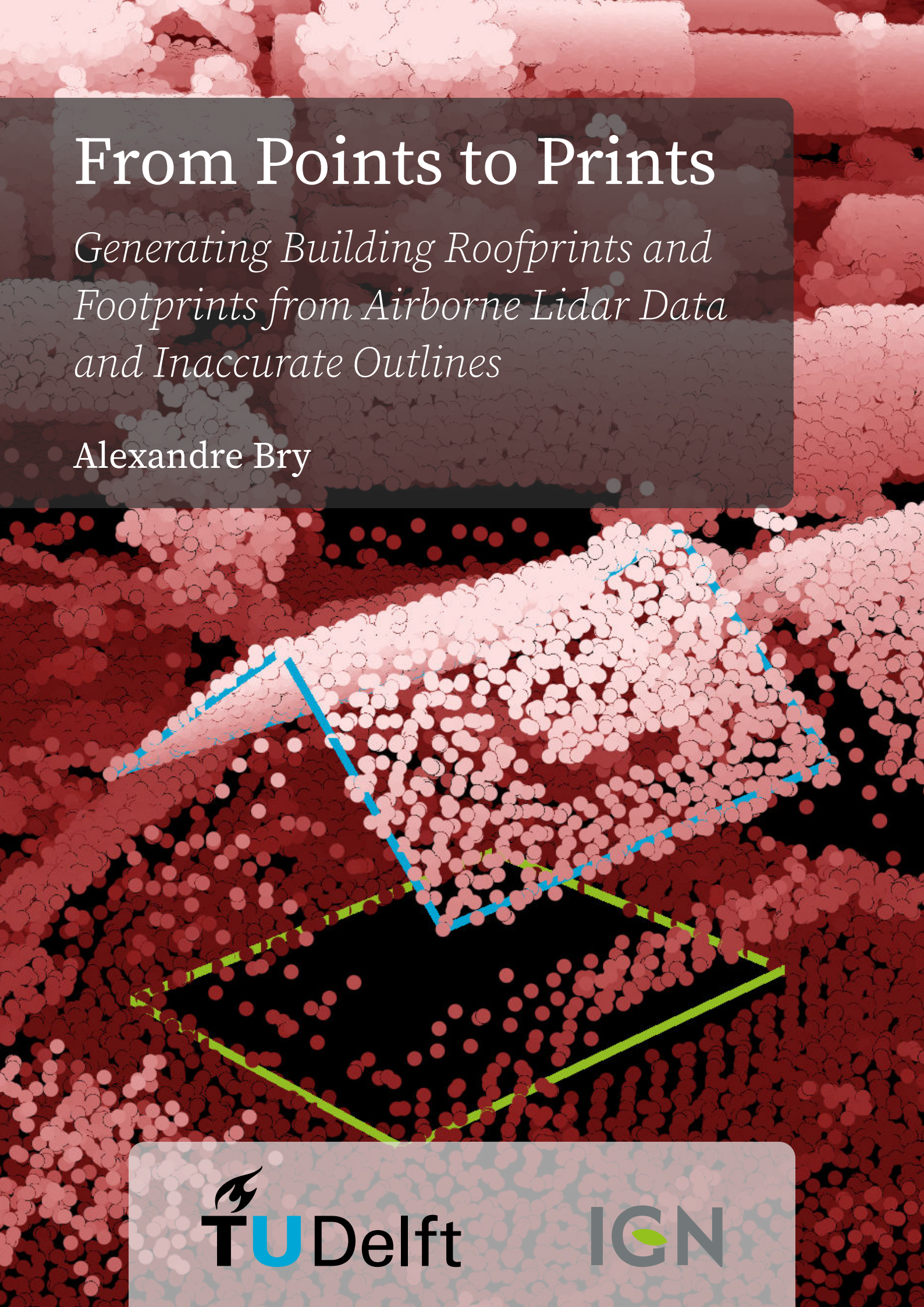
From Points to Prints

Generating Building Roofprints and Footprints from Airborne Lidar Data and Inaccurate Outlines

Alexandre Bry



TU Delft

IGN

From Points to Prints

Generating Building Roofprints and Footprints from Airborne Lidar Data and Inaccurate Outlines

MSc Thesis Report

by

Alexandre Bry
6277535
abry@tudelft.nl

5 June 2026

1st TU Delft supervisor: Hugo Ledoux
2nd TU Delft supervisor: Ravi Peters
IGN supervisor: Bruno Vallet

Abstract

In recent years, several methods have been successfully developed to create 3D building models in LoD2.2 from ALS point clouds. These methods however rely on 2D horizontal building outlines whose definitions and quality vary significantly. There are mainly two types of outlines: roofprints and footprints, and having access to both of them allows to reconstruct the façades and the roof independently, therefore improving from LoD2.2 to LoD2.3 models. Moreover, this distinction also allows more precise analyses depending on the use case. Therefore, we present a method to construct both a roofprint and a footprint from ALS data and existing inaccurate outlines. By first precisely deforming and aligning the outline on the roof to turn it into the roofprint, our method can then accurately identify the few ground and façade points available, if there are any, in order to produce a footprint coherent with the roofprint. Our method preserves the validity of the polygons and the topological relations between them. It has been tested successfully on the French national datasets BD TOPO and LiDAR HD.

Acknowledgements

First, I want to thank my two main supervisors Hugo Ledoux and Bruno Vallet for their steady support over the whole course of the thesis, and for their very valuable ideas and suggestions. The many discussions we had were crucial to take pragmatic and thoughtful decisions all along, and to keep my motivation high.

I want to extend my acknowledgements to everyone who followed my thesis for the regular monthly meetings and came with interesting and refreshing ideas with their viewpoints from the outside. This includes Ravi Peters as my second supervisor for TU Delft, but also Emmanuel Séguin, Teng Wu, Florent Geniet and François Becirspahic on the side of the IGN.

Then, I have to thank all my colleagues from IGN especially in the LASTIG and in the SIMV department who welcomed me during the 5 months I spent there. Because in an intense and self-centred work like research, taking some time for breaks and to enjoy other peoples company is necessary to keep the mental and the ideas flourishing.

Finally, I am really grateful to Élodie, to my family and to my friends, who always supported me whether mentally or financially. They enabled me and encouraged me to live this one-and-a-half year experience in the Netherlands, and welcomed me back in France as if I had never left.

Outline

1. Introduction

In 2024, the Institut national de l'information géographique et forestière (IGN)[1] and two other French public entities have launched an initiative to bring together partners who can contribute to the development of a nation-wide digital twin (IGN, 2024a). This initiative was officially launched in April 2026 under the name of JUNN. The idea behind a digital twin is to gather in one interface many sources and types of data, in order to make them more discoverable, accessible and usable, and by doing so allow for simulations and discussions based on real, accurate and complex data. Ecological planning and sustainable land use are presented as some of the priorities this project should accommodate. In a different post, it is explained how the LiDAR HD[2] — the first project to collect high-density point clouds on almost the whole territory of France — is central to the future digital twin (IGN, 2024b). This comes from the unprecedented precision that it brings compared to previous data used and maintained by the IGN.

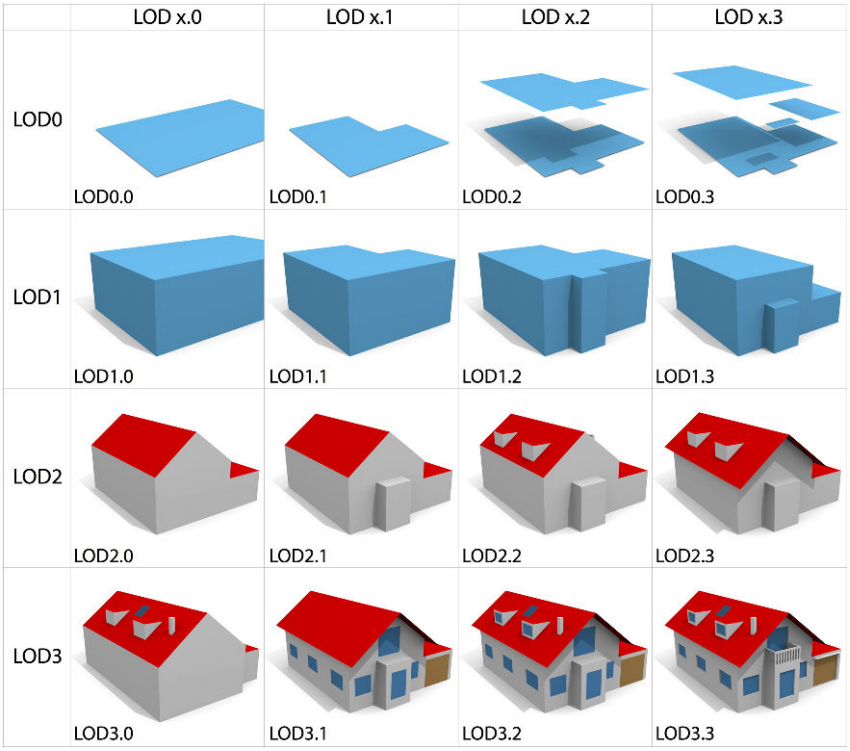


Figure 1: Visual example of the refined LoDs for a residential building (Biljecki et al., 2016).

One of the many components of the future digital twin is buildings. Many algorithms have been developed to try to reconstruct structured and accurate 3D building models from various data sources, including point clouds. To characterise the properties of the buildings created by these different methods, 5 Levels of Detail (LoDs) have been introduced by Gröger et al. (2012), before being extended into 16 LoDs described by Biljecki et al. (2016) and are illustrated in Figure 1. One of the current state-of-the-art algorithms was created by researchers at Delft University of Technology (TU Delft) and is called roofer[3] (Pađen et al., 2024). It was applied to the whole of the Netherlands to create the 3DBAG, the first complete dataset of Dutch buildings in LoD2.2 (Peters et al., 2022). This algorithm however requires two input data: a dense Airborne Laser Scanning (ALS) 3D point cloud and 2D building outlines. In the Netherlands, the Actueel Hoogtebestand Nederland (AHN) was used as the point cloud and the Basisregistratie Adressen en Gebouwen (BAG) was used as the outlines.

[1] https://www.ign.fr/institut/identity-card
[2] https://www.data.gouv.fr/datasets/nuages-de-points-lidar-hd
[3] https://github.com/3DBAG/roofer

However, talking about a building outline is imprecise, as there are many different ways to create a 2D horizontal representation of a building. There are mainly two kinds of 2D building outlines that we consider in this thesis:

- the footprint, defined as the horizontal 2D polygon obtained by projecting vertically the *outer walls/façades* of a building,
- the roofprint, defined as the horizontal 2D polygon obtained by projecting vertically the *roof* of a building and taking its outer boundary.

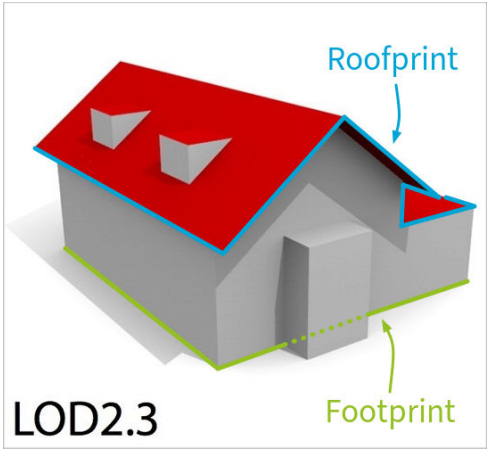


Figure 2: Visual definition of the roofprint and footprint on the LoD2.3 building example by Biljecki et al. (2016). The roofprint in this picture displays the edges of the roof used for the definition and needs to be projected vertically to get the actual roofprint.

These definitions are illustrated in Figure 2. In the rest of this document, I will use the terms roofprint and footprint when possible, and otherwise use outline for a more general term. Usually, due to roof overhangs and gutters, the roof extends further than the walls, meaning that the footprint is included in the roofprint. As an example, the roofprint is what matters in estimating solar energy potential — in combination with other factors such as roof orientation and angle. But in many other applications — such as taxes or energy consumption — an accurate estimation of the area and volume of buildings is necessary, which will be better with a footprint.

The IGN already has a dataset containing building outlines, called BD TOPO. However, this dataset has some issues, that can be explained by how it was historically built from different sources. Most outlines (88.54%) come from terrain measurements and are therefore footprints, but most of the rest (10.09%) come from aerial image detection and are therefore roofprints. A small proportion was automatically generated from the LiDAR HD (0.36%) and the origin of the rest is not specified (1.01%)[4]. The georeferencing of these building outlines is often wrong by up to a few meters. This makes combining them with correctly georeferenced point clouds more complicated.

All in all, the current context combines:

- an objective to build a digital twin of France, including 3D buildings with algorithms which would benefit from or require correct building outlines (such as roofer),
- newly available data with high precision and correct georeferencing (LiDAR HD),
- the example of the Netherlands where a great dataset of 3D building models was built from similar point cloud data (3DBAG from AHN),
- an interesting and not yet fully explored question of the possibility of extracting both an accurate footprint and an accurate roofprint from point clouds,
- an existing dataset that provides nation-wide and potentially great data but is however missing harmonisation and precise georeferencing (BD TOPO).

[4]These numbers come from the 2026-03-15 version of the BD TOPO.

Therefore, the research question of this thesis is:

> ***How to generate coherent building roofprints and footprints from high-density ALS point clouds and existing imprecise outlines?***

A few more specific sub-questions are also addressed in this thesis:

- How to identify and use the points on roof edges in ALS point clouds?
- How to identify and use the points in an ALS point cloud that contain information about the façades despite their sparsity?
- How to deform an imprecise outline with global and local transformations while preserving the angles of the edges?

The core of this report is structured as a research paper, targeted at an expert audience, which can be found in *Chapter 3*. To make its content accessible to non-experts, preliminary materials on several different topics are provided in *Chapter 2*, although they still assume some GIS-related knowledge, to the level of the TU Delft Master of Geomatics. Finally, the report ends with a conclusion and a presentation of potential future work on the topic.

2. Preliminary Materials

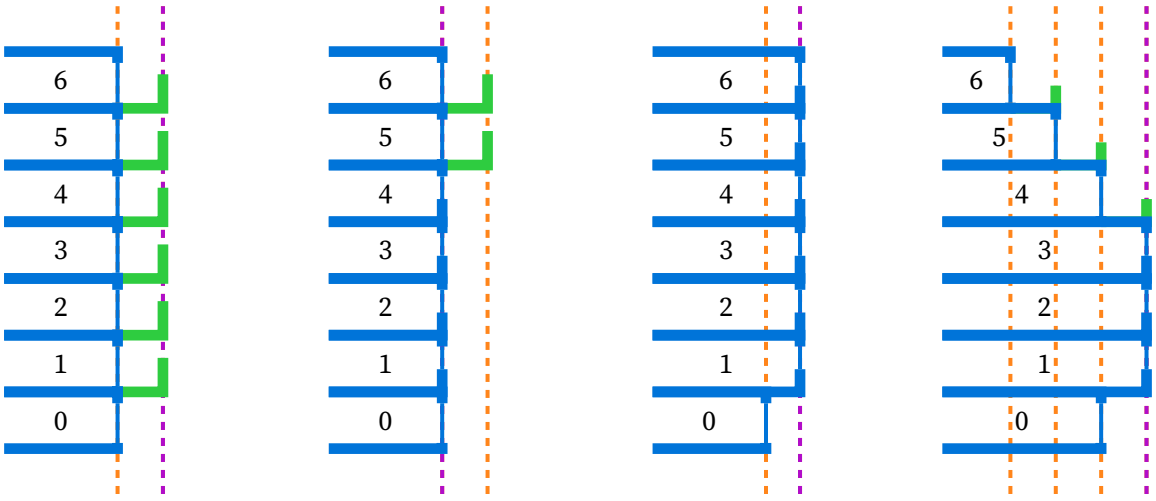
This chapter aims at introducing the knowledge and vocabulary necessary to understanding the core of the thesis (*Chapter 3*). It is intended for readers who are not expert in the topic at hand, however the *Chapter 3* is self-contained.

