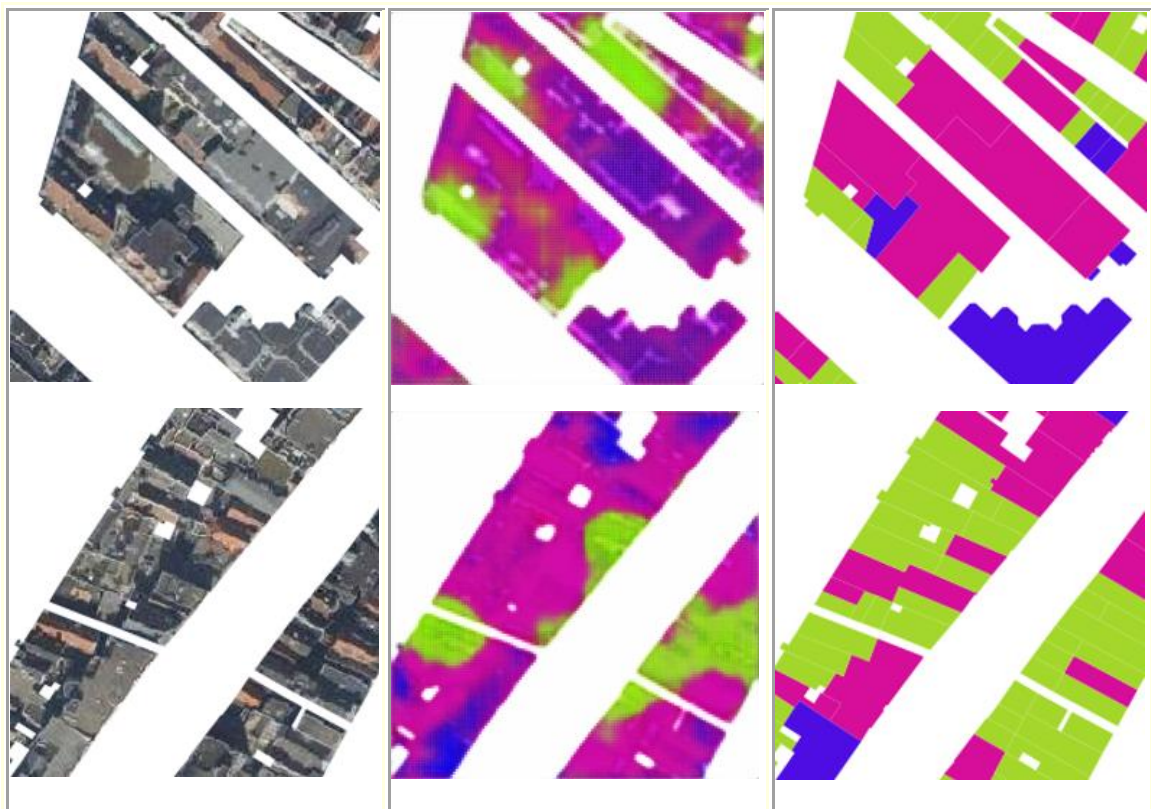


Figure 30. Results from the early stages of training on Luchtfoto - Zonnepanelen en erfgoed, dakeninventarisatie image pairs (left to right: input image, prediction, ground truth).

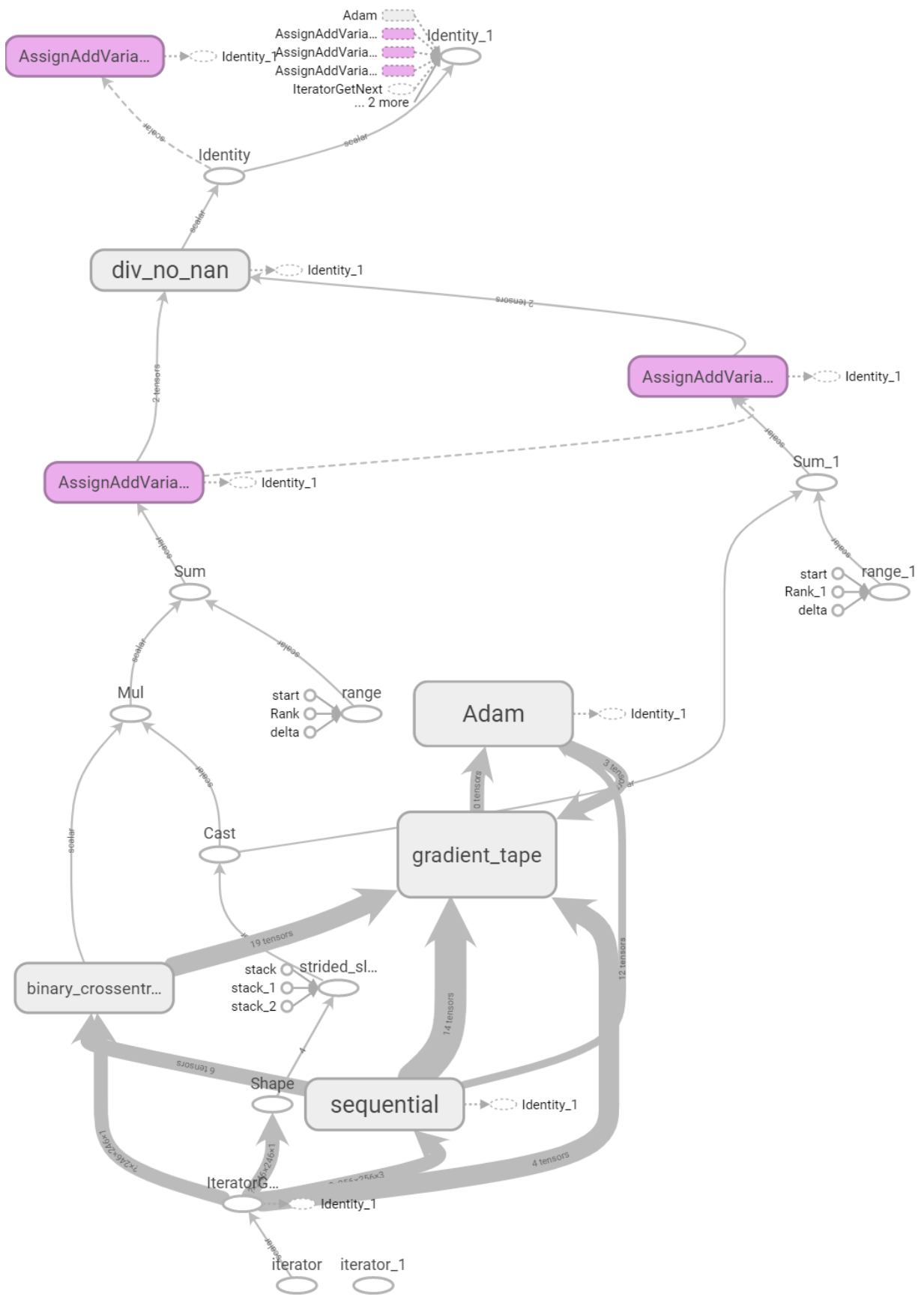


Figure 31. Complete graph of the solar panel detection model (generated with TensorBoard).

3.6 Merging the detection methods

Both geometry-based and image-based classification methods have advantages and disadvantages. For instance, geometry-based classification is efficient to detect chimneys and dormers that stand distinctly over the roofs. Nevertheless, window features over the roofs are not detected due to their small distance difference with the roof planes. On the other hand, image roof classification can detect windows and solar panels with different heights over the roof. In contrast, it detects every different colour of the main roof as obstacle. Thus, if the property has more than one roof material, one of these can be marked as an obstacle. Similarly, the dark shadows impact the detected obstacles.

Therefore, we merge the two approaches to improve the final result and minimize the false-positive errors. For this, the polygons resulting from the geometry-based part are transformed into a raster and masked similarly as done in the image classification part. In this way, both parts' resulting polygons, once rasterised, can be compared using a pixel-based approach (Figure 32, top two images). Once overlaid, if both results detect a pixel as an obstacle, its probability to be an obstacle is settled to 100%. If either of the results detect a pixel as an obstacle, its probability to be an obstacle is settled to 50%. Finally, if none of the approaches detect the pixel as an obstacle, its probability is settled to 0% (Figure 32, bottom right).

