Automatic high-detailed building reconstruction workflow for urban microscale simulations

Pađen, Ivan; Peters, Ravi; García-Sánchez, Clara; Ledoux, Hugo

DOI
10.1016/j.buildenv.2024.111978

Publication date
2024

Document Version
Final published version

Published in
Building and Environment

Building and Environment 265 (2024) 111978

Contents lists available at ScienceDirect

Building and Environment

journal homepage: www.elsevier.com/locate/buildenv

Automatic high-detailed building reconstruction workflow for urban microscale simulations

Ivan Pađen [a,*], Ravi Peters [b], Clara García-Sánchez [a], Hugo Ledoux [a]

[a] *3D Geoinformation Research Group, Delft University of Technology, Delft, The Netherlands*
[b] *3DGI, Zoetermeer, The Netherlands*

ARTICLE INFO

Keywords:
3D city modelling
Automatic building reconstruction
Geometry preparation
Urban wind flow
Computational fluid dynamics (CFD)
Digital twins

ABSTRACT

Reconstructing urban scenarios for computational fluid dynamics simulations typically requires significant manual effort, especially when higher geometrical details are required. To address this issue, we present a workflow to *automatically* reconstruct buildings in three levels of detail (LoDs): LoD1.2, LoD1.3, and LoD2.2, tailored to urban microscale simulations. The workflow uses a combination of building footprints and a point cloud to segment roof planes, create partitions, optimise planes, and finally assemble roof planes into 3D building models. Reconstructed buildings are seamlessly integrated into the terrain together with different surface layers such as water, low vegetation, and paved surfaces. Apart from three general LoDs, building footprints can be simplified as a part of the 2D generalisation; additionally, smaller surfaces such as chimneys and ventilation shafts can be removed using a graph-cut optimisation. The integrated geometry validator can report on validity of building models, such as watertightness, manifoldness, or occurrences of self-intersections. In the case of invalid geometries, we can generate an approximation: geometry repair the with alpha wrapping algorithm, or reconstruction in lower LoD. We tested our implementation on two different real-world datasets — one in The Netherlands, and another one in the USA. The results showed that 95% (Dutch dataset) and 90% (US dataset) buildings were valid according to the ISO 19107 standard. Generated grids showed satisfactory quality as we observed monotonous convergence in simulations with grid convergence indices up to 3.8% for pressure and velocity variables. These results indicate that the workflow is suitable for typical urban microscale simulations.

1. Introduction

Compared to other stages of a simulation, the preprocessing step in a computational fluid dynamics (CFD) workflow typically requires the most human effort [1]. A substantial part of that effort involves preparing geometries for the computational grid generation software [2].

Geometries used in urban microscale (or urban flow) simulations are especially tedious to prepare because they are heterogeneous, complex, and large in scale. Publications such as Toja-Silva et al. [3] and Hågbo et al. [4] used detailed city models and the authors reported the long time required to prepare those models for simulation. Additionally, the input data often come from different sources, requiring format conversions and adaptations that not only take time but can also corrupt data and introduce errors.

Geometries used in urban microscale simulations have to meet certain criteria in order to result in high-quality computational grids. We can divide the criteria into two main categories:

1. Validity — input geometries should not contain errors such as gaps/missing faces, self-intersections, non-manifold edges, degenerate faces with no finite area, duplicate vertices and faces [5,6]. Those criteria can be summarised as the rules for valid 3D geometries in the geoinformation (GIS) community called ISO 19107 [7].
2. Resolution — can be expressed through the level of detail (LoD) [8]. In the broadest sense, geometries should have as many details as the capabilities of the grid generator and computational resources allow.

The influence of the criteria depends on the simulation method and capabilities of different mesh generation software. For example, some misrepresentations such as small gaps have a limited negative impact on mesh generation [9], but self-intersections and non-manifold edges might create bigger issues or even make grid generation impossible, as indicated by Zheng et al. [10] and Lefieux et al. [11]. To cover

* Corresponding author.
E-mail addresses: i.paden@tudelft.nl (I. Pađen), ravi.peters@3dgi.nl (R. Peters), c.garcia-sanchez@tudelft.nl (C. García-Sánchez), h.ledoux@tudelft.nl (H. Ledoux).