2.a. Roofprints and footprints

As presented in *Chapter 1*, the difference between roofprints and footprints is central for this thesis. One of the reasons why both of them are important is that different sources of data often make it easier to get one of the two:

- surveyors in the field mostly use the walls and therefore measure the footprint,
- experts working on aerial imagery can only use the roof as some walls will not be visible, meaning that they measure the roofprint,
- ALS point clouds (such as the LiDAR HD and the AHN) give many points on the roofs and therefore make it easier to extract the roofprint,
- Terrestrial Laser Scanning (TLS) and Mobile Laser Scanning (MLS) point clouds give many points on the walls and therefore make it easier to extract the footprint.

Even with the definitions given before, some aspects still need to be clarified to be able to unequivocally draw a roofprint and a footprint for every building. First, the inclusion of balconies and other outdoor elements of buildings is unclear in many scenarios, as illustrated in Figure 3. In blocks of flats with multiple storeys where every storey except the ground floor has the same exact balcony (see Figure 3a), does the end of the balcony become the position of the façade? Sometimes, the whole building is extruded horizontally except the ground floor (see Figure 3c), in a structure which looks like balconies except they are not open, are these considered as balconies? There are many more questions to consider when trying to properly characterise 2D outlines to create a coherent dataset, and many of these questions do not have a right or wrong answer: the best answer depends on the final application. There are even cases, such as buildings with non-vertical façades, which completely break the purpose of defining a simple and unique 2D outline.



(a) Same balcony at every storey. *(b) Only a few balconies.* *(c) Extruded façade.* *(d) Different façades with some balconies.*

Figure 3: Illustration in profile view of the unclear definition of the façade with balconies. The buildings are in blue with thin lines representing windows and the balconies in green. Dotted lines represent potential façades and the purple one is the one we chose in our definition.

In our case, the potential usage of footprints and roofprints which is considered is the possibility to turn LoD2.2 buildings into LoD2.3 by modelling roof overhangs, as shown in Figure 1 (*Chapter*

1). Since methods like roofer use the points from the roof to reconstruct buildings, they require a roofprint to work properly. But then, since there is only a sparse distribution of points on the façades (if there are any points), identifying planes is often impossible, and the roofs are simply extruded down, creating LoD2.2 buildings. If the footprint was known precisely, it would be possible to instead extrude it up to the roof, assuming that it is contained in the roofprint.

Therefore, our definitions for footprints and roofprints should be the ones that would lead to the most accurate reconstruction of the buildings in 3D. In practice, this means that the façades are defined as the most prevalent vertical surfaces, or in other terms, the vertical plane which has the largest intersection with the actual building (see Figure 3). This will often be the end of the balcony when the exact same balcony is present at every storey such as in Figure 3a. With this definition, the balconies have a significant advantage because in ALS data, a balcony occludes the façade below it in the same way as roof overhangs, so they prevent the potential façade behind them to be seen.

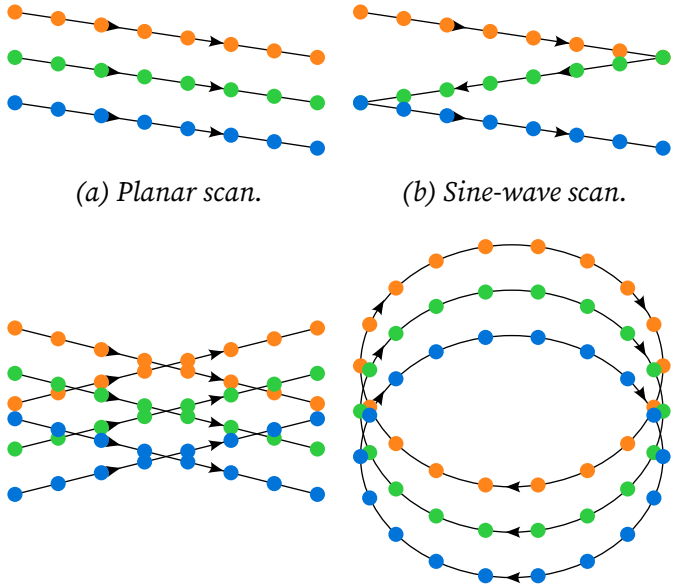
2.b. ALS point clouds

Airborne Laser Scanning (ALS) point clouds are sets of points acquired by a Light Detection and Ranging (lidar) sensor mounted on a flying vehicle. These lidar sensors emit laser pulses at a very high frequency with a varying direction. At the same time, the vehicle usually moves as much as possible in a straight line, at a constant speed and constant height. Each pulse results in a continuous return signal, where peaks correspond to objects on the pulse trajectory, and are called echos. By combining the trajectory of the scanning vehicle, the angle of the lidar sensor and the time between emitting the pulse and receiving the returned peak, a 3D position can be computed for each echo, corresponding to an object in the scene.

The characteristics of the point clouds obtained with this method depend on many parameters (Wu et al., 2026):

- the pulse frequency: a higher pulse frequency results in denser point clouds along the scan lines,
- the altitude of the vehicle: a lower altitude results in denser point clouds covering smaller areas,
- the speed of the vehicle: a slower vehicle results in denser point clouds along the direction of the vehicle,
- the type of lidar sensor: one or multiple fibres (laser emitters) with different motions (planar rotations, conical rotations, oscillations, or even more complex motions) result in different structures.

During this thesis, the main focus was the LiDAR HD, which present similar characteristics to its Dutch counterpart the AHN. However, multiple different ALS sensors have been used by the different contractors who took part in the acquisition of the LiDAR HD. Based on the work of Wu et al. (2026), the different motions for the sensors found in the LiDAR HD are illustrated in Figure 4.



(a) Planar scan. *(b) Sine-wave scan.*

(c) Cross scan. *(d) Circular scan.*

Figure 4: Illustration of different types of ALS sensors, with multiple consecutive scan lines in orange, green and blue (the scanning vehicle is moving towards the bottom of the figures), inspired from (Wu et al., 2026).

Even though they look completely different, these share common characteristics. First, the motions of the fibres are continuous except for potential jumps between the end of a scan line and the beginning of the next scan line. This means that except for rare exceptions, two consecutive pulses are very close geometrically. Then, the directions of the pulses are almost vertical, with an angle rarely reaching more than 20°. This property will be crucial for our method, as will be explained later. Finally, we can always reconstruct a sort of multi-dimensional topology over the point cloud, because:

- all the echos acquired from a single pulse are ordered,
- all the pulses acquired along a single scan line are ordered,
- all the scan lines acquired along a single flight strip are ordered.

Therefore, one can move along any of these three dimensions to make computations in a comparable and predictable way. Compared to the inherently chaotic nature of 3D point clouds, with every point having potentially a very different geometric neighbourhood, this can facilitate studying the properties of the point cloud. More specifically, since the flight strips often overlap, they are merged into a point cloud which topological structure can be defined as illustrated in Figure 5. In the LiDAR HD, a tile corresponds to a square area of 1 km by 1 km.

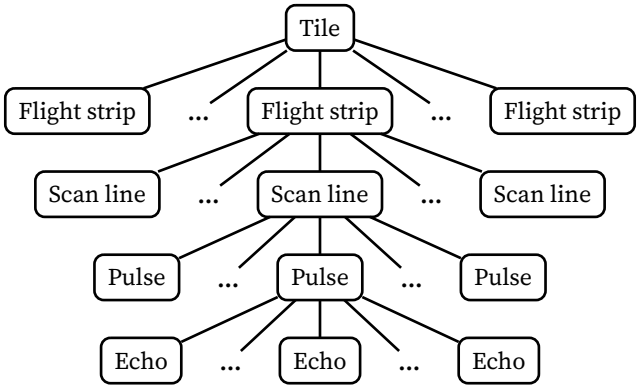


Figure 5: Illustration of the topological structure of a tile of an ALS point cloud.

2.c. Polygon deformation and topology

Polygons are complex shapes on which many different operations are possible. A significant problem about them is the complexity associated with verifying their validity. For example, a simple and brute-force approach to verifying that no pair of edges intersect in a polygon requires $n(n-1)$ checks if the polygon has n edges.

In that regard, modifying a polygon instead of creating a new one from scratch can have some good properties. Depending on the operations that are performed, checking the validity of the polygon can be very simple, or even unnecessary. This is the case for most of the simple global operations on polygons: translations, rotations, scaling and even skewing, as long as they are applied globally, will never break the validity of a polygon. This is because these are rigid transformations: the relative positions of the vertices remain unchanged. However, these operations are very limited in their capabilities: their global aspect prevents them from patching specific regions of the polygons in different ways.

This is where local operations are necessary. For these operations, the two most basic elements to work with are vertices and edges. Choosing between the two is crucial and completely depends on the applications. This choice and the number of constraints enforced in the movement determine how much freedom and how much complexity will be associated with the operation. Here are a few examples, among many different possibilities :

- translating a point freely gives 2 dimensions of freedom: one dimension for each axis (x and y).
- translating the line associated with an edge along its normal gives only 1 dimension of freedom: the distance between the initial and the final line.
- translating and rotating the line associated with an edge gives 2 dimensions of freedom: one for the translation distance and one for the rotation angle.

For most of these local operations, there will be bounds to how much each element can be modified before breaking the validity of the polygon. These bounds correspond to a simple interval if there is only 1 dimension of freedom, and become more and more complex as the number of dimensions of freedom increases. On top of that, these bounds are not independent for every point or edge. Moving one point has an influence on how far its neighbour can be moved at the same time without breaking the polygon. Therefore, the complexity of these operations increases when performing many of them at the same time, and depending on the type of operation, it can be very complex to express and manipulate these relations.

For this thesis, the constraints that we imposed on the initial polygons were the following (more details in *Chapter 3*):

- do not rotate any edge,
- do not flip any edge (see Figure 6).

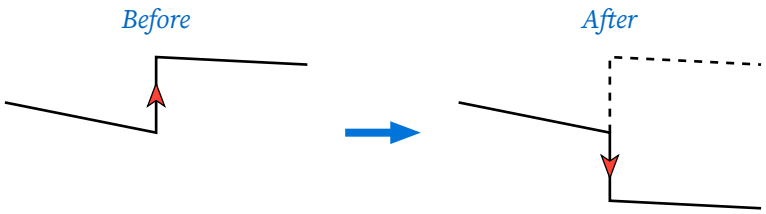


Figure 6: Illustration an edge being flipped by the translation of a neighbour edge.

With these constraints in mind, the simplest way to understand and express the remaining movements is the following. First, each edge is replaced by the infinite line that passes through it. The polygon is therefore defined by a set of ordered lines, which intersections define the vertices (i.e. the two ends of the edges). The only modification allowed is to replace any line by any parallel line, as long as this does not break the validity of the polygon. In practice, this means that there can be

a maximum distance to which this line can be shifted in both directions (see Figure 7). The two directions have different maximum shifts, and only one of them can be infinite.

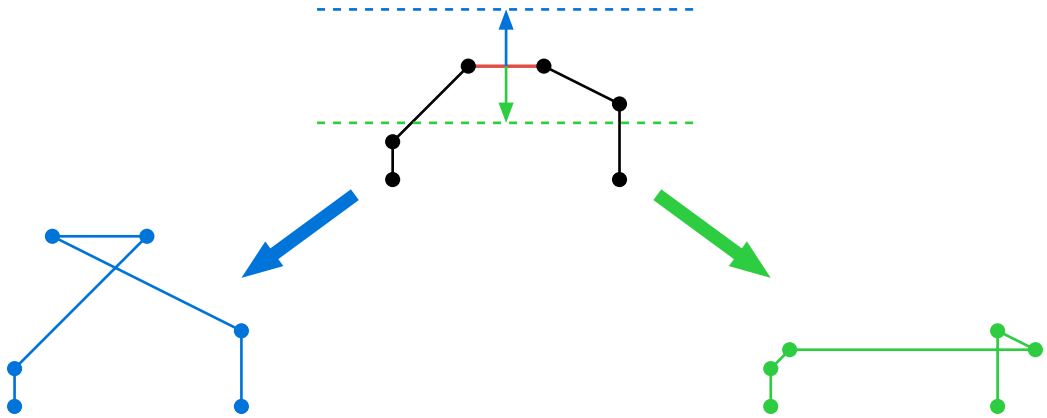
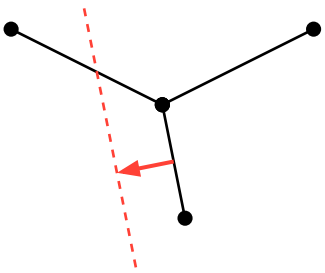
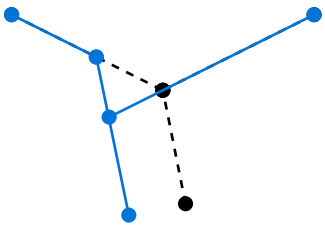


Figure 7: Illustration of an edge being shifted too far in the two directions.

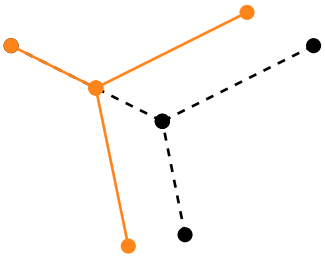
Additionally, since the polygons at hand represent buildings, several different polygons may share edges in situations where the buildings are adjacent. If that is the case in the initial configuration of polygons, our method also tries to keep these adjacencies. One way to consider it is to say that edges that are collinear should remain collinear. This is a rather limited constraint, as it still allows shared vertices to be split into two different vertices. The stronger version of this constraint is to enforce all shared vertices to remain the same vertex. This results in harder constraints that preserve the topology better at the cost of some freedom on the movements of the lines. In this case, if a vertex is shared by 3 non-collinear edges, keeping the vertex unique when shifting one of the edges implies to move at least one of the other edges, even if it was not necessary for the validity of the polygon. Figure 8 illustrates these different constraints on a simple example.



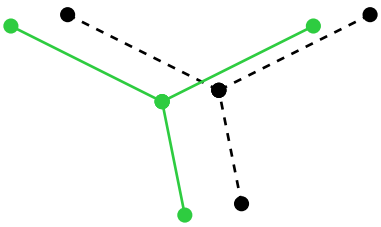
(a) Initial situation



(b) Softer constraint on shared edges.



(c) Harder constraint on shared vertices: a solution.



(d) Harder constraint on shared vertices: another solution.