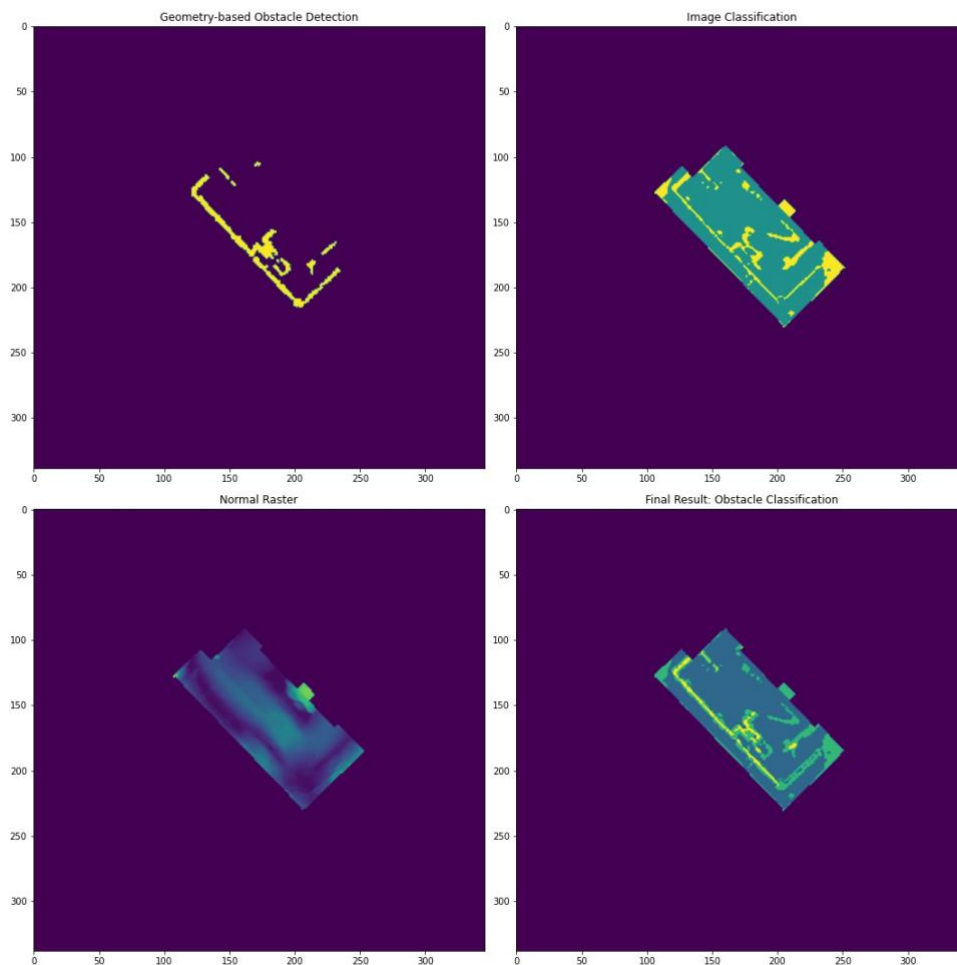


Figure 32. Merging of the different methods' results.

Furthermore, at this point it is possible to compute the 2D area of the obstacle. To retrieve the corresponding area in 3D, the point cloud dataset is used: the normal value of each point is projected in 2D and rasterised to obtain a normal value per pixel. This value evaluates the tilt of the roof per pixel and consequently allows us to compute the corresponding area in 3D. The implementation involves transforming the points into a grid and use an interpolation method to obtain a value for each pixel (Arroyo Ohori *et al.*, 2020). The interpolation is necessary because of the points' heterogeneous density of the AHN3 point cloud - mostly caused by occlusion effects and reflective surfaces (*idem*). The chosen method should cover the whole buildings' extent without losing accuracy on the results. After experimentations with different parameters and interpolation models, an Inverse Distance Weighting (IDW) method, based on the five closest points, is used (Figure 32, bottom left). Having both raster with the normal values and the final overlaid polygons', the area in 3D is computed and stored in a *csv* file.

Finally, the results of the supervised image classification (method III) are also added. A boolean value per building ID is added in the *csv* file as a third column, indicating whether the building has a solar panel or not (Table 2).

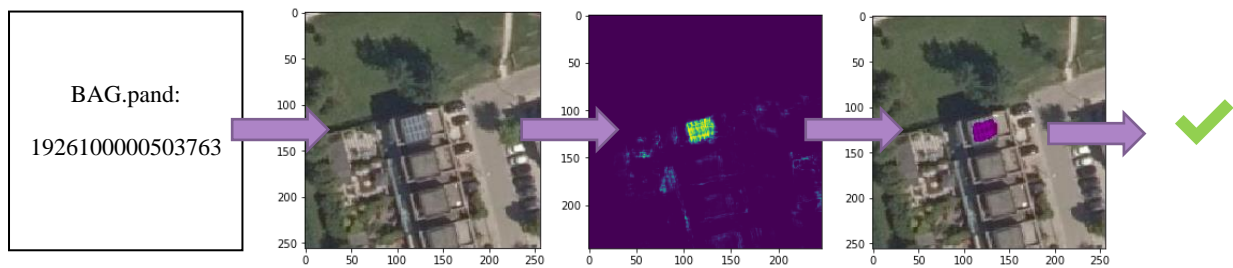


Figure 33. Solar panel test pipeline (left to right: input building ID, aerial image, inference, filtered result, test result).

As a result of the three merged methods, a final *CityJSON* model is produced. It is a semantically enhanced version of the 3D-BAG input model, adding three new attributes per building, read from the *csv* file: the “available roof area”, the “obstacle roof area” and the Boolean value “has a solar panel” (Figure 34).

Identificatie	Obstacle area	Has_solar_panel
0772100000297414	12.80	False
0772100000293562	7.75	False
0772100000293563	7.48	False
0772100000293564	14.38	False
0772100000293565	14.20	True
...

Table 2. Structure of the *csv* file resulting of the merging of the three methods.

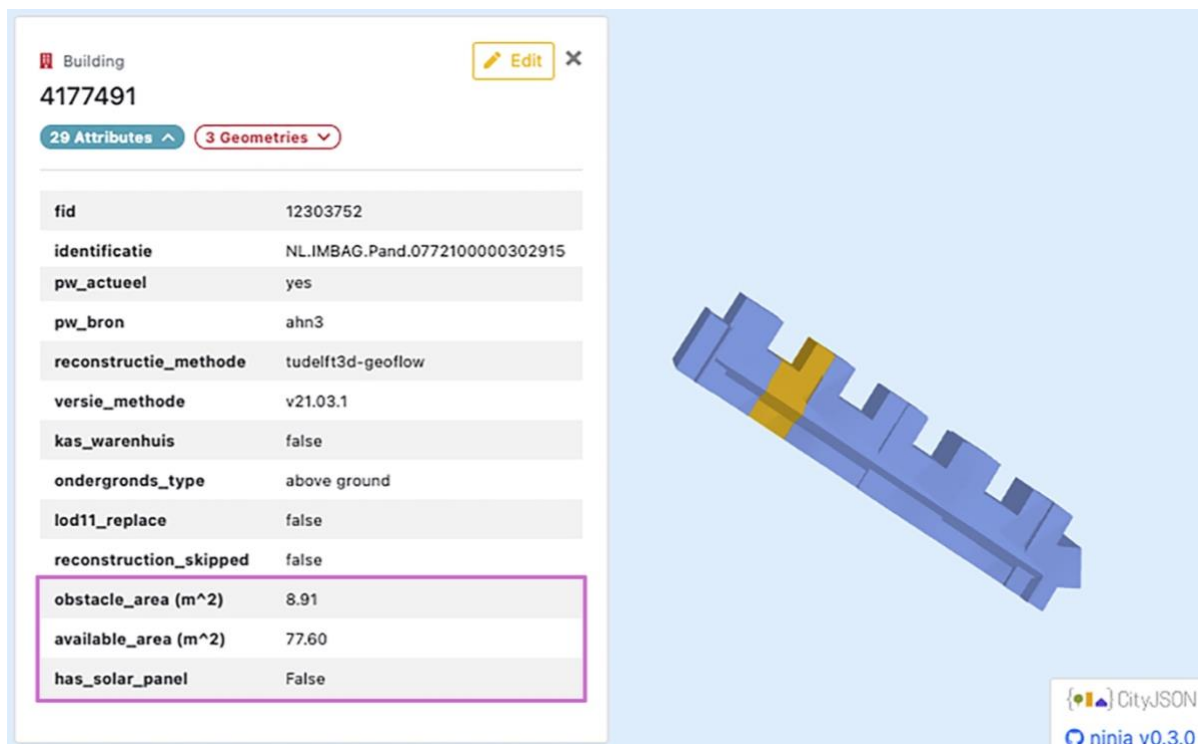


Figure 34. Final CityJSON, semantically enhanced with three new attributes.

4 Discussion of the Results

This section elaborates on the final results obtained by comparing them with the ground truth to assess the accuracy. Accordingly, limitations of the algorithm are also discussed.

4.1 Obstacle detection

4.1.1 Validation of the result, comparison to ground truth

The results obtained for the 50 buildings dataset provided by Brink are assessed below. For this purpose, a ground truth of the obstacle polygons is first manually drawn on QGIS and exported as GeoJSON. The polygons are then rasterised similarly as the ones obtained by the algorithm previously developed, on a 25 cm-resolution raster. Then, a validation method is elaborated, based on a pixel comparison approach.