https://doi.org/10.1016/j.buildenv.2024.111978
Received 27 March 2024; Received in revised form 31 July 2024; Accepted 15 August 2024
Available online 21 August 2024

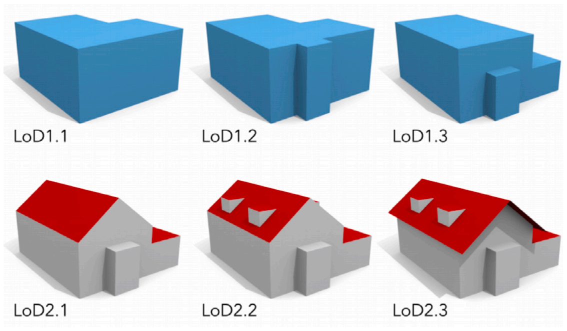

Fig. 1. LoD classification used in this article.
Source: Adapted from Biljecki et al. [16]

as many numerical methods and grid generators as possible, datasets should contain high-quality, error-free geometries.

At the same time, a typical 3D city model contains many errors, as was reported by Biljecki et al. [12]. The reason is that 3D models are created mostly with visualisation in mind [13], whose requirements are less stringent when it comes to validity compared to urban microscale simulations. Even with the use of automatic geometry repair tools, Saeedrashed and Benim [6] showed that manual labour is still necessary. This implies that most existing 3D city datasets are not suitable for numerical simulations and that their preparation is time-consuming; we believe that the fastest approach would be to reconstruct city geometries with purpose-built algorithms.

As we outline in Section 2, purpose-built automatic reconstruction algorithms for urban microscale simulations are scarce. Those that exist and are replicable, to the best of our knowledge, do not reconstruct buildings higher than the simple polygon extrusion, that is, the level of detail (LoD) 1.2 (see Section 2.1 for the concept of LoD). Conversely, authors like Ricci et al. [14] and Hågbo et al. [4] showed that LoD1.2 might not be accurate enough for certain urban microscale simulations. Clearly, there is a need for reconstruction in higher LoDs, which was also noted by Mirzaei [15] in their recent review article.

To address this issue, we adapted and expanded upon the high-detailed, multi-LoD reconstruction algorithm by Peters et al. [17], introducing a larger workflow for automatic reconstruction of 3D city models for urban microscale simulations. The original algorithm uses the combination of building footprints and airborne lidar data to reconstruct building geometries in LoD1.3 and LoD2.2. Additionally, it enables finer control over the final model complexity using the following methods: first, building footprints, and thus outline, can be simplified, and second, a graph-cut optimisation, introduced in Section 3.1.2 can be used to reduce the number of planes.

Our workflow (detailed in Section 3.2) expands on the initial work by reconstructing the rest of the urban scenario required for numerical simulations while ensuring the geometric validity of the output. This includes the reconstruction of terrain from a point cloud with surface smoothing and remeshing, seamless integration of terrain and buildings, imprinting surface layers into terrain, and enclosing the domain with boundaries. Additionally, the workflow can automatically define which buildings are being reconstructed (i.e. the influence region) using the best practice guidelines (BPGs). Furthermore, we integrated the geometry validator of Ledoux [18] and we provide fallbacks in the form of geometry repair with alpha wrapping [19] and reconstruction in lower LoD if the geometric validity is crucial (see Section 3.1.5).

By combining the features above, the output of our workflow is a geometry of an urban scenario containing triangulated buildings which are intended to be valid, i.e., watertight, two-manifold, and without duplicated faces/vertices. The output can be directly used in investigations such as pedestrian wind comfort [20], urban heat island effects [21], pollutant dispersion [22], and urban air mobility [23], or as a surrounding urban scenario that supports investigation of a very detailed and manually reconstructed building model [24].

In Section 4 we report on tests we performed with LoD2.2 reconstruction of two different datasets — one including the campus of the Delft University of Technology in the Netherlands, and another one covering the campus of Stanford University, CA, USA. The results show 95% and 90% of valid buildings respectively according to the ISO 19107 standard. Generated grids resulted in monotonous convergence in simulations, with maximum relative errors of 3% and grid convergence indices (GCIs) of 3.8% for pressure and velocity variables. The tested reconstructions and subsequent simulations suggest that our workflow is suitable for urban microscale simulations.