Figure 8: Illustration of an edge being shifted with one of its vertex shared by three edges. Three different results are shown: Figure 8b displays the softer constraint on shared edges, while Figure 8c and Figure 8d show two different solutions to the harder constraint on shared vertices.

2.d. Optimisation and energy

Once a set of constraints are defined for the deformation of the polygons, this results in an infinite set of potential polygons. The goal is then to find the best polygon for a specific use case. This is called optimisation, and it usually needs two elements that will work together:

1. a criterion, also called energy, which is used to estimate how good a given polygon is,
2. an exploration strategy, which tries to navigate in the complex space of potential solutions to find the polygon giving the best energy.

In this report, the best energy is assumed to be the lowest energy, so the goal of optimisation is to find the polygon that minimises the energy. There are infinitely many ways to create a formula for an energy, and these different ways will result in a different polygon being the best. For the optimisation of the roofprints, the energy should be something that is minimal when the polygon corresponds perfectly to the edges of the roof. More details are given in *Chapter 3* about the energy used in our method.

Often, the energy can be a high-dimensional function, with many parameters to optimise together. In our case, with the constraints that we set on the polygons, there are usually n variables to optimise at the same time if the polygon has n vertices. Moreover, we consider that our starting point is already a good first guess: the initial outlines of our method are assumed to be up to a few meters off, with only small local deformations of the polygon being necessary. In this situation, the simplest procedure is to optimise every single parameter iteratively. In practice, this means:

1. picking one line,
2. computing the energy of the polygon obtained after shifting this line by many different amounts in both directions,
3. picking the polygon with the best energy as the new polygon,
4. starting again with a new line and the new polygon.

This kind of iterative and simplified process can however lead to completely incorrect final configurations. One of its downsides is that the first iterations are the most crucial: since it may start relatively far from its optimal solution, the first iterations may find their minimal values in the wrong direction, which can snowball into finding an incorrect locally minimal configuration. Moreover, the order of the iterations matter: the result may be completely different depending on which lines were treated first. This means that picking a good order is important, and also that it is possible to try different orders and pick the best final solution. In our case, treating the longest edges first seems to be a smart choice, as longer edges are relatively less impacted by their initial incorrect translation than smaller edges. But the main advantage of this method is its simplicity and its speed: it reduces the optimisation of a complex function into a series of simpler one-dimensional problems, which is computationally very quick.

2.e. 3D city modelling

Even if this thesis is focussed on producing 2D polygons, its outputs could greatly improve the generation of 3D building models. To understand why, it is necessary to know about the current existing methods to produce LoD2.2 buildings.

In the last few years, several methods have demonstrated their ability to produce high-quality LoD2.2 building models from two inputs: a high-density ALS point cloud and building outlines (Bauchet et al., 2024, Huang et al., 2022, Pađen et al., 2024). Since ALS data is acquired by planes shooting rays that are mostly vertical, it contains high density of points on horizontal surfaces (such as roofs) and low density on vertical surfaces (such as façades), as illustrated in Figure 9. This low density makes it difficult to use for the façades the same plane fitting techniques that are used for the roofs. Furthermore, the roof overhangs create occlusion which reduces further the amount of points on the façades. For these reasons, the methods mentioned above focus on creating a precise

and accurate model of the roof and simply extrude down the roof to get the façades, which results in LoD2.2 models.

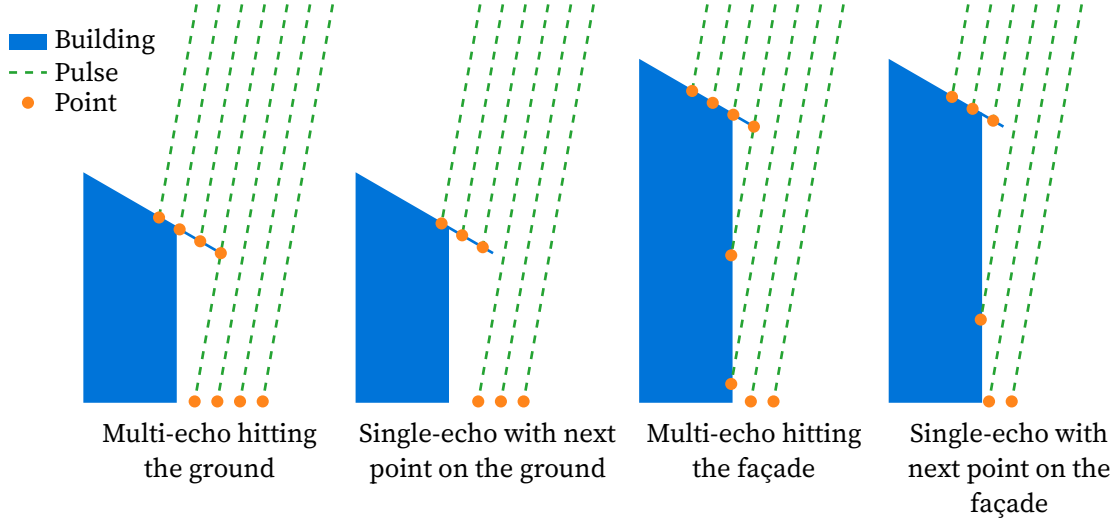


Figure 9: Illustration in profile view of the lack or sparsity of points on the façades.

In all of these methods, the outlines are used initially to select which points in the point cloud should be used for the reconstruction of each building. This is a crucial step, as having too many or too few points can lead to incorrect results. Therefore, to reconstruct a perfect roof, a roofprint would be the best input for these methods, even though the papers mention footprints due to the difference often not being made. On top of that, precise outlines could help extracting precise and well-oriented edges for the roof model, as extracting precise directions from ALS data is difficult.

Besides guiding the reconstruction process better with a roofprint, these methods could also use a footprint to create more detailed models. Instead of using the outline of the final 3D roof model to extrude down and get the final 3D building model, these methods could use the footprint and extrude it up until it reaches the roof model. This would allow to very simply move from LoD2.2 to LoD2.3, assuming that the footprints are correct.

3. Paper

This page is left intentionally blank, with the paper starting in the next page. The paper is fully self-contained, with its own introduction and conclusion, its own numbering and its own references.

From Points to Prints: Generating Building Roofprints and Footprints from Airborne Lidar Data and Inaccurate Outlines

Alexandre Bry[1,2], Bruno Vallet[1], Hugo Ledoux[2]

[1] LASTIG, Université Gustave Eiffel, IGN – Champs-sur-Marne, France – (alexandre.bry, bruno.vallet)@ign.fr
[2] 3D Geoinformation Group, Delft University of Technology (TUD) – Delft, Netherlands – h.ledoux@tudelft.nl

Keywords: Building roofprints, Building footprints, Airborne lidar, Roof overhangs, Semi-rigid polygon deformation.

Abstract

In recent years, several methods have been successfully developed to create 3D building models in LoD2.2 from ALS point clouds. These methods however rely on 2D horizontal building outlines whose definitions and quality vary significantly. There are mainly two types of outlines: roofprints and footprints, and having access to both of them allows to reconstruct the façades and the roof independently, therefore improving from LoD2.2 to LoD2.3 models. Moreover, this distinction also allows more precise analyses depending on the use case. Therefore, we present a method to construct both a roofprint and a footprint from ALS data and existing inaccurate outlines. By first precisely deforming and aligning the outline on the roof to turn it into the roofprint, our method can then accurately identify the few ground and façade points available, if there are any, in order to produce a footprint coherent with the roofprint. Our method preserves the validity of the polygons and the topological relations between them. It has been tested successfully on the French national datasets BD TOPO and LiDAR HD.

1. Introduction

The geometry of buildings in 2D maps and databases is generally represented through 2D polygons describing either the footprint or the roofprint of the building, while in 3D buildings are typically represented by surfacic or volumic polyedral models. To classify building models, 5 Levels of Detail (LoDs) have been introduced by Gröger et al. (2012), before being extended by Biljecki et al. (2016) into 16 different levels of precision of these models with generally increasing complexity and fidelity to the reality. These LoDs are illustrated in Figure 1. As pointed out by the authors, the crucial properties of a building model depend completely on the application. For example, an accurate footprint is more important than a proper roof shape to estimate the exact internal area available in a building. Another example is the volume of a building which requires the combination of a correct roof and correct façades for precise computations. These values are useful for energy estimations, property tax calculation, real estate valuation, and population counts (Boeters et al., 2015, Kaden and Kolbe, 2014).

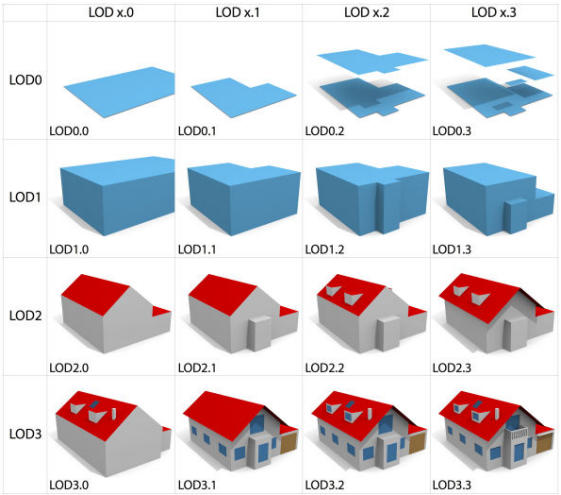


Figure 1. Illustration of the LoDs (Biljecki et al., 2016).

One distinction that is often overlooked and is not present even in the 2D LoD0.x models is the difference between the footprint and the roofprint of a building. Even when representing buildings with a simple 2D polygon corresponding to the horizontal area it occupies, multiple choices can be made. These choices are often data-driven: when measurements are made by experts directly on the terrain, the delimitations of the façades close to the ground are the easiest to measure, leading to the creation of a footprint. When measurements are made from aerial data – usually images or Light Detection and Ranging (lidar) – the roof is the simplest element to identify and annotate, leading to the creation of a roofprint. Other decisions also have to be made when creating a single horizontal polygon for a building, the simplest example being the inclusion or exclusion of other overhanging objects such as balconies.

The difference between the roofprint and the footprint is only meaningful when the roof extends beyond the façades of the building or in cases of underpasses through the building. The parts of the roof beyond the façades are called roof overhangs and can be found in many buildings in Europe. These appear in the LoD classification at levels 2.3 and 3.1. Sometimes, as in the example of the BD TOPO, buildings are represented as a mix of footprints and roofprints. This awkward situations is explained by the fusion of two datasets that were historically created and maintained by two different French institutions. The first one was focused on taxes and therefore looked for precise areas, with footprints measured by experts on the terrain. The second used the best available data at that time at the scale of France, which was aerial imagery, leading to the creation of roofprints.

In the last decade, growing interest for high-density Airborne Laser Scanning (ALS) point clouds has lead many European countries to producing their own datasets. This includes among others Denmark, Finland, France (LiDAR HD), the Netherlands (Actueel Hoogtebestand Nederland (AHN)) and Spain. In the last

few years, several methods have demonstrated their ability to produce high-quality LoD2.2 building models from two inputs: a high-density ALS point cloud and building roofprints (Bauchet et al., 2024, Huang et al., 2022, Pađen et al., 2024). These methods are currently limited to LoD2.2 due to the very nature of ALS data. Since it is acquired by planes shooting rays that are mostly vertical, it contains high density of points on horizontal surfaces (such as roofs) and low density on vertical surfaces (such as façades), as illustrated in Figure 2. This low density makes it difficult to use for the façades the same plane fitting techniques that are used for the roofs. Furthermore, the roof overhangs create occlusion which reduces further the amount of points on the façades.

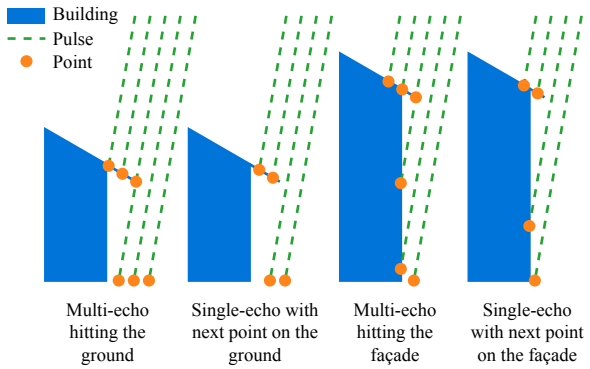


Figure 2. Illustration in profile view of the different scenarios for consecutive pulses around a roof edge.

Therefore, with these methods, the façades are most often only a downwards extrusion of the roof. This results in incorrectly positioned façades, and the volume of the building is often over-valued. Some applications like lighting simulations or texturing of the façades would benefit from correctly positioning both the end of the roof and the façades, which is not possible for building with overhangs in LoD2.2. An illustration of the capabilities of these methods is the 3DBAG, which contains more than 10M buildings covering the whole of the Netherlands (Peters et al., 2022) and was generated using the AHN and the Basisregistratie Adressen en Gebouwen (BAG) as inputs, the Dutch equivalents of the LiDAR HD and the BD TOPO respectively. The 3D models have very accurate roofs, but the façades depend on the input that was used, either a footprint or a roofprint.

As explained in Section 2, there is little research on the distinction between footprints and roofprints, as well as in the ability to produce both from ALS data. This is also the case for modelling the roof overhangs to create LoD2.3 buildings, which is the 3D counterpart of this 2D distinction.

In this context, we propose in this paper a method to address at the same time the registration of building outlines and the creation of both a roofprint and a footprint for each building in Section 3. Our method takes as input building outlines (which can be anything between a footprint and a roofprint) and high-density ALS data, to create consecutively for each building a roofprint and a footprint aligned with the input point cloud (see Section 3.1).

The method was tested on the two French datasets BD TOPO and LiDAR HD with promising results displayed in Section 4. These experiments displayed how the method is capable of repositioning the initial imprecise outlines while deforming them if necessary. On all buildings except low sheds, when comparing to the ground-truth in our validation dataset, the scores significantly improved after 3 iterations of our algorithm compared to the initial outlines in the BD TOPO:

- the average IoU with the ground-truth polygon went from 77.57% to 90.51%,
- the average distance to the ground-truth centroid went from 0.846 m to 0.354 m,
- the average Chamfer distance to the ground-truth polygon went from 0.557 m to 0.22 m.

The best results were reached with the isolated, medium-sized houses, which are the main target of this algorithm:

- the average IoU with the ground-truth polygon went from 74.45% to 93.48%,
- the average distance to the ground-truth centroid went from 0.681 m to 0.117 m,
- the average Chamfer distance to the ground-truth polygon went from 0.55 m to 0.114 m.

The main contributions of this paper are:

- a simple but effective method to extract and use evidence of roof edges from ALS point clouds with topological-aware operations,
- a basic method to extract and use evidence for the footprints despite their sparsity,
- an algorithm to deform polygons while keeping the angles of the edges and the connections between adjacent polygons,
- a fast and iterative optimisation method with specific variations for roofprints and footprints,
- all combined in a method to create coherent polygonal roofprints and footprints directly from ALS point cloud data and an initially imprecise but topologically accurate outline.