4.1.1.1 Validation method

In order to quantify the quality of our results, we use a *contingency matrix* for pixel classification as explained by Janssen (2004). A comparison between the ground truth and our classification is obtained by building an *error matrix* based on the comparison of their respective pixel values. This is done twice with two sets of results applied to the 50 buildings dataset: once, with a manual alignment of the footprints to the aerial images and once with the automatic alignment method we developed. Then, the *error matrix* allows us to compute some objective accuracy parameters: on the one hand, the so-called “producer accuracy” and “user

accuracy”, and on the other hand an “overall accuracy” (Janssen, 2004). The producer accuracy refers to the probability that a pixel’s value (or class) in our classification results is similar to the one expected by the ground truth. On the other hand, the user accuracy is the probability that a certain pixel class of the ground truth - either main roof or obstacle - has also been labelled as that class in our classification result.

The parameters are computed for each building before being averaged for the whole dataset.

4.1.1.2 Application example of the validation method to one building

The main challenge is to adapt the method to our case, since the ground truth and our classification result have a different number of classes (two and three in our results). In the end, the pixels which obtained a probability value of 50% in the classification process are used both as main roof area and roof obstacle area but weighted by 0.5 in the error matrix.

The example below shows how the method is applied to one of the buildings. Both manual and automatic alignment results are shown for the building with id 0772100000296999.

In the example (Table 3), the raster contains 8509 pixels. From top to down are the results of the ground truth considered: in the bottom the “total” row indicates the total number of pixels of each value for the ground truth while the “producer accuracy” indicates to what extent our result matches these for each pixel value. So, in the case of perfect alignment (Table 3) 86% of the pixels expected to be non-obstacle class are correctly found by our classification, while 19% of the pixel expected to be obstacle are correctly retrieved by our classification.

Then, from left to right, are the classification results considered. The “user accuracy” indicates in this example that 88% of the pixel classed as main roof area are matching the ones of the ground truth, while only 28% of the pixels labelled as obstacle class were expected to be so considering the ground truth.

		Ground truth			
		Main part (0)	Obstacle area (1)	Total	User acc. (%)
Result	Main part (0)	6305	889	7194	88
	Obstacle area (1)	952	364	1316	28
Total		7257	1252	8509	
Prod. accuracy (%)		86	19		78

Table 3. Error matrix for an example building, using a perfect alignment.

		Ground truth			
		Main part (0)	Obstacle area (1)	Total	User acc. (%)
Result	Main part (0)	6330	889	7219	88
	Obstacle area (1)	977	364	1340	27
Total		7307	1252	8559	
Prod. accuracy (%)		87	29		76

Table 4. Error matrices for an example building, using the automatic alignment method.

The overall accuracy corresponds to the number of pixels of each class which were found similarly by the ground truth and our classification result. Therefore, the diagonal values are summed up and divided by the total number of pixels. 78% of overall accuracy is obtained in the previous example.



Figure 35. Comparison between the obstacle detection method with manual alignment and the ground truth with the image of the respective building.



Figure 36. Comparison between the obstacle detection method with automatic alignment and the ground truth with the image of the respective building.

4.1.1.3 Overall result, general statistics:

The first results are made with the fully automated method of combining the geometry-based obstacle detection and the image classification with the BAG footprint alignment correction.

Afterwards, we ran the method given the parameters of the alignment correction for each building.

Method	User Accuracy Main Roof	User Accuracy Obstacles	Producer Accuracy Main Roof	Producer Accuracy Obstacles	Overall Accuracy
Fully Automated	87%	19%	85%	21%	76%
Alignment given	89%	31%	87%	31%	78%

Table 5. Overall Results Accuracy.

The *user accuracy* for the main roof is the total of pixels grouped as main roof in the result that are main roof pixels on the ground truth, over the total of pixels detected as main roof in the project. On the other hand, the *user accuracy* of the obstacles is the quantity of pixels that was detected as an obstacle and was indeed an obstacle in the ground truth, over the total of pixels labelled as an obstacle by our method. This value depends on the ability of the method to detect the obstacles.

Moreover, the *producer accuracy* of the main roof is the number of pixels detected as main roof, over the total number of pixels of main roof as expected in the ground truth. As the *user accuracy*, it has higher values for the main roof parts than the obstacles. In contrast, the *producer accuracy* of the obstacles is the total number of pixel obstacles detected, over the total number of obstacles in the ground truth. The Figure 37 and 38 show the overall results, using the two alignment methods.

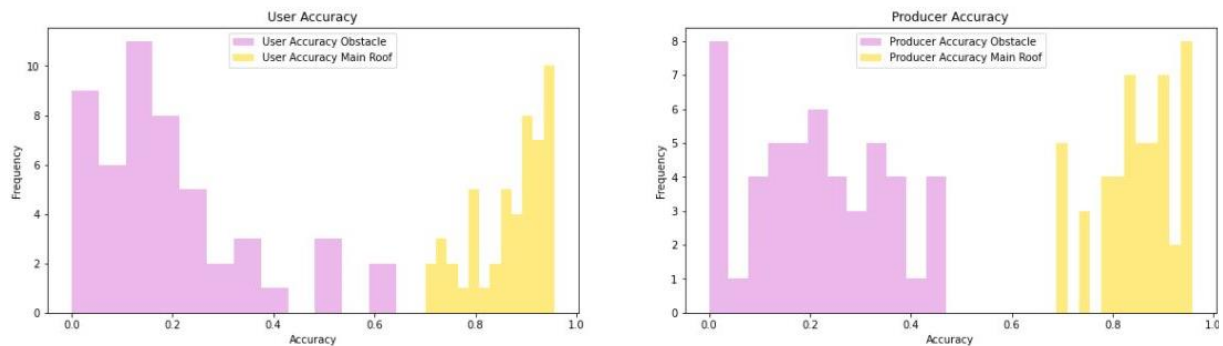


Figure 37. Fully automated Histogram Accuracy.

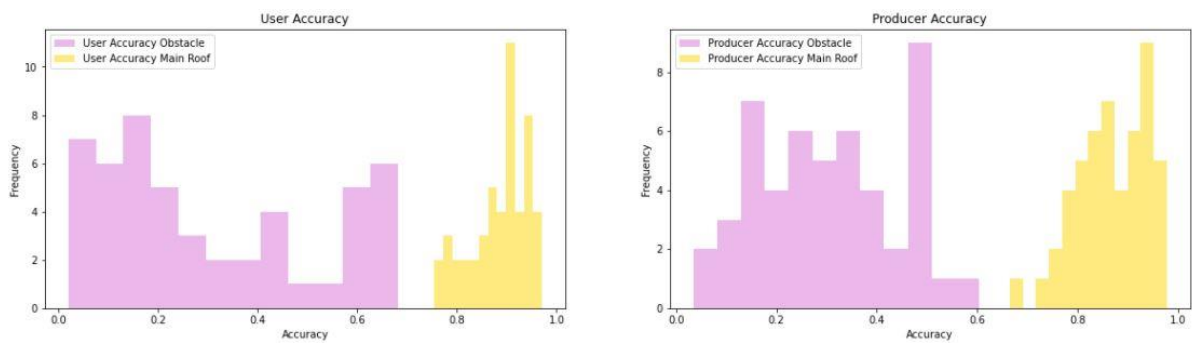
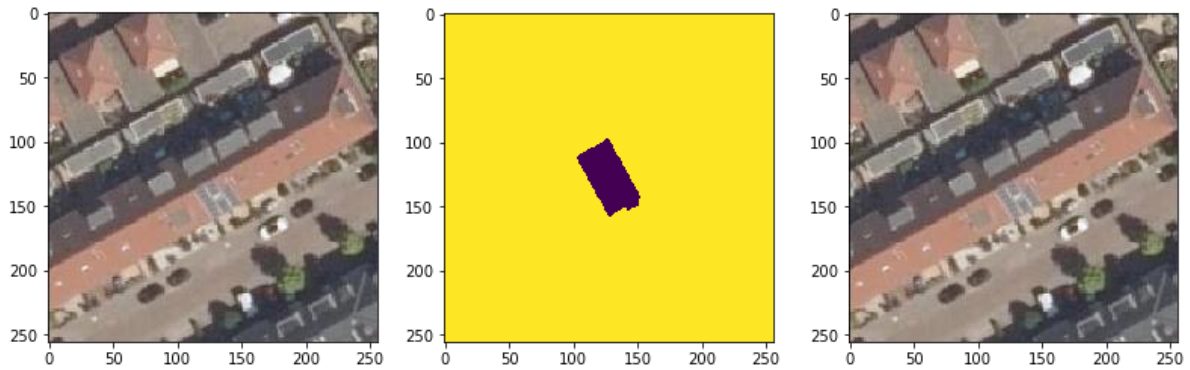


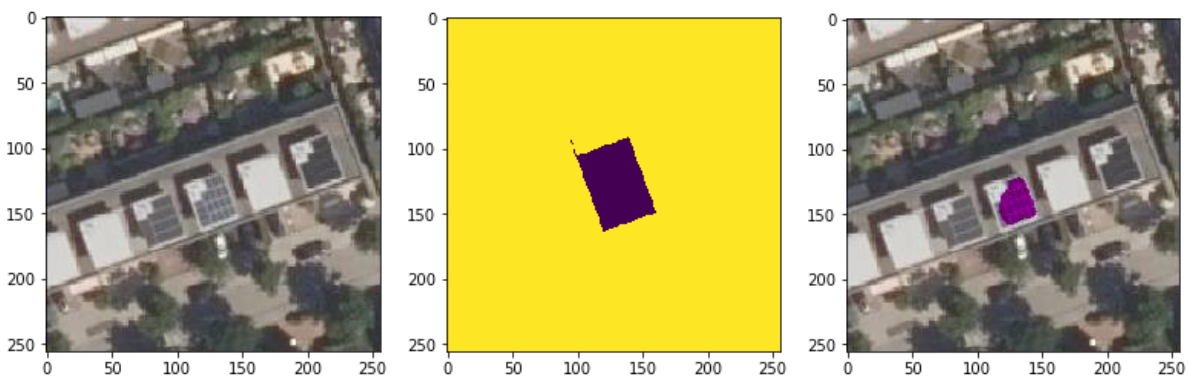
Figure 38. Alignment given Histogram Accuracy.