2. Related work

2.1. The effect of the level of detail

There is no agreement on the LoD classification of geometries used in urban flows. Therefore, before mentioning publications that investigated the effect of LoD, we need to define the LoD classifications. We propose and will use throughout this paper the classification by Biljecki et al. [16] as it is well established in the 3D city modelling field. Fig. 1 shows a few LoDs relatable to urban microscale simulations.

Few authors have made initial steps in investigating the effect of different LoDs. For example, Ricci et al. [14] conducted experimental and numerical comparisons of three LoDs (visually, we identified them as LoD1.1, LoD1.3 and LoD2.1); the authors indicated that LoD1.1 showed large differences in mean velocity in the influence region at pedestrian height compared to LoD1.3 and LoD2.1, while LoD1.3 and LoD2.1 produce similar results. García-Sánchez et al. [25] observed pronounced differences in velocities at a pedestrian height between different LoDs. Findings by Hågbo et al. [4] were similar — LoD1.2 and LoD2.1 models they compared showed notable differences in the results of pedestrian wind comfort.

These first investigations suggested that LoDs of LoD1.3 and higher might be necessary for urban flow simulations. Further research is necessary to give a definitive answer on the acceptable LoD, but to achieve this answer, building reconstruction in multiple LoDs is greatly beneficial.

2.2. Automatic building reconstruction for urban microscale applications

We concluded in the previous section that multi-LoD building reconstruction and buildings of LoD1.3 and higher might be necessary for urban microscale simulations. Conversely, most published workflows dealing with geometry creation for urban flow simulations do not reconstruct buildings in more detail than LoD1.2.

For example, the workflow by Gargallo-Peiró et al. [26] uses typical inputs from GIS – 2D footprints or topographical maps and point clouds acquired by airborne lidar or photogrammetry — to reconstruct terrain and buildings in LoD1.2, later used with TetGen [27] to create the simulation grid.

Camelli et al. [28] used the combination of a gridded elevation model and building footprints containing height data (stored as an attribute) to create LoD1.2 building geometries of Oklahoma City. Sun et al. [29] combined lidar data with deep learning to reconstruct LoD1.2 buildings, terrain and vegetation which were then meshed and simulated using Phoenics CFD software. Naserentin and Logg [30] integrated tetrahedral mesh generation in their 3D city reconstruction workflow, which also reconstructs buildings in LoD1.2 Even our recent publication [31] uses LoD1.2 for building reconstruction. The reason is that automated LoD1.2 reconstruction is very simple in complexity compared to higher LoDs. The only relevant parameters are the building base and rooftop elevations.

Recently, Alemayehu and Bitsuamlak [32] published an article focusing on high-fidelity reconstruction for urban flows using deep learning with satellite imagery and lidar data as input. It is, to the best of

Table 1
Reproducible automatic LoD2 building reconstruction algorithms.

Publication	Input
Zhou and Neumann [33]	PC
Gui and Qin [34]	SI
Peters et al. [17]	FP, PC
Huang et al. [35]	FP, PC

FP - footprints, *PC* - point cloud, *SI* - satellite imagery.

our knowledge, the first publication dealing with LoD2 reconstruction tailored to urban microscale simulations. The authors classify their resulting buildings as LoD3, but through visual inspection, we would categorise them as LoD2.2 due to slanted roofs and *2.5D geometry with vertical walls* (see Section 3.1 for the description of the term). Reconstructed parts of London, Ontario, Canada, were used to simulate pedestrian wind comfort using the StarCCM software.

2.3. High-detailed building reconstruction

While workflows tailored to urban microscale simulations are mainly developed for LoD1.2 building reconstruction, the developments in the 3D city modelling field are showing an emergence of LoD2 reconstruction algorithms that aim to automatically generate high-quality, robust, valid, and large-scale building geometries. While there are many publications dealing with building surface reconstruction, we will focus on a few of them that are, first, in our opinion applicable to urban flow studies, and second, reproducible in terms of published code. The algorithms are summarised in Table 1.

The publication by Zhou and Neumann [33], based on the 2.5D dual contouring method presented in [36], enables LoD2.2 building reconstruction from airborne lidar. Buildings are guaranteed to be watertight and the authors report a 0.6% to 1.1% building error rate on datasets with up to 418 buildings.