2. Related work

The topic of roof overhangs reconstruction is not widely covered in the literature, with only a few papers addressing issues related to roof overhangs. The difference between footprints and roofprints is often not acknowledged, with the term outlines often used interchangeably, or even roofprints being called footprints.

On the other hand, the extraction of building outlines from ALS data has been intensively studied, with the output being most of the time roofprints. Many methods try to extract them directly without any other input. Among these methods, some use variations of famous geometric algorithms such as the Hough transform (Widyaningrum et al., 2019), the medial axis transform (Widyaningrum et al., 2020), the alpha-shape (Liu et al., 2024), triangulations (Awrangjeb, 2016), or even combinations of those (Albers et al., 2016). More recently, many deep-learning methods have emerged using for example generative adversarial networks (Kong et al., 2022), or U-net architectures with attention modules (Dai et al., 2025). However, most of these methods struggle to produce clean polygons directly and therefore need to end with regularisation steps. This is also often true for methods that use images as an input (in combination or not with ALS data), as going from rasters to clean and accurate polygons is not trivial. Therefore, starting from already clean polygonal outlines to improve them can significantly improve the quality of the final outlines.

Regarding roof overhangs, the most common method to estimate them among the few related articles consists in taking the vertical plane that passes through the existing edge, and sweep it in the perpendicular direction. Depending on what was computed first, the existing edge can be a roofprint edges or a footprint edge. Then, a best-fitting plane is determined among these planes with different criteria, by matching with the available data. In Panday

and Gerke (2012), a correlation score is computed for each plane using a point cloud, and the best result is kept only if it represents a sharp enough peak compared to its neighbours. For each edge of the footprint, Dahlke et al. (2015) computes the median height on segments parallel to the edge using a precise 2.5D Digital Surface Model (DSM) with a resolution between 5 and 20 cm. Then, they use the inflection point of the height variation as the roofprint edge. In Frommholz et al. (2017), the roofprint is projected onto the 5 cm resolution DSM and the zero-crossings of the second-order derivative of height variation are used to estimate the size of the roof overhang.

Other methods are proposed by Goebbels and Pohle-Fröhlich (2023) to extend LoD2.2 models by identifying potential overhang edges from the footprints and computing the size of the overhangs from either oblique images or point clouds. Using obliques images and assuming angles of 45°, they identify the roof overhang in the texture using either edges detection or colour regions, and compute the size of the overhang from this. The method with point clouds extends the roof planes and identifies the inliers with a threshold.

Some interesting machine-learning methods were also proposed to compute building outlines from point clouds and could potentially be used to compute both footprints and roofprints. Girard et al. (2020) uses a machine learning model to compute, for each pixel of a RGB aerial oblique image, a classification of building and building edges, as well as a frame field defining tangents and normals of the buildings. Then a multi-step geometric process is used to construct roofprints as polygons. With this kind of aerial data as an input, reconstructing roofprints is easier than footprints due to half of the façades being occluded. Dai et al. (2025) uses a deep learning model that inputs a point cloud and outputs footprints as a binary raster. The model uses sparse voxel representations for the point cloud and decoder/encoder architectures with a specific 3D attention module. Saadaoui et al. (2025) on the other hand uses an almost full deep learning pipeline to produce roofprints as polygons, getting rid of the constraints of rasterization. A first model identifies building pixels, followed by a residual auto-encoder to regularise the segmentation, and finally a lightweight CNN that extracts building corners that can be used for polygonization.

Registration of building outlines based on point cloud data has already been studied by Boussik et al. (2026). Different kinds of transformations were assessed: rigid transformations combining global translations and rotations, non-rigid transformations with individual displacements of the vertices, and semi-rigid transformations with individual displacements of edges along their normals. Semi-rigid transformations of the polygons gave the best results in their case studies and guided us towards this choice. However, we introduced a different approach to clustering the edges to prevent self-intersections. In their method, they cluster the edges beforehand based on angles to prevent self-intersections for neighbour edges which are almost parallel. We softened the constraint by clustering dynamically when displacements break the validity of the polygon, therefore allowing for more precise modifications of the polygons in cases of rounded outlines. As illustrated by Oosterom et al. (2005), the validity of polygons is a very complex topic, but in this paper, we focus solely on preventing rings from self-intersecting, and assume that the method can be extended with more complex validity checks if necessary.

Regarding reference data, to the best of our knowledge, there is a lack of datasets featuring ALS data combined with both footprints and roofprints, making it difficult to evaluate the results of the methods that we propose. The only mention of a similar dataset that we found is Dai et al. (2025) stating that they will release a dataset with more than 3000 building footprints based on the ALS dataset called DALES (Varney et al., 2020). However, it is not yet available at the time of writing this paper.

3. Methodology

Our method, shown in Figure 3, aims at generating two 2D polygons for each building — one for the roofprint and one for the footprint — from two main inputs: ALS data and an initial building outline. Due to the nature of ALS data, the density of points on roof surfaces is significantly higher than on façades, which is the reason why we first focus on the estimation of roofprints and then footprints.

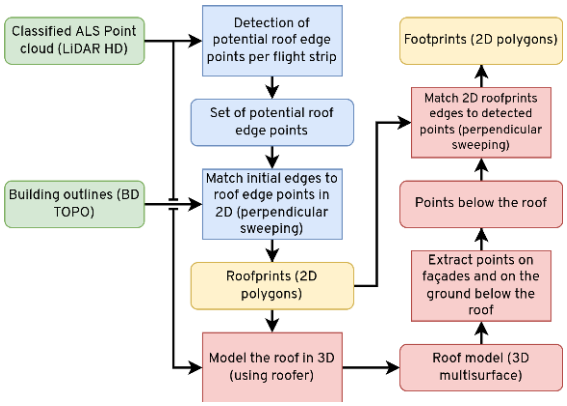


Figure 3. Overview of the proposed pipeline for generating building roofprints and footprints.

The pipeline can be summarised as follows:

1. identify points corresponding to roof edges,
2. use these points to deform the initial outline into a roofprint,
3. identify points corresponding to façades and ground below the roof,
4. use these points to deform the roofprint into a footprint.

3.1 Input data

Our method has only been tested with a high-density ALS point cloud as the first input (about 10 points/m²) so the minimum density for accurate results is unknown. The point cloud should be semantically segmented and contain at least separate classes for vegetation and ground, while the rest is considered as potential building points. A better input would also contain specific and reliable classes for the buildings, and even potentially a separation between roof and façade points. But building annotations often follow different conventions (sometimes roof only, sometimes the façades as well), and the separation between the two is rare, as it requires more complex 3D annotations. Therefore, we aimed for our method to have the least requirements possible on classification.

The point cloud should also contain enough information to isolate the flight strips and order the points in acquisition order. A GPS time attached to each point is sufficient for this purpose, and an ID specific to each flight strip can simplify the process. GPS time can be used because every return coming from the same pulse gets the same exact GPS time, and the GPS time is different for every pulse. To isolate flight strips, one can use their specific IDs if they are available, but it is also possible to isolate them based on their GPS times, as there will be very large gaps between the flight strips. Finally, the trajectory of the sensor is also necessary, but if

it is missing it can be computed automatically using for example the multi-echo pulses of the point cloud (Wu et al., 2026).

The second input consists of building outlines. These can be either footprints, roofprints, or a mix of both. They are expected to be roughly the correct size and roughly in the right location up to a few metres. However their main characteristic is that the directions of their edges and their connections to the other polygons (topology) are assumed to be perfect and will not be modified. These two requirements come from the constraints imposed on the optimisation process, as explained in Section 3.3.

Our method expects the point cloud to be precisely and accurately georeferenced, contrarily to the outlines which are only expected to have an approximate position because they might come from another acquisition method such as imagery or field work. Figure 4 presents examples of the expected relations between the ALS data and the initial outlines.

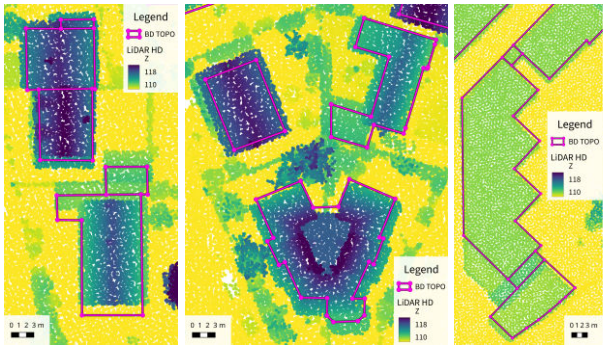


Figure 4. A few examples of inputs with the expected strengths and weaknesses: great shape, approximate positioning and size. Buildings from the BD TOPO located in Ozoir-la-Ferrière.

3.2 Identification of roofprint and footprint location evidence

To compute the roofprints and footprints, a crucial part of the process is to extract evidences of their location. In both cases, the deformation of the polygon are conducted in 2D by dropping the vertical component of the points, but the process of selecting the points is inherently 3D. After selecting the points, the process to produce roofprints and footprints is explained in Section 3.3.

3.2.1 Roofprint evidence: To select points of interest for the roofprint, we look for points at the edges of the roof. We use two properties of ALS point clouds:

1. the angle between the pulses and the vertical direction is small,
2. neighbour pulses in the same scan line or in consecutive scan lines can be identified and are very close.

Together, these two properties imply that when the scanner hits the edge of the roof, one of the two situations is very likely:

- the pulse hits first the roof and then either the ground or the façade at a lower vertical position, or
- the pulse hits only the roof and the next pulse hits either the façade or the ground at a lower vertical position.

These different scenarios are illustrated in Figure 2. In both cases, the roof edge is therefore characterised by a high vertical gap between points from the same pulse or from neighbouring pulses. Each pulse can have up to 4 neighbouring pulses: the previous and next pulses in the same scan line, and the closest pulse in the previous and next scan lines. The vertical gap is defined for each echo, but differently whether the echo is part of a single-echo or a multi-echo pulse. For single-echo pulses, it is the maximum vertical distance between the echo and all the other echoes of the neighbouring pulses. For multi-echo pulses, it is the maximum vertical distance between the echo and all the other echoes of the same pulse. The result on some scan lines of the LiDAR HD is shown in Figure 5.

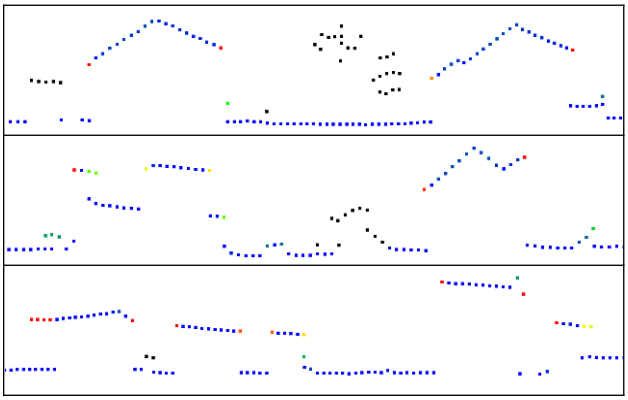


Figure 5. Examples of the vertical gap applied to a few scan lines. Black points are vegetation points, and for the others the colour corresponds to the vertical

This property can be used by setting a threshold defining the minimum vertical gap γ_r to be considered a point on a roof edge. However, there are three main types of points which would also get a high value with this criterion. The first one are vegetation points, and more precisely points coming from sparse vegetation. Sparse vegetation often results in many returns including a return for the ground, and therefore all the points above the ground could get a high vertical gap. The second one are surfaces such as glass which create a return but also let the pulse through. Elements with glass roof such as greenhouses or conservatories result in pulses very similar to roof edges, with one return on the roof and one on the ground. The third one are vertical façades, which can get several points below each other at different heights.

3.2.2 Footprint evidence: Points useful for the footprints are sparser and more difficult to identify than points on the roof. Therefore, we look for these points only once the roofprint has already been computed to use it as a guide.

Since we assume that the façade can only be inside of the roofprint, our method uses a roof reconstruction algorithm to build the roof in 3D. We use roofer[1] (Pađen et al., 2024) in our pipeline, but other algorithms could be used as long as they allow to produce 3D roofs that match with the 2D roofprints. Once the roof is built, all the points directly below it are selected to match the footprint on it. This mostly includes four types of points (see Figure 2):

- points on the façades which can be used to directly estimate the position on the façade,
- points on the ground outside of the building which indicate that the façade is likely behind them,
- points on a lower part of the roof, which can be used similarly to ground points,
- points inside of the building which are rather misleading for our method.

3.3 Polygon deformation for roofprints and footprints

To be able to transform the initial outlines into accurate roofprints and footprints, we need a deformation algorithm flexible enough to handle two different errors at the same time:

[1] https://github.com/3DBAG/roofer

1. a global shift of the polygon in one direction to account for imprecise georeferencing,
2. individual displacements of edges to account for size and shape differences between footprints and roofprints.

On the other hand, based on the previous work of Boussik et al. (2026), some constraints are necessary to reduce the complexity of the search and to preserve the quality of the initial outlines. The three main constraints are the following:

1. edges cannot be rotated, as their angles are assumed to be accurate,
2. polygons should keep their shape and edges cannot be flipped (see Figure 6) but can be reduced to a single point,
3. adjacencies between buildings should be preserved as much as possible, in our case by ensuring that edges shared between different building remain collinear.

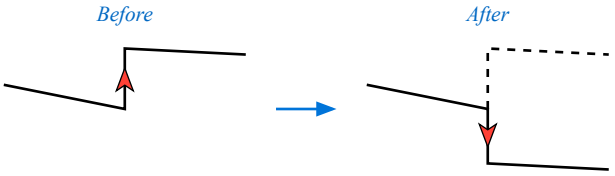


Figure 6. Illustration an edge being flipped by the translation of a neighbour edge.

We therefore define the polygon deformation problem as an optimisation problem, combining two parts. We present in Section 3.3.1 an iterative algorithm to incrementally solve this problem. The first element of this algorithm is a method to produce new propositions of polygons which satisfy the aforementioned constraints, explained in Section 3.3.2. The second element is an energy that the optimal polygon should minimise, which is based on the input point cloud. This energy is completely different for the roofprints and the footprints, as presented respectively in Section 3.3.3 and Section 3.3.4.

3.3.1 Matching algorithm: The general idea behind the matching algorithm is simple. For each group of lines:

1. a set of potential shifts is defined,
2. for each shift, the final configuration of the polygon is computed with the polygon deformation algorithm described in Section 3.3.2,
3. the best configuration is identified with a criterion and replaces the previous one,
4. a new group of lines is picked.

A pseudocode implementation of the algorithm is shown in Algorithm 1. This algorithm takes as input all the groups of lines G and all the points P that the polygons must be deformed on. Groups of lines contain the lines that correspond to shared edges: edges from different polygons that are aligned and intersect. This is explained further in Section 3.3.2.