4.2 Solar panel inference

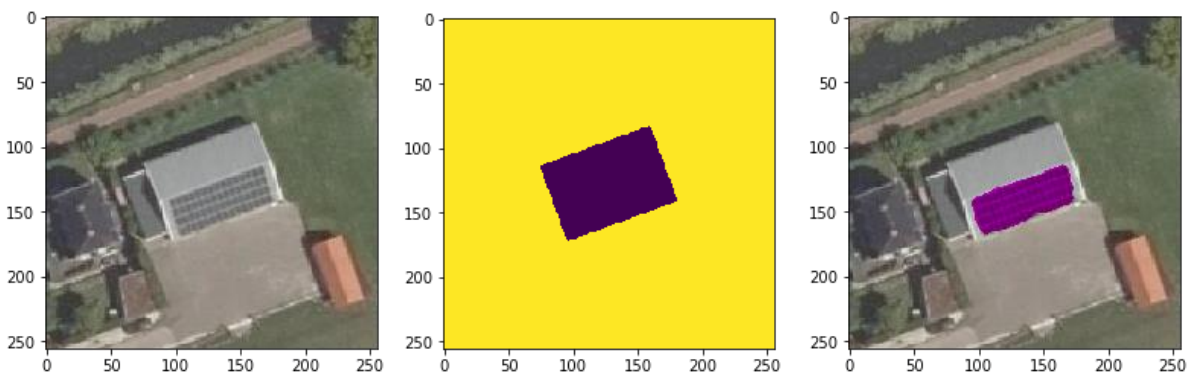
For the considerably small amount of data, the solar panel detection model provides fairly good results for various buildings in our validation dataset that we subsampled from the training data acquired in Delft (see Figure 39). However, the accuracy drops much lower for the validation dataset in Eindhoven, provided by Brink, containing new building typologies that are missing in the training data.



Simple sloped roof building, urban setting (BAG.Pand:1842100000002484)

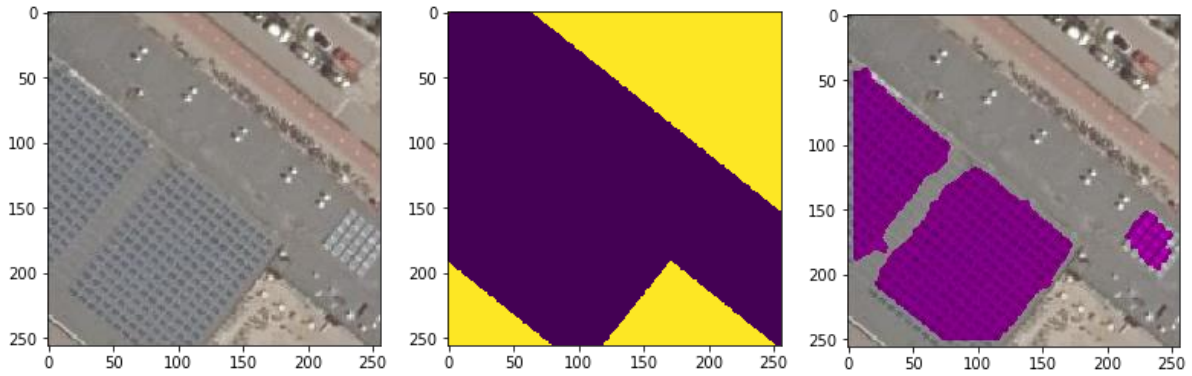


Simple flat roof building, urban setting (BAG.Pand:1926100000497049)

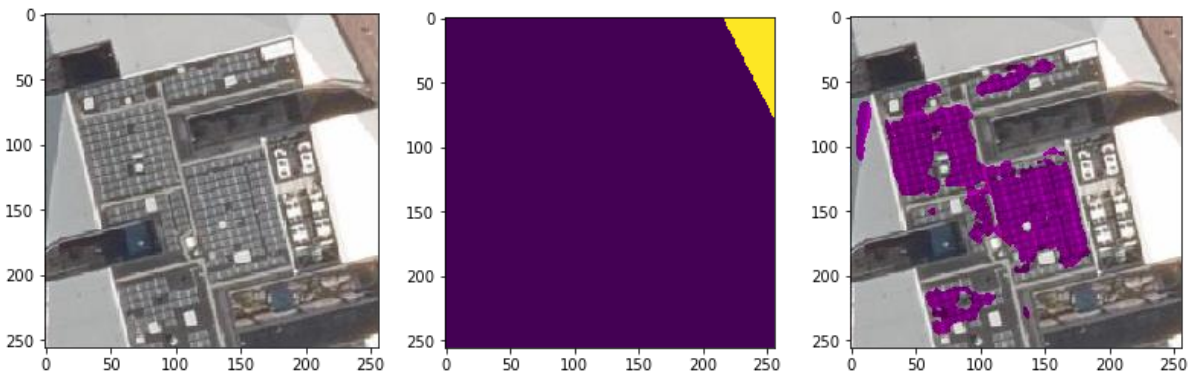


Simple sloped roof building, rural setting (BAG.Pand:1926100000484391)

Figure 39. Selected solar panel inference results (part 1/2) (left to right: input image, area of interest, detected panels).



Industrial roof building (BAG.Pand:1926100000487840)



Complex roof building, Delft station (BAG.Pand:0772100001000734)

Figure 40. Selected solar panel inference results (part 2/2) (left to right: input image, area of interest, detected panels).

For the 50-building test dataset provided by Brink a 70% prediction accuracy (25% accuracy for the positive cases, 84.2% accuracy for the negative cases) is achieved. The detailed results for each of the 50-building test dataset are in the appendix (Appendix II).

Cases	Accuracy	Correct	Incorrect
<i>All</i>	70.0%	35	15
<i>Positive</i>	25.0%	3	9
<i>Negative</i>	84.2%	32	6

Table 6. Accuracy of the solar panel detection for the 50-building dataset provided by Brink.

Observed cases where the model is prone to errors:

- Detecting windows on walls which are visible in oblique angles (thanks to not using perfect orthophotos) as solar panels.
- Sloped roofs in general (training dataset has a majority of flat roofs)
- Solar collectors (water heating) are much less likely to be detected than PV panels (in labelling process there is no distinguishing between them).

4.3 Additional semantics

From the results generated by using the pix2pix GAN mode, the results for inferring solar potential from AHN3 DSM model as input seem to be most promising. The predicted results even from our small training dataset are approaching the ground truth. Other training data combinations we experimented with (e.g., Luchtfoto - *Zonnepanelen en erfgoed*, AHN DSM – *Zonatlas, Geslacht*) do not yield to satisfying results.