It must be noted that the order in which the groups of edges are processed can lead to significantly different results, as shifting one group can lead to moving many other groups, and this temporary result will be the starting point for the next group. However, it is unclear which ordering strategy is the best. Two simple strategies however emerged with their own advantages:

- random order allows next iterations of the algorithm to potentially fix biases introduced by the previous iteration,
- ordering groups by the length of the edges in decreasing order makes the algorithm start with the easiest edges, as long edges are less affected than small edges by the same absolute shift in the input data.

function matching_algorithm(G, P)
  ▷ *The order to iterate over G matters*
  for g_0 in G **do**
    $S \leftarrow$ a list of potential shifts perpendicular to the lines in G
    ▷ *Compute the deformed polygons and their energy*
    shifted_groups $\leftarrow$ empty dictionary
    energies $\leftarrow$ empty dictionary
    for $\vec{s}$ in S **do**
      shifted_groups$[\vec{s}] \leftarrow$ **deformation_one_edge**$(g_0, \vec{s})$
      energies$[\vec{s}] \leftarrow$ **compute_energy**(shifted_groups$[\vec{s}]$, $\vec{s}$, P)
      ▷ ***deformation_one_edge*** *is defined in Algorithm 2 and* ***compute_energy*** *is defined in Section 3.3.3 and Section 3.3.4*
    end
    ▷ *Select the best shift and apply it to the relevant lines*
    $\vec{s}_{\max} \leftarrow$ shift $\vec{s}$ with the lowest energy
    for g in shifted_groups$[\vec{s}_{\max}]$ **do**
      Shift all the lines in g by $\vec{s}_{\max}$
    end
  end
end

Algorithm 1. Polygon matching algorithm. This represents one iteration, which consecutively focuses on each group of lines to find its optimal shift.

This blueprint algorithm is then adapted to produce roofprints or footprints, because they use different evidences, with different distributions and different objectives, so the energy depends on the application.

3.3.2 Deformation algorithm: To satisfy all the properties listed at the beginning of Section 3.3, our algorithm is designed as an iterative process where each iteration focuses on moving a specific edge along its normal while allowing the displacement of this edge to influence the whole polygon in order to avoid self-intersections and edge flips. Let's assume that we have a polygon with n vertices defined as a set of consecutive points $\{p_i, i \in [\![0, n-1]\!]\}$. For the rest of the paper, we assume that the indices of the objects are modulo n, meaning that p_n refers to p_0. We consider the polygon as a set of oriented lines $\{l_i, i \in [\![0, n-1]\!]\}$, where l_i is the infinite line passing through p_i and p_{i+1} and oriented from p_i to p_{i+1}. The direction d_i is the unit direction vector of l_i. Figure 7 illustrates all these definitions.

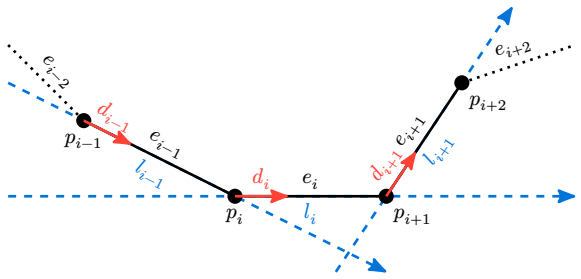


Figure 7. Illustration of the mathematical objects for a few edges of a polygon.

With these considerations, if we assume that two consecutive lines are never parallel, the point p_i can be retrieved as the intersection between l_{i-1} and l_i, meaning that the lines contain the full definition of the polygon. These lines are however more convenient to handle compared to the points, as they can be translated freely along their normals without breaking the constraints.

With this, the property that an edge e_i should not be flipped can still be checked relatively easily by intersecting l_i with both l_{i-1} and l_{i+1} and comparing the edge obtained with d_i. Let's call this property

$\mathcal{P}_1(l_i)$: *the edge defined by l_{i-1}, l_i and l_{i+1} is not flipped.*

This property is enough in practice in most cases to ensure that the final polygon will be valid, because it prevents self-intersections between l_{i-1} and l_{i+1}. However, to prevent all potential self-intersections, it is necessary to check for intersections between each shifted edge and all other edges, which is much more computationally extensive. Let's call this new property

$\mathcal{P}_2(l_i)$: *the edge defined by l_{i-1}, l_i and l_{i+1} does not intersect any other edge in the polygon.*

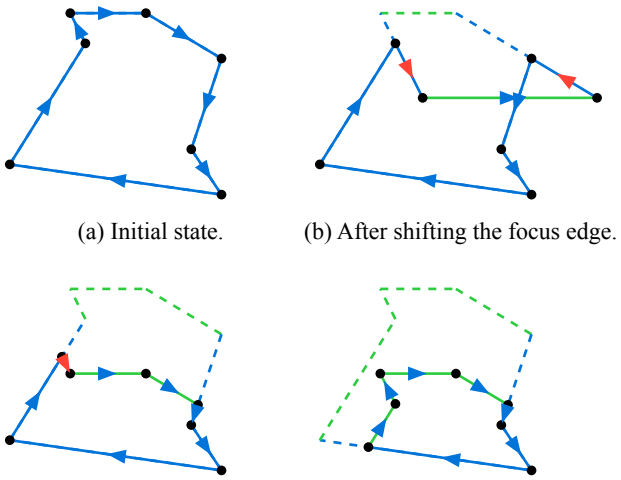
These two properties are enough to create valid polygons that satisfy the first and second constraints mentioned in the introduction of Section 3.3. The last constraint is to keep edges shared by adjacent polygons collinear. This is accomplished by grouping together the lines corresponding to these edges and shifting them by same amounts in the same directions.

The proposed algorithm is detailed in Algorithm 2, taking as input a line group to shift g_0 and a shift to apply $\vec{s}$. It is also illustrated in Figure 8 Its main idea is to propagate the shift through the connections between groups (both connections in polygons and between polygons), as long as $\mathcal{P}_1$ or $\mathcal{P}_2$ is not verified. Here are descriptions for the functions used in this pseudocode:

- get_group(l): returns the group of overlapping lines from different polygons which contains l,

```
1:  function deformation_one_edge(g_0, s⃗)
2:      shifted_groups ← {}
3:      groups_to_process ← {g_0}
4:
5:      while groups_to_process is not empty do
6:          g ← pop(groups_to_process) ▷ Order does not matter
7:          if g ∉ shifted_groups then
8:              to_shift ← False
9:
10:             ▷ Decide whether to shift the group g
11:             for l in g do
12:                 if not check_p1(s⃗, shifted_groups, l) or not
                    check_p2(s⃗, shifted_groups, l) then
13:                     to_shift ← True
14:                     break
15:                 end
16:             end
17:
18:             ▷ If necessary, shift the group g and add its neigh-
                bours
19:             if to_shift then
20:                 push(shifted_groups, g)
21:                 for l_i in g do
22:                     g_prev ← get_group(l_{i-1})
23:                     g_next ← get_group(l_{i+1})
24:                     push(groups_to_process, (g_prev, g_next))
25:                 end
26:             end
27:         end
28:     end
29:     return shifted_groups
30: end
```

Algorithm 2. Polygon deformation algorithm, responsible for computing the new configuration of all the adjacent polygons after one group of shared edges (g_0) was shifted (by $\vec{s}$).



(a) Initial state. (b) After shifting the focus edge.

(c) First step of the resolution. (d) Second and final step of the resolution.

Figure 8. Illustration of the polygon deformation algorithm on an isolated polygon. Flipped edges are indicated with red arrows, and shifted edges are green.

- check_p1($\vec{s}$, shifted_groups, l): returns a boolean stating whether the property $\mathcal{P}_1(l)$ is verified for the current polygon if the groups in shifted_groups are shifted by $\vec{s}$.
- check_p2($\vec{s}$, shifted_groups, l): same as check_p1($\vec{s}$, shifted_groups, l) but for $\mathcal{P}_2(l)$.

The pseudocode in Algorithm 2 is not optimised for speed but rather simplified for clarity. For a given line group g, we use its perpendicular unit vector $\vec{n}$ as the direction for the shift. Along this direction, we define a set of shifts to try in both directions, $\{kt\vec{n}, k \in [\![-m, m]\!]\}$, with user-defined values for the step size t and maximum shift length ms. A full iteration of the algorithm in Algorithm 1 calls this function once for every pair of group g and shift $\vec{s}$. Moreover, if computing the resulting deformed polygon for increasing shifts such as $\{kt\vec{n}, k \in [\![1, m]\!]\}$, the result obtained for $kt\vec{n}$ can be used as a starting point for $(k+1)t\vec{n}$, therefore increasing the computing speed. On a side note, the order in which the groups are processed in groups_to_process does not matter: the algorithm will always terminate and return the same results.

3.3.3 Particularities for the roofprints: To compute the roofprint, the algorithm given in Section 3.3.1 is adapted to use the points identified as explained in Section 3.2.1. The energy presented below is based on careful analyses of portions of BD TOPO and LiDAR HD, as it is the central part of the optimisation. We use our properties which we expect to find in most ALS point clouds with similar or higher density.

First, to prevent lines from matching with points that correspond to neighbouring building, we define what we call the inward direction of a point. The goal of the inward direction is to be as close as possible to the 2D normal of the corresponding roof edge, oriented towards the inside of the building. To compute it for a given point p, we use the following method:

1. identify the set of points $p_j \in \mathcal{B}(p, \delta)$ that are less than a certain distance δ from p,
2. compute the inward vector v as the average of the vectors unit vectors from p to p_j:

$$v = \frac{1}{|\mathcal{B}(p,\delta)|} \sum_{p_j \in \mathcal{B}(p,\delta)} \frac{p_j - p}{|p_j - p|} \tag{1}$$

The idea behind this formula is that points on the edge of the roof should have in their neighbourhood many points towards the inside of the building, and few points towards the outside. Using unit vectors and averaging them gives all points in the neighbourhood the same weight, resulting in a value that could be interpreted as a proxy of the direction in which the density of points is higher in the neighbourhood. Moreover, the magnitude of the inward vector is an indication of the confidence.

With this inward direction, the score of a point p_i for a line l_j can be defined. It is 0 if any of the following conditions is true:

- the orthogonal projection $p_{i\perp j}$ of p_i on the line l_j is outside of the segment defined by l_{i-1} and l_{i+1}, or
- the distance from p_i to $p_{i\perp j}$ is greater than a certain threshold ε_r, or
- the dot product of the inward vector v_i of p_i with the normal n_j of l_j oriented towards the inside of the building is negative.

Otherwise, the score is:

$$\text{score}(p_i, l_j) = \underbrace{(v_i \cdot n_j)}_{\substack{\text{alignment of} \\ \text{point and edge} \\ \text{'normals'}}} \times \underbrace{\left(1 - \frac{|p_i - p_{i\perp j}|}{\varepsilon_r}\right)}_{\text{proximity to the edge}} \geq 0 \quad (2)$$

We also associate an intrinsic weight to every point in the point cloud to give more value to the most promising points. The weight w_i is a product of two factors between 0 and 1:

- one giving higher values to points coming from multi-echo pulses because they should give a more accurate position for the roof edge,
- one giving higher values to points classified as building if the input point cloud has this classification.

Finally, the energy to minimise can be defined as:

$$E = \underbrace{-\sum_{i \in P} w_i \max_{j \in \mathcal{L}} \{\text{score}(p_i, l_j)\}}_{\text{proximity to the points}} + \alpha \underbrace{\sum_{j \in \mathcal{L}} \left(|l_j| - |l_j^0|\right)^2}_{\text{similarity to the initial edges}} \quad (3)$$

where:

- P is the set of points, and $\mathcal{L}$ is the set of lines,
- $|l_j|$ and $|l_j^0|$ are the length of edge l_j in the current and initial configurations respectively,
- α is a parameter to adjust between the two terms.

In this energy, 5 main parameters need to be tuned: δ for the inward direction, ε_r for the score of a point on a line, α to balance the proximity and regularisation terms in the energy, and two parameters for the intrinsic weights of points.

3.3.4 Particularities for the footprints: To compute the footprints, we also need to adapt the algorithm, but this time for different reasons.

First, a constraint specific to footprints is their respective roofprints to cover them. It is tempting to try to enforce this by allowing the polygon deformation algorithm to only shift edges towards the inside of the polygon. However, this would create two problems. Firstly, if two edges make an angle smaller than 90°, their inward directions are opposite, meaning that the first one could limit the movements of the second one. Secondly, if the movement of one edge brings another edge too far towards the inside, this last edge will not be able to go back to a better position. For these two reasons, trying to enforce the inclusion in the algorithm is not trivial and would require deeper changes. We therefore decided to enforce the inclusion at a later stage as post-processing by intersecting the footprint with the roofprint.

Then, the energy for the footprints must cater to the particularities of its relevant evidences, such as the high variance in the amount of pulses that reached the façades. As illustrated in Figure 9, the amount of such points depends on many variables for which small variations can have a significant impact:

- the height of the building,
- the size of the roof overhang,
- the geometry of the lidar scanning device,
- the alignment between the flight axes and the façades.

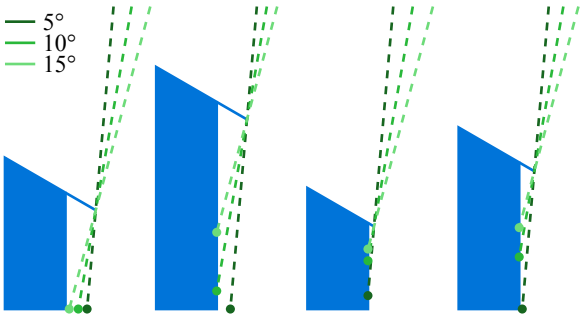


Figure 9. Illustration in profile view of the highest possible hit on the façade depending on the building and the angle of the pulse.

However, even when there is no point on a façade, a pulse can still hit the ground under the roof. In most cases, these hits happen outside of the building. Therefore, the corresponding points can be used to push the footprint edges further towards the inside. But if the façade is made of transparent material, as happens with windows for example, the pulses can also hit the inside of the building. Often, in this case, the pulse returns two hits: one for the façade and one on the floor of the building. The first one is very useful because it indicates the position of the façade, while the second one is misleading because in our case we assume that ground points are outside of the building.

The energy that we use for footprints tries to combine the points on the façades and the points on the ground, by looking for:

- a concentration of points close to the segment,
- as little points as possible behind the façade.

We use the energy illustrated in Figure 10. It is a combination of two terms: a proximity term and a penalty term. The proximity term counts the points close to the edge higher absolute values when they are closer to the edge, with a maximum value of 1 for points on the edge and 0 for points further than a certain distance ε_f from the edge. The penalty term counts the points further towards the inside of the building. To achieve this, we use a signed distance defined as $d_s(p, e) = (p - p_{\perp e}) \cdot e_\perp$ where $p_{\perp e}$ is the projection of p on e and $e_\perp$ is the unit vector perpendicular to e pointing towards the inside of the building. This distance is therefore positive if the point is towards the inside of the building and negative otherwise. We use this distance to add a penalty which goes from 0 when $d_s(p, e) = \varepsilon_f$ to a maximum value of 1 when $d_s(p, e) = \gamma_f$. ε_f and γ_f are parameters to tune based on the density and noise of the point cloud. On top of that, based on experiments and qualitative observations, the values of the energy are scaled differently for ground points. The intuition behind this is the following. For clustering, points on the façades are the most important. For the penalty, we want to prioritise a high density of points instead of having a more inward candidate to win simply with the penalty.

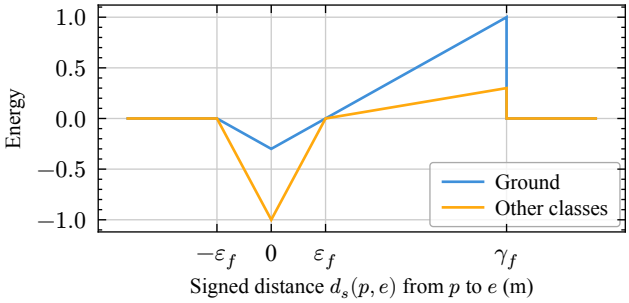


Figure 10. Energy of a point p for a footprint edge e.

4. Experiments with real-world data and discussion

4.1 Validation dataset

To assess and validate the method, we manually created a validation dataset with accurate and precise roofprints. Our main objectives with the creation of this dataset are as follows:

1. it should allow for one-to-one comparisons on individual buildings with the BD TOPO: the identifiers of the new dataset are the ones of the BD TOPO and hopefully correspond to the same actual buildings,
2. the roofprints should be aligned on the LiDAR HD dataset, which is assumed to be the most precise and accurate data source available,
3. the dataset should not be specific to our method, but rather contain precise and accurate roofprints independently from their shape and position in the BD TOPO.

In practice, we used as much information as necessary to identify correct positions for vertices and edges. The positions of the points in the LiDAR HD was the central criterion, but we also used the intensity when necessary as it sometimes allow to differentiate between different adjacent roofs. Annotations were conducted in 2D using QGIS, but a 3D viewer was also used to get a better understanding of the point cloud. When something was unclear, aerial photos or street view photos were also useful.

Regarding the actual shapes of the roofprints, a few decisions were taken to create accurate and coherent roofprints. The version of the BD TOPO that we use as the basis is the current latest one (2026-03-15). First, even if the roofprints are based on the BD TOPO, we have a complete freedom to modify the polygons: vertices can be freely added, moved or removed. Changing the topology and the relations between the different buildings was also allowed. One example is the creation or deletion of an inner ring representing a hole in the building. Another example is adding a space between buildings that are adjacent in the BD TOPO. Moreover, getting approximately right angles was sometimes prioritised over following precisely the points from the LiDAR HD, since the alignment of points in ALS point clouds is biased by the trajectory of the scan. This was adapted individually to every building, with aerial photos helping to identify right angles.

However, we did not create a dataset for footprints. The main reason for this is the lack of time which meant having to prioritise some elements in the dataset compared to others. Multiple reasons lead us to dismissing footprints in the dataset to the benefit of roofprints:

- qualitatively, the method seemed more robust and accurate for roofprints than for footprints,
- a roofprint dataset with diverse buildings seemed more interesting than two smaller datasets for roofprints and footprints,
- approximately 88.5% of the BD TOPO outlines are already footprints.

We then picked three areas with different types of buildings, to test the method in different situations. We split the buildings in four different categories (see Figure 11):

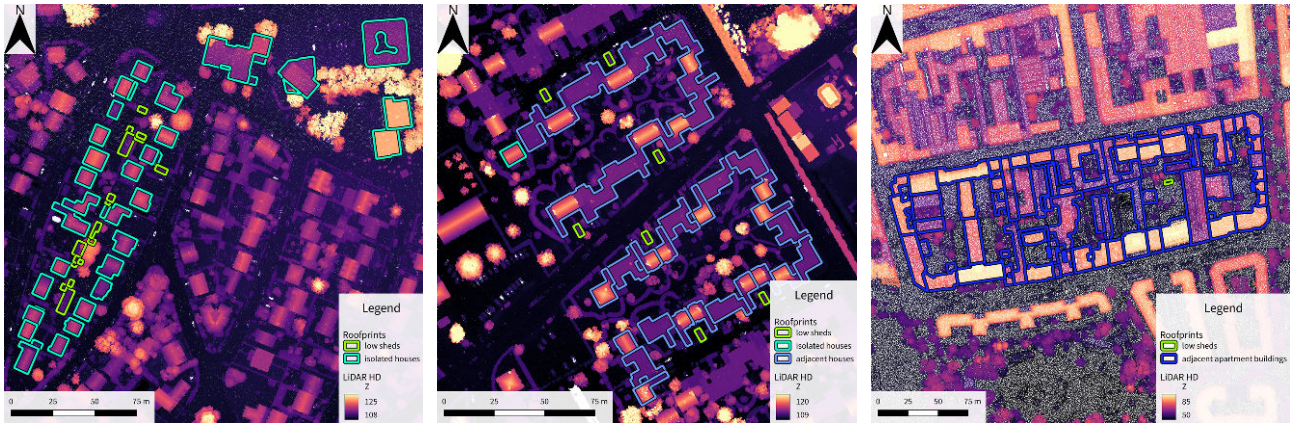
- 40 “isolated houses”: medium houses and buildings having up to a few storeys, isolated from the buildings around,
- 25 “adjacent houses”: blocks of adjacent and medium houses and buildings having up to a few storeys,
- 25 “low sheds”: low buildings in height, usually in gardens,
- 80 “adjacent blocks of flats”: blocks of adjacent and high buildings.

Note that these numbers are the exact numbers of roofprints, and buildings were split exactly as in the BD TOPO. This means that one roofprints may contain many building units, or one building unit may be split in multiple roofprints. For example, the “adjacent houses” contains two blocks, one of them being split in individual houses while the other one is not (as shown in Figure 11b). Therefore the amount of building units is likely closer to 45.

4.2 Results

We use three different metrics to evaluate a given polygon $\mathcal{P}$ compared to the ground-truth polygon $\mathcal{Q}$:

1. The intersection over union (IoU) defined as:



(a) Area in Ozoir-la-Ferrière with isolated houses and small sheds.

(b) Area in Ozoir-la-Ferrière with mainly adjacent houses.

(c) Area in Paris with mainly adjacent blocks of flats.

Figure 11. Validation dataset.

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.735	0.604	0.939
Rfpt 1 iter	0.842	**0.331**	**0.541**
Rfpt 2 iter	0.851	0.335	0.548
Rfpt 3 iter	**0.853**	0.342	0.566

(a) Whole dataset.

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.776	0.557	0.846
Rfpt 1 iter	0.89	0.243	0.388
Rfpt 2 iter	0.902	0.226	0.362
Rfpt 3 iter	**0.905**	**0.22**	**0.354**

(b) Whole dataset except the "low sheds" category.

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.497	0.876	1.477
Rfpt 1 iter	**0.562**	**0.842**	**1.426**
Rfpt 2 iter	0.557	0.97	1.631
Rfpt 3 iter	0.549	1.045	1.793

(c) Category "low sheds".

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.744	0.55	0.681
Rfpt 1 iter	0.888	0.165	0.215
Rfpt 2 iter	**0.935**	**0.114**	0.12
Rfpt 3 iter	**0.935**	**0.114**	**0.117**

(d) Category "isolated houses".

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.797	0.448	0.638
Rfpt 1 iter	0.901	0.196	0.309
Rfpt 2 iter	0.91	0.175	0.269
Rfpt 3 iter	**0.914**	**0.168**	**0.259**

(e) Category "adjacent houses".

Dataset	IoU	Chamfer (m)	Centroid (m)
BD TOPO	0.785	0.595	0.994
Rfpt 1 iter	**0.888**	0.297	**0.499**
Rfpt 2 iter	0.882	0.298	0.512
Rfpt 3 iter	0.887	**0.289**	0.503

(f) Category "adjacent blocks of flats".

Figure 12. Average metrics over different subsets of the validation dataset.

$$\mathrm{IoU}(\mathcal{P}, \mathcal{Q}) = \frac{\mathrm{area}(\mathcal{P} \cap \mathcal{Q})}{\mathrm{area}(\mathcal{P} \cup \mathcal{Q})} \tag{4}$$

The value is between 0 and 1 and higher values are better.

2. The Chamfer distance. Points are sampled on the two polygons with a step size δ, resulting in two point clouds P and Q. The distance is then defined as:

$$\mathrm{Chamfer}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2}\left(\sum_{p \in P} \min_{q \in Q} |p - q| + \sum_{q \in Q} \min_{p \in P} |q - p|\right) \tag{5}$$

3. The centroid distance, defined as the distance between the centroids of the two polygons.

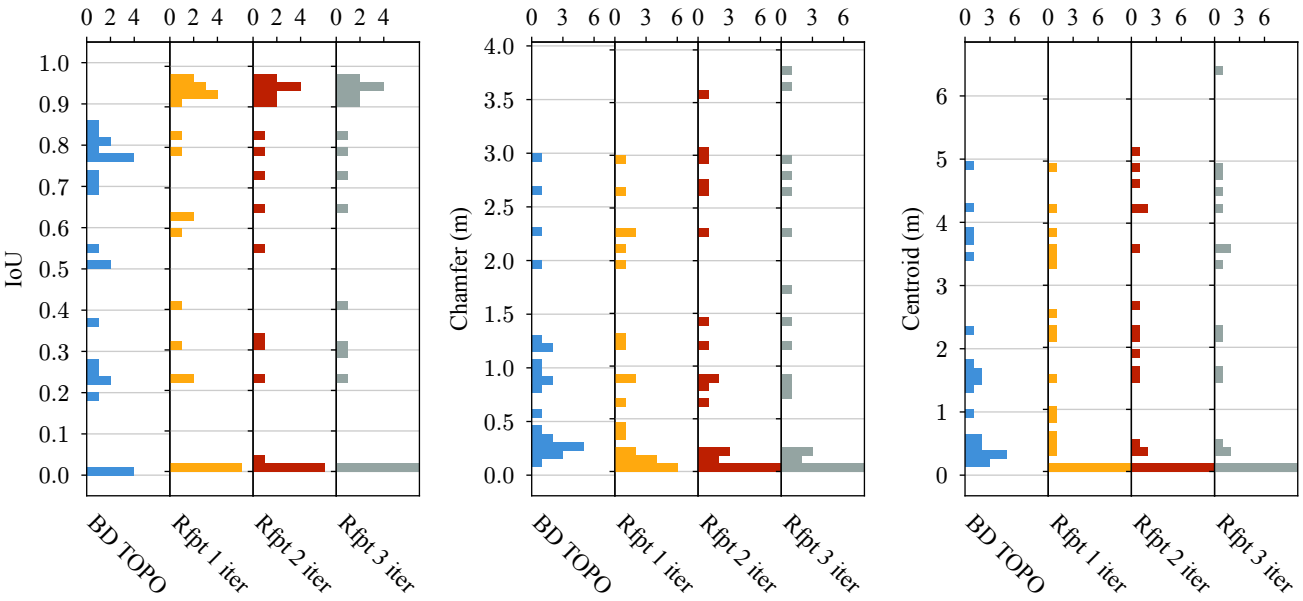
Overall, the different metrics show that our method reconstructs roofprints that really improve the initial outlines from the BD TOPO. It has to be noted however that most of the outlines in the validation dataset correspond to footprints in the BD TOPO, so comparing them to the hand-annotated roofprints puts them at a disadvantage.

In all the results shown below, we use the abbreviation "Rfpt N iter" for the roofprints resulting of N iterations of the polygon matching algorithm (Algorithm 1). As displayed in Figure 12a, all the metrics are significantly improved by our method on average for all the buildings.

But to see things clearer, the sub-results for the different categories must be considered. What this shows is that the second and third iterations only worsen the results for "low sheds" (see Figure 12c), while improving them or keeping them constant in the other categories. Moreover, the final metrics are much worse for the "low sheds" category compared to the others, which comes from the combination of multiple factors. First, buildings in this category usually have smaller areas, meaning that the same shift will result in lower IoU values. Then, their initial position is significantly worse than for the other categories, as can be seen with the high values of Centroid distance for the BD TOPO. Finally, we used for this experiment $\gamma_r = 2$ m (see Section 3.2.1) therefore targeting buildings which roof edges were at at least 2 metres above the ground, which is not always the case in this category.

To get a better understanding of what actually happens, we display the results in two different ways. First, Figure 13 shows the distributions of these metrics at every iteration of the algorithm. Then, Figure 14 shows the evolution of the metric for every single building in the "low sheds" category. This shows that most of the buildings for which it was initially correct get better, while for some of them there is no change, which is often explained by the algorithm having identified no useful point in their neighbourhood. It only gets really worse for 5 out of the 25 in that category.

A large subset of the actual roofprints are displayed in Figure 15, to compare the evolution between the BD TOPO and of the Rfpt 3 iter, on top of the ground-truth polygons. This shows that the method works well in most cases, and gives insights on the reasons why it fails in some cases. Figure 15a and Figure 15b show the different behaviours for the sheds depending on their surroundings. When there is a large building close by, the polygon



(a) IoU. (b) Chamfer distance. (c) Centroid distance.

Figure 13. Distribution of the metrics for the polygons in "low sheds".

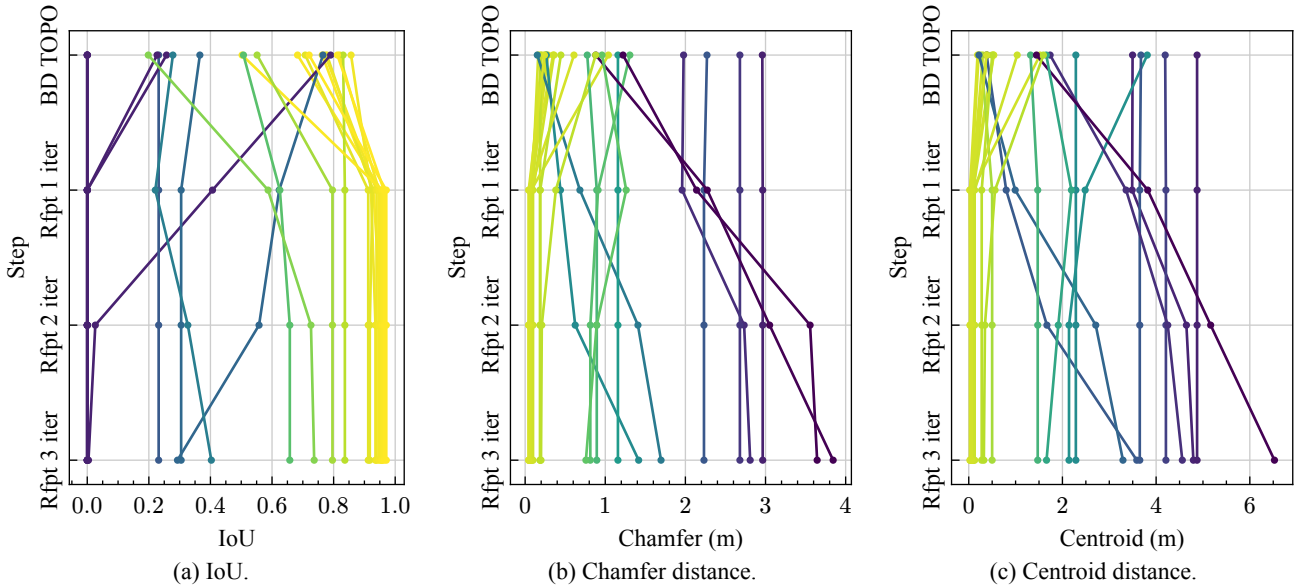
ends up fitting to this building instead of the shed, whereas when the neighbourhood is empty of higher points, the polygon stays in position.