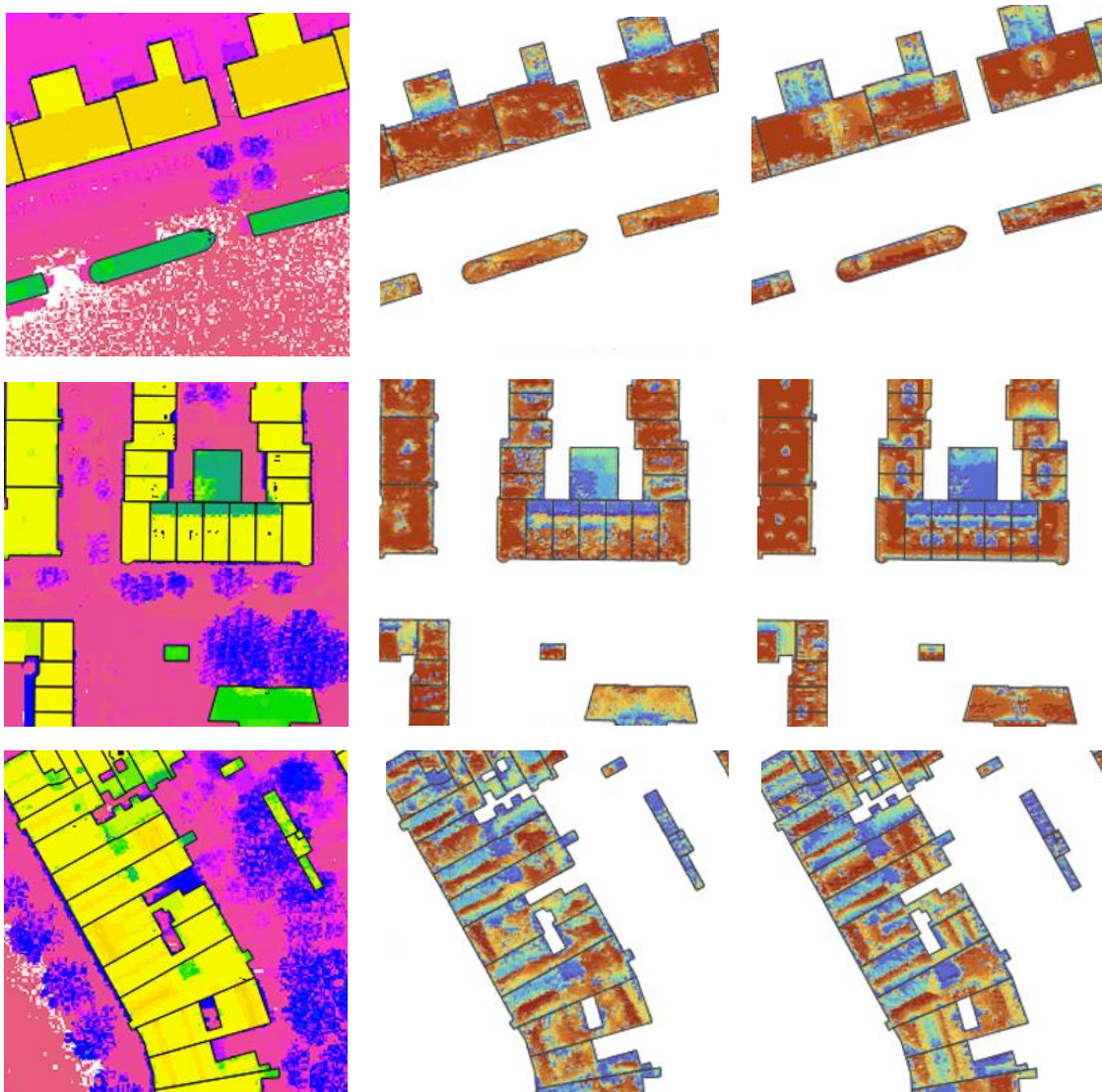


Figure 41. Selected solar potential inference results (left to right: input, prediction, ground truth).

The results often differ from ground truth in areas where true height information is obscured thanks to the raster DSM distributed in PDOK being split into discrete bands with 1-2 m height steps. With better DSM raster, the results would probably get even better.

4.3.1 Assessment and discussion of the results

4.3.1.1 Impact 1 - Rasterization, accuracy loss

The statistics built upon the results show that the obstacle-detection overall accuracy oscillates between 2% to 60%, whereas the main-roof area detection accuracy presents much better results: between 70% and 90%. This can be explained by the fact that the main roof area is much easier to detect in percentage since it concerns a much higher number of pixels. On the other hand, the obstacles represent a small amount of pixel, and the multiple rasterization process loses accuracy on their location. Consequently, it is unlikely that the pixels' location exactly matches between the ground truth and our classification.

4.3.1.2 Impact 2 - BAG and aerial image alignment

Also, we can observe from Figure 37 and Figure 38 that the results obtained when using a perfect alignment of the BAG footprint with the aerial image are significantly improved regarding the obstacle detection accuracy. The impact of the alignment between the two datasets represents thus a key factor for results' quality.

4.3.1.3 Impact 3 - Sample data

Moreover, the obstacle-detection accuracy is highly impacted by the different types of roof structures and materials. It is thus highly dependent on the sample data used. We could observe a high variability of buildings' type in our datasets: housing in rows such as bigger isolated buildings, with various roof materials/colours.

4.3.1.4 Impact 4 – Validation method used

Finally, the validation method we chose impact the quality assessment. Indeed, we take into account the results' quality by considering the pixels one by one, in a very accurate way. Another parameter, without entering in such details, would be to compare the overall obstacle surface detected in both cases (ground truth and classification results). However, the obstacle alignment would not be guaranteed between the ground truth and our result and thus not reveal much.

4.3.1.5 Impact 5 – Generalizability of input data

The solar panel detection model was trained only on a small neighbourhood in Delft and struggled to generalize to areas outside of Delft. It generated good results for validation dataset obtained from area close by to the training dataset, even when the validation dataset was obtained at different zoom level. However, once the location changed, either the building typology or possibly the slightly different light conditions negatively influence the accuracy.

Solution would be to collect more data for more diverse cases and potentially improve the architecture of the model itself, since the model is currently overly simplified. Modifying the networks meta-parameters can be daunting, especially without much prior experience and could easily result in increased training times without accuracy gains but is probably necessary.

4.3.2 Limitations

The combined results are primarily dependent on the three methods used and their respective limitations. Therefore, we analysed them as follows:

4.3.2.1 Geometry-based classification

The geometry-based approach uses the 3D BAG as main input dataset, but this does not represent every building *consistently* due to the parameters used in the automatic 3D model generation. Among others, an important parameter is the number of points supporting a generated plane, therefore it is impacted by the point cloud density. Also, the 3D model aims at keeping a low complexity. As a result, for instance, some dormers are represented in the 3D BAG model whereas others are missing. First of all, the definition of an obstacle is thus set into question: should a dormer be considered as an obstacle or not? Secondly, depending on the input building the obstacle points retrieved can be of different nature, for instance belonging to a dormer or not. As a solution, we agreed on taking as basis the 3D model input as it is, and considering each surface represented there as non-obstacle - since it is probably big enough to be used for solar panel installation - and each prominent element missing as an obstacle. However, the other parts of the implementation, based on images, use another definition of “obstacle”.

To conclude with, this obstacle-detection method remains very dependent on the input 3D model and the way the latter was generated.

A second limitation of this method is its dependency to the other input dataset’s accuracy: the buildings’ point cloud. The latter has still a limited accuracy for this application purpose. We noticed that by setting the distance parameter down to 30 cm instead of 40 cm, a large number of “incorrect” points were selected. Therefore, our result’s quality is limited by the point cloud’s resolution. Additionally, the acquisition technique of AHN3 dataset based on aerial scanning, results in parallel lines of points which impacted a lot the shapes of the obstacle polygons finally retrieved. Both the parallel repartition of the AHN3 points and the distance between them are greatly affecting the quality of our results.

The third identified limitation is the method used to group the points together to form distinct obstacle polygons. Offsetting the points and merging them offered the best results in comparison to clustering methods. However, this limits the ability to recover meaningful polygons shapes. When observed on aerial images obstacles usually have a rectangular shape (dormer) or square one (chimneys), which is never the case in our implementation. On the top of it, the rasterization performed for merging the different methods alter the shapes even more.

4.3.2.2 Unsupervised image classification

The method II often detects the shadows of an obstacle rather than the obstacle itself. While this still indicates that there is something, it leads to non-overlapping results of method I and II. Additionally, the resolution of 25 cm of the open aerial image is not enough to detect small obstacles and to perform a better automatic alignment. Another limitation is that because the image is tilted, shadows are encountered in some buildings. Hence, the algorithm detects it as

an obstacle as it performs the clusterisation looking at the coloured value of the pixel. In addition, the presence of shadows in the image can bias the clusterisation as it changes the colour of the pixel if different parts of it have different orientations, marking one part as an obstacle and the other as the main roof.

Furthermore, a limitation is present on labelling the classes correctly into the main roof class and the obstacle class since the method only groups the pixels without assigning them to the according class. As a result, for buildings with more of the roof area covered by obstacles, it will detect it as the main roof.

4.3.2.3 Supervised image classification

The limitation is mostly in the amount and diversity of training data that we could generate. The trained models are too context sensitive (e.g., model trained in small neighbourhood of Delft not being able to generalize to typologies from different locations), or on the other hand training sets are so broad, that even with larger number of images, the model struggles with delivering reliable results (this happens with the model trained for Amsterdam Amstel station area, a very diverse area with a mix of office, residential, industrial building and gardener ‘colonies’).

Perhaps a more robust network architecture would elevate the issues. Fairly shallow network with just three convolutional layers is more likely to rely on superficial properties such as absolute pixel colour instead of more complex geometrical features and relative relations between them for classification. For an example see Figure 42 where network classifies pale blue roof as a solar panel instead of the actual black coloured solar panel to the right.