4.3 Discussion

The strengths of the method come from its ability to combine the strengths of the two data sources: the well estimated façade orientations and topology of the initial outlines and the positioning accuracy of the ALS point cloud. Thanks to its ability to not only shift the polygons but also deform them, the method can take any outline as input, as long as it has the characteristics mentioned in Section 3.1. The method is robust to outliers and to partial occlusions thanks to considering the outlines globally and not only individual edges.

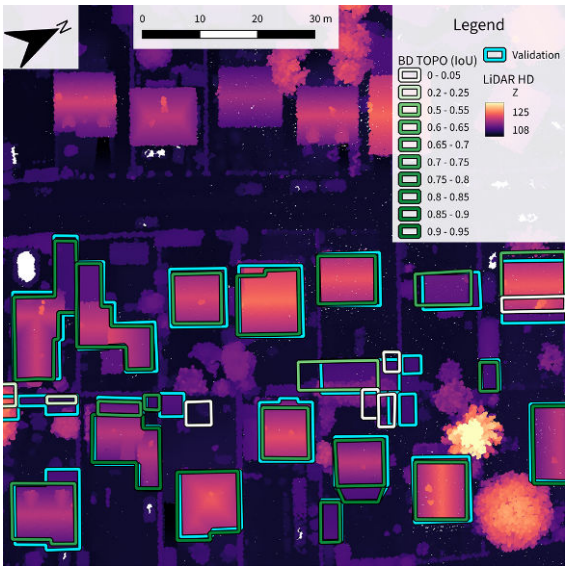
However, relying on the angles of the initial outlines is not always desirable. Moreover, our assumption is that the method can produce accurate roofprints even given footprints as an input, because we observed in practice that they are most of the time less complex. However, it also happens sometimes that roofs have details that are not present at the bottom of the buildings and therefore the perfect roofprint is more complex than the initial footprint. It is for example the case when a building has the same exact balcony at every storey, resulting in an extrusion of the façade that could also be part of the roofprint. In this case as well as many other cases where the initial outline simply lacks some details, it would be necessary to complexify the initial outline to get more accurate results.

The classification of the point cloud is also a crucial aspect of the input. Our method requires points corresponding to high vegetation to be dismissed, because these points will otherwise be identified as points of the roof edges. Since the method completely removes these points to compute the roofprint, it can lead to inaccurate results if a whole side of a building is classified as vegetation. Regarding the classification of ground points for the

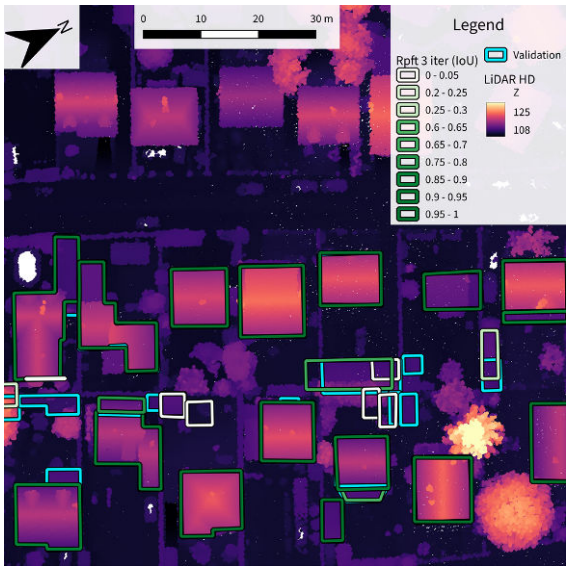


(a) IoU. (b) Chamfer distance. (c) Centroid distance.

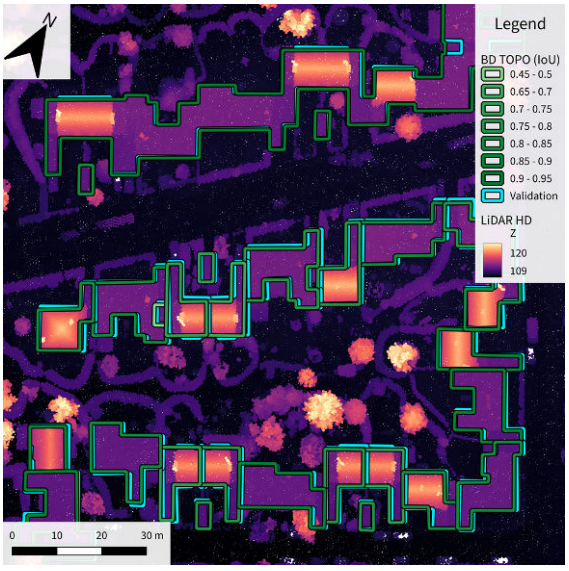
Figure 14. Distribution of the metrics for the polygons in "low sheds".



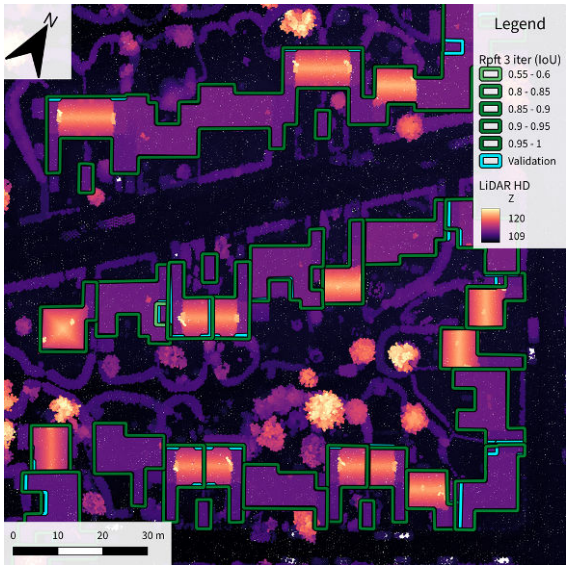
(a) Validation and BD TOPO.



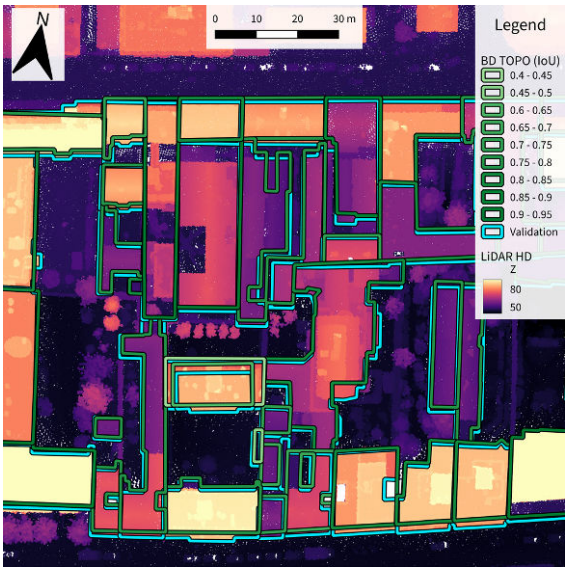
(b) Validation and Rfpt 3 iter.



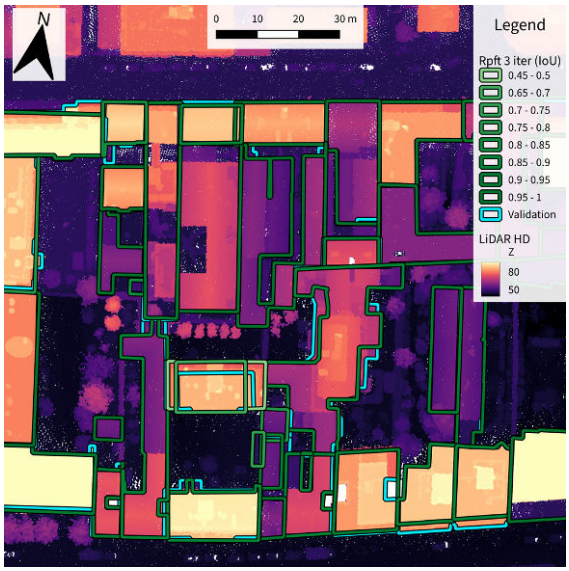
(c) Validation and BD TOPO.



(d) Validation and Rfpt 3 iter.



(e) Validation and BD TOPO.



(f) Validation and Rfpt 3 iter.

Figure 15. Visualisation of the results in different parts of the dataset. The polygons of the assessed datasets are coloured based on their IoU values compared to the ground-truth polygons.

footprints, it is a less crucial requirement for two reasons. First, ground points are usually well classified, with geometric methods such as CSF by Zhang et al. (2016) already producing great results. Then, the proposed metric used for footprints still looks improvable, and it may be possible to get similar or even better results without relying on this separation.

On top of that, the method has not really been tested yet with lidar obtained with different ALS sensor geometries. The results are expected to be similar for roofprints, but potentially very different for footprints, as the number of points on the façades and on the ground below the roof depends heavily on the scan angle.

Finally, the method has a significant number of parameters to tune. These parameters are related to different parts of the method:

- the extraction of the points on the roof edges,
- the amount of shift to try to apply to the edges for roofprints and for footprints,
- the metrics used to score the roofprints and the footprints.

Many of these parameters have interpretations that can help setting them to correct defaults by computing properties over the point cloud and by having information about the buildings to identify, such as the point cloud density or the height of the buildings. But some of them are mathematical objects that are more complex to optimise, such as the weighting of the different terms in the metrics, which can be interpreted as balancing the quality of the point cloud and the initial outlines, and requires more human input and knowledge about the datasets.

5. Conclusion and future work

5.1 Conclusion

Our method is innovative for two reasons. First, we introduced a simple but efficient way to create new valid polygons from existing polygons with our polygon deformation algorithm. It successively leverages the strengths of existing outlines while preserving a lot of freedom in the deformation of the polygons, allowing not only for global translations and scaling but also more complex local deformations. Then, we also introduced new and simple yet effective ways to extract information from ALS point clouds for roofprints and footprints. Our identification of points on roof edges is promising and relying only on having a correct classification of high vegetation. As for the footprints, one of our most important conclusions is the necessity to clearly identify the roof first, because one of the main challenges of producing footprints is to identify points on the façades, and knowing the exact shape of the roof transforms this into a simple geometric operation.

More generally, we also underlined the importance of making correct distinctions between roofprints and footprints and more generally to define precisely what we include or not in our 2D representations of buildings. The complexity and diversity of buildings even sometimes in a single region requires careful definition of what is included or not in their 2D models.

5.2 Future work

There are many elements of the method which could be improved to produce more robust and predictable results.

An interesting property of most buildings that is not used by this method is the symmetry. Panday and Gerke (2012) use it to assume that roof overhangs are the same on opposite sides when the roofs are slanted. This could help computing better estimations by increasing the amount of data that can be used, either by averaging the results on both sides, or by making a reasonable guess if a side is empty based on the other side. Symmetry is also applicable in many cases to other aspects of buildings such as the slopes of the roofs or the directions of the roof edges.

Improvements that would improve the results are related to the polygon deformation algorithm. This still lacks three key properties to be fully satisfying: continuity, preservation of shared vertices and relations between disconnected polygons. First, discontinuity comes from having a binary decision on shifting or not each edge, instead of gradually shifting edges as little as possible when the polygon would become invalid. Secondly, vertices shared by two polygons are currently lost when their respective edges in each polygon are not shared. Solving this would imply adding new constraints to the deformation algorithm, as when a vertex is shared by three or more edges, moving one edge requires to move the two other edges by the exact same shift. Finally, nothing currently prevents two polygons to end up overlapping. This happens for example when a small building moves too far and ends up matching with the points of another larger building. This is however undesirable and could be prevented by considering also the neighbouring buildings during the optimisation.

More complex modifications of the polygon deformation method could involve relaxing some of the constraints that were introduced in the algorithm. For example, one could allow small rotations of the edges or even of the whole outlines if it results in much better scores. Another idea would be to allow the algorithm to insert new edges, for example by extruding a part of an edge. These two ideas come from the observation of actual situations where the BD TOPO could have been improved with one of these two techniques. However, allowing this kind of changes opens the door for very difficult considerations regarding the complexity of the outlines and necessary regularisation.

Another aspect that could significantly improve the results would be better and more reliable ways to classify the point clouds or extract the points of interest. There are two main directions for this. The first one would be to try to improve the way roof edge points are currently identified, to make it more robust to vegetation, better at differentiating façade and roof edges, or even more robust to transparent surfaces. One idea that we had was to identify straight segments corresponding to roof surfaces as in Wu et al. (2016), in order to consider the end of the segments as potential roof edges. The other path would be to focus more generally on ALS point cloud semantic segmentation, in order to classify separately at least roofs, façades and ground, and even potentially other classes such as balconies or stairs. There are a few great datasets which consider these differences in their classification, starting with the oldest: Vaihingen ISPRS by Rottensteiner et al. (2012) DublinCity by Zolanvari et al. (2019), Hessigheim 3D by Kölle et al. (2021) and CENAGIS-ALS by Zachar et al. (2023). These datasets with detailed classification are however difficult to produce, and methods that perform well on the rare classes of façades, balconies or chimneys are still to be developed.

Finally, this whole work revealed limitations in the way building outlines are currently represented and considered. Roof overhangs and balconies often expand outside of the outline of the buildings at ground level. Looking at it in 2D from above, a given position may contain both ground and roof, or balcony and roof, or even all of them at the same time. The roof overhangs of one building can also be above the roof of another building. Therefore, representing all these objects using a strict partition of space in 2D prevents accurate representations of all these elements at the same time. Creating representations that allow for robust and

clear delineation of ground, roofprints, footprints, balconies and potentially other outdoor elements, by allowing adjacent roofprints and footprints to either overlap or share edges depending on the situation is to our knowledge still an open question. Such a representation of space would certainly benefit from integrating some vertical information about the different objects and could then pave the way for the creation of more detailed 3D models of buildings from ALS point clouds.

References

Albers, B., Kada, M., Wichmann, A., 2016. Automatic Extraction and Regularization of Building Outlines from Airborne Lidar Point Clouds. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLI-B3, 555–560. 10.5194/isprs-archives-xli-b3-555-2016.

Awrangjeb, M., 2016. Using point cloud data to identify, trace, and regularize the outlines of buildings. *International Journal of Remote Sensing*, 37(3), 551–579. 10.1080/01431161.2015.1131868.

Bauchet, J.-P., Sulzer, R., Lafarge, F., Tarabalka, Y., 2024. SimpliCity: Reconstructing Buildings with Simple Regularized 3D Models. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, 7616–7626. 10.1109/cvprw63382.2024.00757.

Biljecki, F., Ledoux, H., Stoter, J., 2016. An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, 59, 25–37. 10.1016/j.compenvurbsys.2016.04.005.

Boeters, R., Arroyo Ohori, K., Biljecki, F., Zlatanova, S., 2015. Automatically enhancing CityGML LOD2 models with a corresponding indoor geometry. *International Journal of Geographical Information Science*, 29(12), 2248–2268. 10.1080/13658816.2015.1072201.

Boussik, A., Vallet, B., Duan, L., 2026. A non-rigid polygon Registration Framework and its application to Enhancing Building Footprint Accuracy using aerial LiDAR. *(To appear) Annals of the ISPRS Congress 2026*.