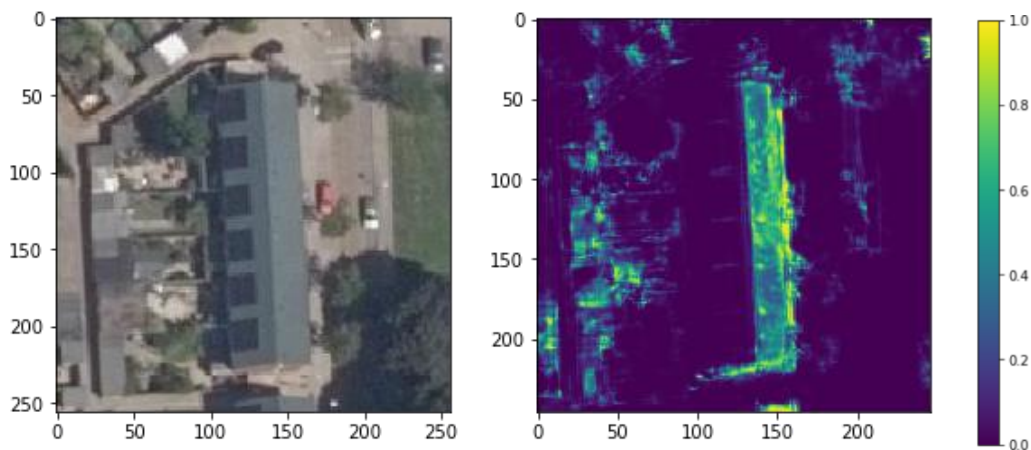


Figure 42. Solar panel detection model confused by the blue coloured roof (left: input, right: inference).

5 Reflection

This section evaluates the success of the project plan by comparing the end result with the MoSCoW requirements while emphasizing the encountered problems throughout the project as well as the lessons learnt. Finally, a connection between the project and the Geomatics courses is provided.

5.1 MoSCoW requirements

At the beginning of the project, the scope and objectives were determined with the MoSCoW method by identifying the MUST haves, SHOULD haves, COULD haves, and WON'T haves of the project. After having meetings and discussions with the company and the mentors, the final scope of the project was redetermined and the objectives were updated accordingly (Table 7).





MUST 	SHOULD 
<p><u>End-product</u></p> <ul style="list-style-type: none"> - Detect roof obstacles - Calculate the total area of obstacles on roofs - Add semantics to the LoD2 building models - Stay in close contact with the company so the end-product meets their needs <p><u>Method</u></p> <ul style="list-style-type: none"> - Base the method on open data so that it is applicable for the whole Netherlands (or more) 	<p><u>End-product</u></p> <ul style="list-style-type: none"> - Use training data to detect solar panels with deep learning - Consider the works previously elaborated within the company for automatic detection of building elements (openings...) and use similar concepts <p><u>Algorithm, implementation</u></p> <ul style="list-style-type: none"> - Use/get inspired from already existing algorithms and methods and develop our own pipeline upon them
COULD 	WON'T 
<p><u>End-product</u></p> <ul style="list-style-type: none"> - Detect the most common roof materials - Calculate insolation and solar potential - Detect dormers on the roofs <p><u>Algorithm, implementation</u></p> <ul style="list-style-type: none"> - Optimize the code to make it faster 	<ul style="list-style-type: none"> - Detect geometry of roof tops - Focus on other building components such as façades - Develop our own deep learning architecture, but base our work on existing ones

Table 7. MoSCoW Method of Requirements and Objectives.

The end-product of the project shows that all the MUST haves are realized as well as the SHOULD haves, since we were able to detect the solar panels and integrate this information to the 3D BAG LoD2 models. Moreover, the input datasets used consisted only of open data. Already existing algorithms/ libraries were also extensively used and adapted to create our own pipeline. On the other hand, COULD have objectives such as detecting roof materials and

calculating insolation and solar potential were considered too extensive for the duration of the project; therefore, these objectives were mentioned as a topic for future work.

5.2 Problems encountered & lessons learnt

The first problem we have encountered was to define the scope and the focus of the project by ourselves, since Brink has given us the freedom to explore multiple different topics, they might be interested in. Even though it was interesting to search about previous works and implementations, this made it harder for us to decide on a topic and narrow it down. This experience has taught us to directly communicate with the company and the mentors to discuss our ideas and to get regular feedback from them so that the final scope of the project can be determined in a shorter time without stealing too much time from the project plan.

Secondly, we have learnt that we need to search a topic more extensively without choosing it as the main focus of the project. We have experienced this at the beginning when we had to shift the focus from roof material detection to roof obstacle detection after realizing that there was almost no open training data about roof materials to use for the deep learning approach. Therefore, it is important to already start thinking about the input data that might be needed in the project so that the main topic can be chosen more consciously.

For geometry-based classification, the first problem encountered was the criteria to determine obstacle points from the point cloud. For instance, the use of the points' normal helped us to improve the results by discarding points belonging to vertical surfaces. On the other hand, other criteria did not give appropriate results, and a lot of experimentations were performed (e.g., the normal rate of change was discarding too many points that should have been kept). Moreover, finding a good clustering method that would meet our requirements for the output was a challenge. After trying various ones, we replaced the clustering technique by another approach giving more satisfactory results. Thus, it is important to explore different possibilities and dare experimental processes, instead of focusing on one single option.

For unsupervised image classification, a problem encountered was the alignment between the buildings' footprint and the images since we had to mask the latter to perform a successful roof classification. This alignment can be ignored having a true orthophoto. Additionally, labelling the groups of the method is a challenging task, but it was discovered that for most of the tested buildings, the area of the main roof is bigger than the area of the obstacles.

For supervised image classification, the most important lesson to take away would be to always define validation and accuracy assessment metrics as soon in the process as possible. Making decisions on changing the model meta-parameters and choosing the best format of the input data can be hard when not informed by their direct influence on model's objective performance.

5.3 Connection to previous courses in Geomatics

Table 8 shows how most of the first-year Geomatics courses have a direct connection with the project since many concepts and algorithms have been covered in these courses, both providing with the theoretical background and enabling the students to have practical assignments.

Geomatics courses	Connection to the project
Python Programming for Geomatics (GEO1000)	Almost whole pipeline of the project was created in Python (using various libraries).
Geographical Information Systems (GIS) and Cartography (GEO1002)	QGIS was utilized to generate training data and obtain the ground truth by manually digitizing the roof features.
3D Modelling of the Built Environment (GEO1004)	Method I. depends on the concepts and encodings covered in this course; moreover, the construction algorithm of 3D BAG was introduced here as well.
Sensing Technologies and Mathematics for Geomatics (GEO1001)	Method II utilizes K-means clustering which was taught in this course and supported with an assignment to give an opportunity to experiment.
Geo-information Organisation and Legislation (GEO1009)	The concept of open data and its advantages were emphasized in this course.
Geoweb Technology (GEO1007)	OGC's Web Map Services (WMS) and Web Feature Services (WFS) were both heavily used in Method II and Method III

Table 8. Connection between the project and Geomatics courses.

6 Conclusions and Future Work

6.1 Conclusions

The main objective of this project is to infer semantic properties of roofs by specifically focusing on roof obstacles such as dormers, chimneys, and solar panels. Previous work has shown that there are various strategies/algorithms to detect obstacles on roofs. Inspired by these existing algorithms, we have developed three methods; The first method depends on a geometry-based classification in which the distance between the AHN3 point cloud and the 3D BAG LoD2 model determines the obstacle points, which are then offset and merged to form the obstacle geometry. The second method is based on an unsupervised image classification where the first step includes aligning the aerial image with the 3D BAG footprints while in the second step K-means clustering is used to create two clusters with obstacle and non-obstacle features. Finally, the third method is based on a supervised image classification with the aim of detecting solar panels on roofs, in which we first create the training data by manually labelling over a thousand individual solar panels and generate XYZ map tiles with the input data and output labels to create the training pairs. We use a simple CNN outputs a probability value of a solar panel located at a given pixel.

Since the accuracy of each method is not high enough individually, we combine the three methods by first merging the first two parts, then adding the information coming from the third one. To couple the first two methods, we rasterize the polygons resulting from the first method and overlay it with the output of the second, in order to use a pixel-based approach and decide if a pixel contains an obstacle or not. Moreover, the area of the obstacles in 3D is calculated with the help of each points' normal indicating the roof's tilt. Finally, this information is added to the 3D BAG LoD2 models in CityJSON format as an attribute while the information of having a solar panel or not is also added as an attribute as a Boolean value.

To validate and assess the results, we use an error matrix that compares the ground truth with our classification. The results show that the overall accuracy of obstacle detection changes between 2% and 60% while the detection of main-roof area presents much better results with an accuracy of 70% to 90%. This high difference in accuracy can be explained by the different number of pixels concerned by each class: since much less pixels are depicting obstacles, retrieving the same ones through different methods is less probable. Moreover, the overall result is impacted by a number of limitations coming from the geometry-based and unsupervised image classification parts of the algorithm, such as the quality of the 3D BAG dataset, the resolution of aerial images and the quantity of input data affecting the success of the results. As for semantics inferred by the supervised prediction model, we achieve 70% accuracy for the test dataset provided by Brink, although the results are better for other validation datasets typologically closer to the training data. Additionally, we achieved interesting results with predicting the solar radiation values at the roof surface using picture-to-picture generative adversarial networks.

6.2 Future Work

This project explores different methods/algorithms to reach high accuracy in the results; however, there are still improvements that could be worked on in the future to enhance each method. In parallel with each methods improvement, the overall computation time could be reduced by using another programming language such as C++ which is compiled-based, unlike Python. Merging the methods' results could also be thought in a different way and improved to avoid the rasterization process of the polygons retrieved with the geometry-based method, and to integrate more actively the unsupervised image classification results.

To improve the first method's results, the construction algorithm of the 3D BAG LoD2 model could be analysed, and the parameters used improved to get a more appropriate building model as input. This is desired because some dormers and solar panels are currently included as part of the roof geometry and are therefore not detected through the geometry-based classification method. Alternatively, a variant of the existing pipeline could be developed for the LoD2 model generation, instead of using the already existing 3D BAG model. This way, the parameters of the model could be determined from scratch by taking into consideration the input point cloud and the building geometries while keeping in mind the final purpose of the model. As an additional improvement, the input point cloud could be automatically pre-processed, since the points' normal are currently added through CloudCompare manually. Including their computation in the main algorithm would allow to process AHN3 dataset directly in our pipeline. A higher resolution AHN3 dataset would also improve the results. Secondly, the various criteria to detect obstacle points could be refined. Finally, more flexible clustering methods for the obstacle points could be implemented, and then the convex hull/ alpha shape could be used instead of offsetting points to generate the obstacle geometry. Even though several of these implementations were tested, more emphasis could be put on this, to try different alternatives and compare the results' quality.

For the unsupervised image classification of roof obstacles, the method can be further improved by using higher resolution images and true orthophotos to mask the images with the buildings' footprints. Additionally, the use of more advanced clusterisation algorithms mixed with intense values of a point cloud can improve the results and help to semantically label the groups. Another future work is to classify the obstacles detected to infer the type of obstacles such as solar panels, dormers, windows, or chimneys. In this way, it can be added as a standard object in 3D building models.

The solar potential prediction model shows some promise. Especially if combined with the obstacle model it could help us specify which obstacles actually have influence on the realistic solar potential of the roof. The inference times for a single tile were in order of 10-50ms, which is probably better or comparable with computing the solar potential over a raster by brute force ray-casting operation (depending on the sky model complexity). Training the model with higher resolution (non-binned) DSM raster could provide interesting results.

7 References

- 3D BAG (2021). Available at: <https://docs.3dbag.nl/en/> (Accessed: 14 June 2021).
- Aarsen, R. *et al.* (2015) *Decentral solar energy database*. Available at: <https://repository.tudelft.nl/islandora/object/uuid%3Ac9a9a9b5-b6bf-4385-b54c-9ab29fe3e7cb?collection=education>.
- AHN.NL (2021). Available at: <https://www.ahn.nl/geschiedenis> (Accessed: 14 June 2021).
- Al-doski, J., Mensorl, S. and Mohd Shafri, H. (2013) ‘Image Classification in Remote Sensing’, 3(10).
- Alexandridis, V. *et al.* (2020) *3D city modeling for automated verification of safety compliance rules*.
- Arroyo Ogori, K., Ledoux, H. and Peters, R. (2020) *Computational modelling of terrains*.
- Arroyo Ogori, K., Ledoux, H. and Peters, R. (2021a) ‘3D building reconstruction’.
- Arroyo Ogori, K., Ledoux, H. and Peters, R. (2021b) ‘Semantic 3D city models’.
- BAG viewer (2021). Available at: <https://bagviewer.kadaster.nl/>.
- Bergamasco, L. and Asinari, P. (2011) ‘Scalable methodology for the photovoltaic solar energy potential assessment based on available roof surface area: Further improvements by ortho-image analysis and application to Turin (Italy)’, *Solar Energy*, 85(11), pp. 2741–2756. doi: 10.1016/j.solener.2011.08.010.
- Brito, M. C. *et al.* (2019) ‘3D Solar Potential in the Urban Environment: A Case Study in Lisbon’, *Energies*, 12(18), p. 3457. doi: 10.3390/en12183457.
- Chronis, A. *et al.* (2020) ‘INFRARED: An Intelligent Framework for Resilient Design’, in: *25th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA)*, Association for Computer-Aided Architectural Design Research in Asia, pp. 1–10.
- Dutch New Energy Research (2021) *Dutch Solar Trend Report 2021*. Available at: <https://www.solarsolutions.nl/trendrapport/>.
- Dutch PV Portal* (2021). Available at: <https://pvportal-3.ewi.tudelft.nl/PVP3.1/Info/ModelExplanation.php#>.
- Freitas, S. *et al.* (2015) ‘Modelling solar potential in the urban environment: State-of-the-art review’, *Renewable and Sustainable Energy Reviews*, 41, pp. 915–931. doi: 10.1016/j.rser.2014.08.060.
- Gassar, A. A. A. and Cha, S. H. (2021) ‘Review of geographic information systems-based rooftop solar photovoltaic potential estimation approaches at urban scales’, *Applied Energy*, 291, p. 116817. doi: 10.1016/j.apenergy.2021.116817.

Google Maps (2021). Available at: <https://www.google.com/maps/@52.0066964,4.3545798,68m/data=!3m1!1e3>.

International Energy Agency (2020) *The Netherlands 2020 - Energy Policy Review*. Paris: IEA. Available at: <https://www.iea.org/reports/the-netherlands-2020>.

Isola, P. *et al.* (2018) ‘Image-to-Image Translation with Conditional Adversarial Networks’, *arXiv:1611.07004 [cs]*. Available at: <http://arxiv.org/abs/1611.07004> (Accessed: 18 June 2021).

Janssen, L. L. F. (2004) *Principles of remote sensing: an introductory textbook*. Enschede: ITC.

Kanters, J., Wall, M. and Kjellsson, E. (2014) ‘The Solar Map as a Knowledge Base for Solar Energy Use’, *Energy Procedia*, 48, pp. 1597–1606. doi: 10.1016/j.egypro.2014.02.180.

Kumar, A. and Chellappa, R. (2017) ‘A Convolution Tree with Deconvolution Branches: Exploiting Geometric Relationships for Single Shot Keypoint Detection’.

Ledoux, H. (2019) ‘cjio, or CityJSON/io’. Available at: <https://cjio.readthedocs.io/en/latest/>.

Li, Y. *et al.* (2020) ‘Implementation of deep-learning algorithm for obstacle detection and collision avoidance for robotic harvester’, *Computers and Electronics in Agriculture*, 174, p. 105499. doi: 10.1016/j.compag.2020.105499.

Machete, R. *et al.* (2018) ‘The use of 3D GIS to analyse the influence of urban context on buildings’ solar energy potential’, *Energy and Buildings*, 177, pp. 290–302. doi: 10.1016/j.enbuild.2018.07.064.

Overheid (2021). Available at: <https://data.overheid.nl/> (Accessed: 16 June 2021).

PDOK (2021). Available at: <https://www.pdok.nl/introductie/-/article/actueel-hoogtebestand-nederland-ahn3-> (Accessed: 14 June 2021).

Polder PV - detailed analyzes of the solar power market (no date). Available at: <http://www.polderpv.nl/> (Accessed: 18 June 2021).

Prakash, C. D., Akhbari, F. and Karam, L. J. (2019) ‘Robust obstacle detection for advanced driver assistance systems using distortions of inverse perspective mapping of a monocular camera’, *Robotics and Autonomous Systems*, 114, pp. 172–186. doi: 10.1016/j.robot.2018.12.004.

Romero Rodríguez, L. *et al.* (2017) ‘Assessment of the photovoltaic potential at urban level based on 3D city models: A case study and new methodological approach’, *Solar Energy*, 146, pp. 264–275. doi: 10.1016/j.solener.2017.02.043.

Satari, M. (2012) ‘Recognition of Dormers from lidar data using support vector machine’, in *2012 IEEE International Geoscience and Remote Sensing Symposium. IGARSS 2012 - 2012 IEEE International Geoscience and Remote Sensing Symposium*, Munich, Germany: IEEE, pp. 6005–6008. doi: 10.1109/IGARSS.2012.6352239.

Solargis (2021). Available at: <https://solargis.com/maps-and-gis-data/download/netherlands>.

Suri, M. *et al.* (2020) *Global Photovoltaic Power Potential by Country*. 149846. Washington D.C.: World Bank Group. Available at: <http://documents.worldbank.org/curated/en/466331592817725242/Global-Photovoltaic-Power-Potential-by-Country>.