Dahlke, D., Linkiewicz, M., Meissner, H., 2015. True 3D building reconstruction: façade, roof and overhang modelling from oblique and vertical aerial imagery. *International Journal of Image and Data Fusion*, 6(4), 314–329. 10.1080/19479832.2015.1071287.

Dai, H., Xu, J., Hu, X., Shu, Z., Ma, W., Zhao, Z., 2025. Deep projective prediction of building façade footprints from ALS point cloud. *International Journal of Applied Earth Observation and Geoinformation*, 139, 104448. 10.1016/j.jag.2025.104448.

Frommholz, D., Linkiewicz, M., Meissner, H., Dahlke, D., 2017. Reconstructing Buildings with Discontinuities and Roof Overhangs from Oblique Aerial Imagery. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-1/W1, 465–471. 10.5194/isprs-archives-xlii-1-w1-465-2017.

Girard, N., Smirnov, D., Solomon, J., Tarabalka, Y., 2020. Polygonal Building Segmentation by Frame Field Learning. 10.48550/ARXIV.2004.14875.

Goebbels, S., Pohle-Fröhlich, R., 2023. Automatic Reconstruction of Roof Overhangs for 3D City Models. *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, SCITEPRESS - Science, Technology Publications, 145–152. 10.5220/0011604200003417.

Gröger, G., Kolbe, T.H., Nagel, C., Häfele, K.-H., 2012. OGC City Geography Markup Language (CityGML) Encoding Standard. Open Geospatial Consortium. https://portal.opengeospatial.org/files/?artifact_id=47842.

Huang, J., Stoter, J., Peters, R., Nan, L., 2022. City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds. *Remote Sensing*, 14(9), 2254. 10.3390/rs14092254.

Kaden, R., Kolbe, T.H., 2014. Simulation-Based Total Energy Demand Estimation of Buildings using Semantic 3D City Models. *International Journal of 3-D Information Modeling*, 3(2), 35–53. 10.4018/ij3dim.2014040103.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J.D., Ledoux, H., 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 1, 100001. 10.1016/j.ophoto.2021.100001.

Kong, G., Fan, H., Lobaccaro, G., 2022. Automatic building outline extraction from ALS point cloud data using generative adversarial network. *Geocarto International*, 37(27), 15964–15981. 10.1080/10106049.2022.2102246.

Liu, K., Ma, H., Zhang, L., Gao, L., Xiang, S., Chen, D., Miao, Q., 2024. Building outline extraction using adaptive tracing alpha shapes and contextual topological optimization from airborne LiDAR. *Automation in Construction*, 160, 105321. 10.1016/j.autcon.2024.105321.

Oosterom, P. van, Quak, W., Tijssen, T., 2005. *About Invalid, Valid and Clean Polygons*. *Developments in Spatial Data Handling*, Springer Berlin Heidelberg, 1–16. 10.1007/3-540-26772-7_1.

Pađen, I., Peters, R., García-Sánchez, C., Ledoux, H., 2024. Automatic high-detailed building reconstruction workflow for urban microscale simulations. *Building and Environment*, 265, 111978. 10.1016/j.buildenv.2024.111978.

Panday, U.S., Gerke, M., 2012. Fitting of Parametric Building Models to Oblique Aerial Images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVIII-4/W19, 233–238. 10.5194/isprsarchives-xxxviii-4-w19-233-2011.

Peters, R., Dukai, B., Vitalis, S., Liempt, J. van, Stoter, J., 2022. Automated 3D Reconstruction of LoD2 and LoD1 Models for All 10 Million Buildings of the Netherlands. *Photogrammetric Engineering and Remote Sensing*, 88(3), 165–170. 10.14358/pers.21-00032r2.

Rottensteiner, F., Sohn, G., Jung, J., Gerke, M., Baillard, C., Benitez, S., Breitkopf, U., 2012. The ISPRS benchmark on urban object classification and 3d building reconstruction. 10.15488/5042.

Saadaoui, H., Farah, S., Lechgar, H., Ghennioui, A., Rhinane, H., 2025. Advancing Urban Roof Segmentation: Transformative Deep Learning Models from CNNs to Transformers for Scalable and Accurate Urban Imaging Solutions—A Case Study in

Ben Guerir City, Morocco. *Technologies*, 13(10), 452. 10.3390/technologies13100452.

Varney, N., Asari, V.K., Graehling, Q., 2020. DALES: A Large-scale Aerial LiDAR Data Set for Semantic Segmentation. 10.48550/ARXIV.2004.11985.

Widyaningrum, E., Gorte, B., Lindenbergh, R., 2019. Automatic Building Outline Extraction from ALS Point Clouds by Ordered Points Aided Hough Transform. *Remote Sensing*, 11(14), 1727. 10.3390/rs11141727.

Widyaningrum, E., Peters, R.Y., Lindenbergh, R.C., 2020. Building outline extraction from ALS point clouds using medial axis transform descriptors. *Pattern Recognition*, 106, 107447. 10.1016/j.patcog.2020.107447.

Wu, T., Hu, X., Ye, L., 2016. Fast and Accurate Plane Segmentation of Airborne LiDAR Point Cloud Using Cross-Line Elements. *Remote Sensing*, 8(5), 383. 10.3390/rs8050383.

Wu, T., Vallet, B., Brédif, M., Caraffa, L., Vosselman, G., 2026. Reverse engineering LiDAR pulse emission points. *(Manuscript under review) ISPRS Open Journal of Photogrammetry and Remote Sensing*.

Zachar, P., Bakuła, K., Ostrowski, W., 2023. CENAGIS-ALS Benchmark - New Proposal for Dense ALS Benchmark Based on the Review of Datasets and Benchmarks for 3D Point Cloud Segmentation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-1/W3-2023, 227–234. 10.5194/isprs-archives-xlviii-1-w3-2023-227-2023.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., Yan, G., 2016. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sensing*, 8(6), 501. 10.3390/rs8060501.

Zolanvari, S.M.I., Ruano, S., Rana, A., Cummins, A., Silva, R.E. da, Rahbar, M., Smolic, A., 2019. DublinCity: Annotated LiDAR Point Cloud and its Applications. 10.48550/arXiv.1909.03613.

4. Conclusion

How to generate coherent building roofprints and footprints from high-density ALS point clouds and existing imprecise outlines?

- How to identify and use the points on roof edges in ALS point clouds?
- How to identify and use the points in an ALS point cloud that contain information about the façades despite their sparsity?
- How to deform an imprecise outline with global and local transformations while preserving the angles of the edges?

To answer the main research question, I proposed a method to turn imprecise outlines into roofprints and footprints sequentially from ALS data. The same general idea is used for the two outlines: first identify the points containing information about the outline, and then iteratively deform the initial imprecise outline to fit with the selected points. The resulting roofprints were tested successfully on the French national datasets BD TOPO and LiDAR HD, while the footprints were only qualitatively assessed and still require further work.

Regarding points on roof edges, I showed how most of them can be identified with simple topology-aware operations on ALS point clouds. These points can then be used to compute a roofprint in 2D, without having to manipulate the points in 3D for the optimisation of the polygon. It is however necessary to compute for each point on the roof edge the orientation of the building it corresponds to, in order to prevent the points from a given building to be used by another building.

As for the points containing information about the façades, I showed that it becomes significantly easier to identify them once a 3D roof model is obtained using the roofprint computed previously. This allows to simply select all the points below the roof, a simple solution that works well despite the sparsity of the ALS data on the façades. Therefore, the order in which creating the roofprint and the footprint is of great importance. Being able to use at the same time the points on the ground and the points on the façades proved to be challenging but very promising, as the points hitting the ground under the roof are very valuable and sometimes the only available information about the façade.

The question of preserving the interesting properties of the input outlines was tackled by developing a robust deformation algorithm enforcing the necessary constraints. This algorithm is capable of preserving the angles of the edges and the validity of the polygons while still leaving a lot of freedom for deformations. This algorithm proved to be very effective to tackle at the same time globally translating the outlines and locally modifying the relations between neighbouring edges. By iteratively focussing on edges one by one, it also reduces a complex problem with 2 dimensions of freedom per point into a succession of 1-dimensional problems.

This thesis overall fits perfectly into the programme of the MSc Geomatics for the Built Environment. Buildings and their different representations are at the core of some courses. The combination of different data sources is also at the core of the programme, in this case high-density ALS point clouds, which are becoming increasingly available in several countries, with the existing building outlines datasets, which many countries have in one form or another. Developing this whole method took a lot of the knowledge acquired during the master, such as how lidar works and how it affects the structure and properties of ALS point clouds, how validity of polygons is crucial to many applications, and how current methods to reconstruct 3D models of buildings work.

References

This section only contains the references mentioned outside of Chapter 3. They are treated independently of the references in Chapter 3, meaning that there could be duplicates.

Bauchet, J.-P., Sulzer, R., Lafarge, F., Tarabalka, Y., 2024. SimpliCity: Reconstructing Buildings with Simple Regularized 3D Models. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, 7616–7626. 10.1109/cvprw63382.2024.00757.

Biljecki, F., Ledoux, H., Stoter, J., 2016. An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, 59, 25–37. 10.1016/j.compenvurbsys.2016.04.005.

Gröger, G., Kolbe, T.H., Nagel, C., Häfele, K.-H., 2012. OGC City Geography Markup Language (CityGML) Encoding Standard. Open Geospatial Consortium. https://portal.opengeospatial.org/files/?artifact_id=47842.

Huang, J., Stoter, J., Peters, R., Nan, L., 2022. City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds. *Remote Sensing*, 14(9), 2254. 10.3390/rs14092254.

IGN, 2024a. Un appel à communs pour la conception du Jumeau numérique de la France et de ses territoires. https://www.ign.fr/appel-communs-jumeau-numerique-france-et-territoires (30 December 2025).

IGN, 2024b. La recherche au défi du jumeau numérique de la France. https://www.ign.fr/mag/la-recherche-au-defi-du-jumeau-numerique-de-la-france (30 December 2025).

Pađen, I., Peters, R., García-Sánchez, C., Ledoux, H., 2024. Automatic high-detailed building reconstruction workflow for urban microscale simulations. *Building and Environment*, 265, 111978. 10.1016/j.buildenv.2024.111978.

Peters, R., Dukai, B., Vitalis, S., Liempt, J. van, Stoter, J., 2022. Automated 3D Reconstruction of LoD2 and LoD1 Models for All 10 Million Buildings of the Netherlands. *Photogrammetric Engineering and Remote Sensing*, 88(3), 165–170. 10.14358/pers.21-00032r2.

Wu, T., Vallet, B., Brédif, M., Caraffa, L., Vosselman, G., 2026. Reverse engineering LiDAR pulse emission points. *(Manuscript under review) ISPRS Open Journal of Photogrammetry and Remote Sensing*.

A. Declaration of AI/LLM usage

Some AI tools were used during this thesis. The main usage comes from the use of GitHub Copilot, which was used to assist in writing code at all stages of the project. Then, Perplexity was used to look for explanation of concepts and to find relevant literature to start with at different stages of the project. No AI tool was used when writing this report.

All methods, results, analyses, and conclusions are my own, and I verified that the content is original and meets academic integrity standards.

B. Reproducibility

All data used during this thesis is openly available on the websites where they are distributed (data.gouv.fr):

- The BD TOPO dataset is available at data.gouv.fr/datasets/bd-topo-r
- The LiDAR HD dataset is available at data.gouv.fr/datasets/nuages-de-points-lidar-hd

An implementation of the method proposed in this thesis is available at github.com/alexandre-bry/MSc_Thesis-Code. All instructions to run the code and reproduce the results are available there, including the versions of the dependencies, and commands to download the data from the official sources. The validation dataset will also be published, and the link will be displayed in the same repository.

There is also another repository containing the notes, weekly reports and other materials produced during the thesis at github.com/alexandre-bry/MSc_Thesis-Report. The content of this repository is also available in the form of a website, which link can be found in the GitHub repository.

C. Glossary

Datasets

BAG (Basisregistratie Adressen en Gebouwen) — The Dutch dataset of buildings outlines with addresses and other attributes. More information at https://www.kadaster.nl/zakelijk/registraties/basisregistraties/bag. 1

3DBAG — An open 3D building data set that is generated fully automatically based on lidar data and covers the whole of the Netherlands. More information at https://docs.3dbag.nl/en/. 1, 2

AHN (Actueel Hoogtebestand Nederland) — A programme to produce high-density point clouds on the whole Dutch territory, with a new version every few years. More information at https://www.ahn.nl/. 1, 2, 4, 5

BD TOPO — The French dataset that contains (among many other types of objects) the buildings outlines, combining footprints and roofprints. More information at https://www.data.gouv.fr/datasets/bd-topo-r. i, 2, 12, A-2

LiDAR HD — The first project to collect high-density point clouds on the whole territory of France (except Guyane). More information at https://www.data.gouv.fr/datasets/nuages-de-points-lidar-hd. i, 1, 2, 4, 5, 6, 12, A-2

JUNN — A French public project officially launched in 2026 aiming at creating a nationwide digital twin. More information at https://junn-france.fr/. 1

Entities

TU Delft (Delft University of Technology) — The Dutch university in which this project was carried on. More information at https://www.tudelft.nl/en/. ii, 1, 3

IGN (Institut national de l'information géographique et forestière) — The reference public operator for geographic and forest information in France. More information at https://www.ign.fr/institut/identity-card. ii, 1, 2

Software

roofer — An algorithm to reconstruct a 3D building model from a point cloud and a 2D roofprint polygon in different LoDs up to 2.2. More information at https://github.com/3DBAG/roofer. 1, 2, 5

Vocabulary

footprint — The 2D outer boundary defined by the vertical projection of the outer walls/façades of a building. i, 2, 3, 4, 5, 10, 12, A-3

roofprint — The 2D outer boundary defined by the vertical projection of the roof of a building. i, 2, 3, 4, 5, 9, 10, 12, A-3

outline — Purposefully vague term designing either a roofprint or a footprint or when they are mixed and the differentiation cannot be made. i, 1, 2, 3, 4, 9, 10, 12, A-3

LoD (Level of Detail) — A concept to differentiate multi-scale representations of semantic 3D city models. It is in practice principally used to indicate the geometric detail of a model, primarily of buildings. More information at https://3d.bk.tudelft.nl/lod/. i, 1, 2, 4, 5, 9, 10, A-3

lidar (Light Detection and Ranging) — A method for determining ranges by targeting an object or a surface with a laser and measuring the time for the reflected light to return to the receiver. (Wikipedia) More information at https://en.wikipedia.org/wiki/Lidar. 5, 12, A-3, A-4

ALS (Airborne Laser Scanning) — Aircraft-mounted lidar that emits laser pulses toward the ground, capturing millions of georeferenced points for large-scale terrain and vegetation models. More information at https://en.wikipedia.org/wiki/Lidar. i, 1, 3, 4, 5, 6, 9, 10, 12

MLS (Mobile Laser Scanning) — Vehicle-mounted lidar that records dense point clouds while moving, used for detailed street-level 3-D mapping. More information at https://en.wikipedia.org/wiki/Lidar. 4

TLS (Terrestrial Laser Scanning) — Tripod-mounted static lidar that scans surroundings from fixed positions, ideal for high-resolution models of structures and heritage sites. More information at https://en.wikipedia.org/wiki/Lidar. 4