Thorpe, D. (2018) *Solar energy pocket reference*. Second edition.. New York, NY : Routledge,.

de Vries, T. N. C. *et al.* (2020) ‘A quick-scan method to assess photovoltaic rooftop potential based on aerial imagery and LiDAR’, *Solar Energy*, 209, pp. 96–107. doi: 10.1016/j.solener.2020.07.035.

Yang, Y. *et al.* (2020) ‘Potential analysis of roof-mounted solar photovoltaics in Sweden’, *Applied Energy*, 279, p. 115786. doi: 10.1016/j.apenergy.2020.115786.

Yoo, H. *et al.* (2016) ‘Real-time rear obstacle detection using reliable disparity for driver assistance’, *Expert Systems with Applications*, 56, pp. 186–196. doi: 10.1016/j.eswa.2016.02.049.

Zonatlas (2021). Available at: <https://www.zonatlas.nl/start/>.

Zonnepanelen en erfgoed - dakeninventarisatie (no date). Available at: https://maps.amsterdam.nl/zonnedaken_erfgoed/?LANG=nl (Accessed: 18 June 2021).

8 Appendix I

8.1 Team members

Irène Apra



Email: ireneapra@gmail.com

Background: BSc and MSc in Architecture (École Nationale Supérieure d'Architecture de Paris-val-de Seine, Paris | France, 2013-2018). Exchange year in Moscow Architecture Institute (MArkHI, Moscow | Russia, 2016-2017)

Experience: Intern in architecture office Loci Anima (Paris | France, 2017); Intern and employee in architecture office Auer Weber (Munich | Germany, 2018-2020)

Motivation: “Combine my prior knowledge in architecture with building-analyses developed through deep-learning methods. Learn tools allowing us to move towards a more sustainable urban design.”

Carolin Bachert



Email: carolinbachert@gmail.com

Background: BSc in Geography with a specialisation in GIS and Cartography (University of Vienna, Vienna | Austria, 2016 – 2020)

Experience: Intern at the National Mapping Agency Austria (Vienna | Austria, 2019), Tutor for two GIS courses at the University of Vienna (Vienna | Austria 2019-2020)

Motivation: “I am very excited to work together with a company on a problem in the field. This project is a great opportunity to apply what we have already learnt and to dive deeper into topics that we’ve only heard theory about so far.”

Camilo Caceres



Email: ca.caceres3@gmail.com

Background: BSc in Environmental Engineer & BSc in Industrial Engineer (Universidad de los Andes, Bogota | Colombia, 2012-2017).

Experience: Data Analyst in PedidosYa company owned by Delivery Hero from Germany (Montevideo | Uruguay, 2018-2020).

Motivation: “I enjoy working with data for sustainable development and I think this project will bring these two aspects together.”

Özge Tufan



Email: ozgetufann97@gmail.com

Background: BSc City and Regional Planning, Minor GIS and Remote Sensing (Middle East Technical University, Ankara | Turkey, 2016 - 2020).

Experience: Intern at the Directorate General of Geographic Information Systems (Ministry of Environment and Urbanisation of Turkey, 2019).

Motivation: “Being able to integrate my prior knowledge and what I have learnt so far during the masters while experiencing new areas such as deep learning makes this project highly appealing to me.

Ondrej Veselý



Email: ondrej-vesely@seznam.cz

Background: BSc Architecture (Brno University of Technology, Bauhaus Universität Weimar)

Experience: Researcher at Digital Resilient Cities (Austrian Institute of Technology, Vienna | Austria, 2017-2019), Computational Developer (OMRT, Amsterdam | Netherlands, 2019 – current)

Motivation: “Netherlands have a very advantageous position in the sense of the amount of data on the built environment being openly available and working on any project that builds upon this foundation is simply just a pleasure”.

8.2 Roles

Team members are given specific roles during the project to ensure an efficient strategy for the allocation of tasks. Along with their responsibility to jointly develop the end-product and to participate in any discussions/meetings with the client, the team members are responsible for the following roles:

Project Manager | Irène Apra

The Project Manager keeps track of the tasks completed and/or in progress and warns the team members in case of unfinished tasks on time. She regularly checks the decided timeline to ensure that all the steps are completed without delay, and she also arranges the content of each meeting in accordance with the current progress.

Communication Manager | Carolin Bachert

The Communication Manager is responsible for contacting the client and the mentors to arrange meetings and/or discussion sessions. She also takes notes from the meetings in an organized way and shares them with the rest of the team, the client, and the mentors.

Technical Managers | Camilo Caceres & Ondrej Vesely

The Technical Managers make sure that the correct pre-processing steps are taken for the input data. They are in constant discussion with the rest of the team to choose the best possible option for the software to use and the steps to be taken to reach the end-product.

Report Manager | Özge Tufan

The Report Manager is responsible for creating the report layout to present all the information in the best possible way. She regularly checks the report to fix any grammar mistakes and/or inconsistencies and marks the parts that need to be revised.

9 Appendix II

9.1 Additional figures

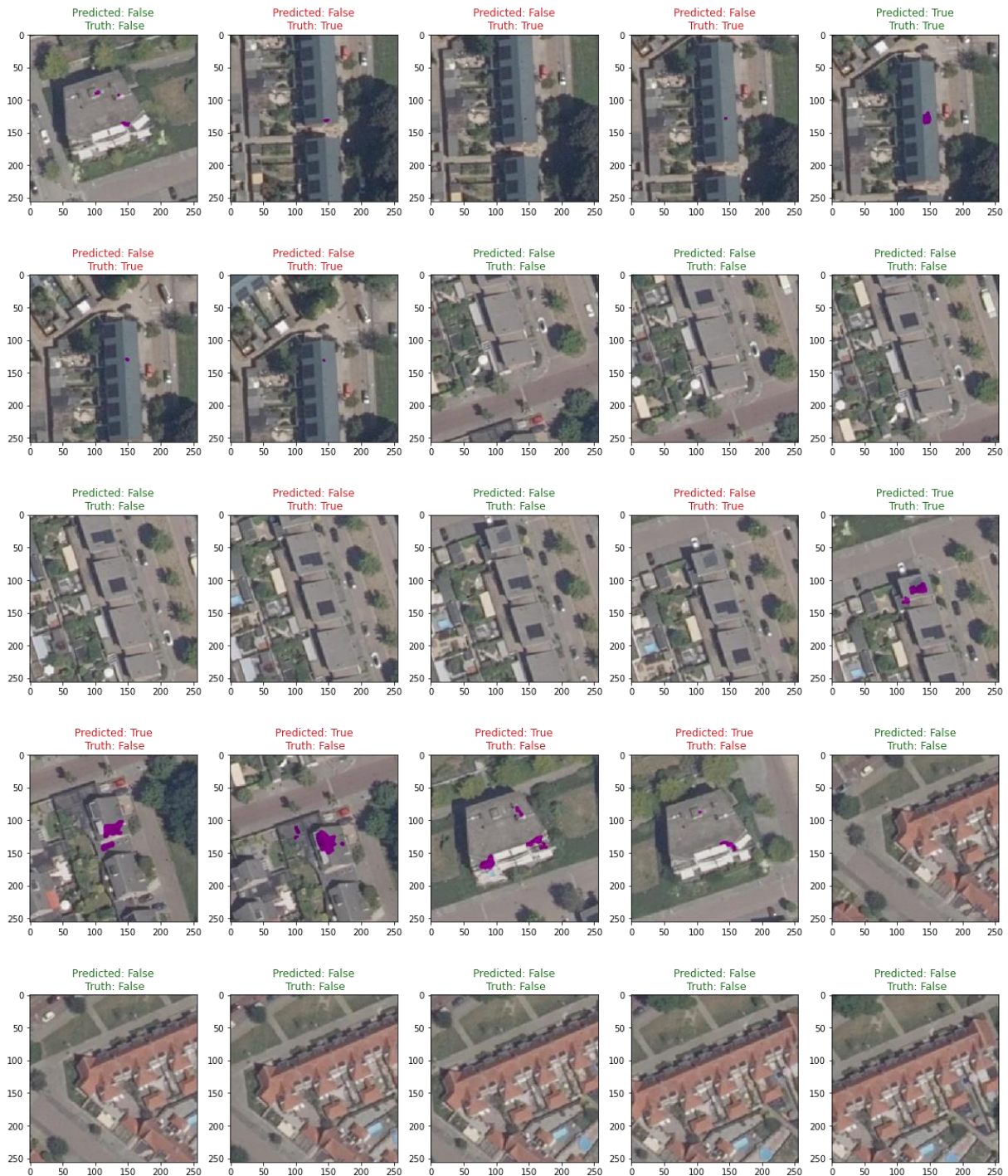


Figure 43. Results for the Brink provided 50 building data set (part 1/2).



Figure 44. Results for the Brink provided 50 building data set (part 2/2).



Figure 45. Results of detecting obstacles with geometry-based and image classification.