Gui and Qin [34] developed a workflow that reconstructs LoD2.1 models combining a satellite-derived digital surface model (DSM) and an orthophoto. It is a GPU-based implementation which is, according to authors [37], able to process 2.5×2.5 km domains at a time.

Huang et al. [35] developed a large-scale LoD2.2 reconstruction algorithm based on the work of Nan and Wonka [38]. The algorithm conducts optimisation to ensure a correct topology and enhance the recovery of details. While the authors report that their algorithm ensures watertight and manifold building models, the method can be very computationally demanding for complex buildings. The authors used the algorithm to reconstruct 20 thousand real-world buildings.

The algorithm presented in [17] enables building reconstruction in LoD1.2, LoD1.3, and LoD2.2. The authors demonstrated the efficiency of the algorithm by reconstructing 10 million buildings in the Netherlands as a part of the 3DBAG dataset [39]. In addition, the algorithm aims to reconstruct buildings valid according to the ISO 19107 standard, and the quality assessment of the 3DBAG in [40] reported 90% geometric validity of LoD2.2 models, which was increased to 99% validity in subsequent versions of the 3DBAG dataset, using *val3dity* [18] and a root mean squared distance of 10 cm from the input point cloud to models. We found this combination of computational efficiency and geometric validity, among other features (detailed further in Section 3.1), a compelling case for its application in urban microscale simulations; therefore, we proceeded with this algorithm as a base for our implementation.

3. Methods

3.1. Automatic LoD1.3 and LoD2.2 reconstruction algorithm

The automatic reconstruction algorithm by Peters et al. [17], which we use as a base for our implementation, combines building footprints

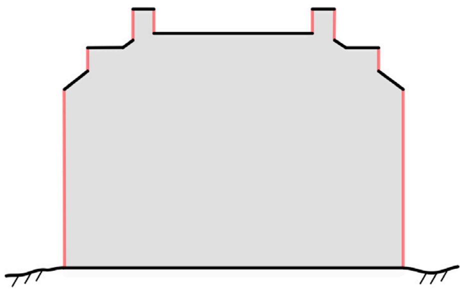
Fig. 2. Schematic representation of a 2.5D building model with vertical walls (coloured in red).

and a point cloud for input. While the method is primarily data-driven (based solely on the input data) it also encompasses some model-driven properties (based on strong assumptions about the shape of the building) [41]. The assumptions used in the model-driven approaches typically aim to overcome defects in the input point cloud and lower the model complexity. We use the following assumptions:

- Building shapes are approximated with planar faces extracted from a point cloud — in other words, they are *piecewise planar*,
- Building models are *2.5D with vertical walls*, as shown in Fig. 2; as a result, building roofs can be reconstructed with 2D planar partitions of roof planes. This simplifies the problem and results in a fast algorithm; however, this also means that features like roof overhangs and balconies cannot be modelled. The 2.5D assumption is in line with the state-of-the-art automatic 3D reconstruction algorithms where the emphasis is placed on the reconstruction of building roofs, as the building sides suffer from missing data, typically due to the occlusion effect in the case of airborne lidar [35].

Apart from these assumptions, additional requirements are set on the input data to ensure high-quality reconstruction for urban microscale simulations:

- Point clouds and footprints are properly aligned. Not only must the input data be in the same coordinate reference system (CRS), but also the positional error (horizontal shift) between them must be minimised. Otherwise, the reconstruction could result in a bad data fit.
- Point clouds should have buildings and terrain classified. While our workflow contains the optional cloth simulation filter (CSF) by Zhang et al. [42] to separate ground and non-ground points, the non-ground points still contain vegetation and other off-ground artefacts that might interfere with the reconstruction.

The main goal of the algorithm is to extract a so-called *roof partition* — a planar partition of the building footprint where each face corresponds to a roof plane projected to the ground (see Fig. 3).

The following sections give comprehensive information about the reconstruction algorithm, complementing the original publication [17] with additional details.

3.1.1. Feature extraction and regularisation

The first part includes plane detection, line extraction, and line regularisation from the input point cloud. Initially, building planes are detected using a region-growing algorithm that segments the point cloud into groups of points that fit well with a locally estimated plane. Afterwards, boundary and intersection lines are derived from detected planes (Fig. 4(a)); the boundary lines represent the outer boundary of a detected plane's α-shape [43], whereas the intersection lines are located at intersections of adjoining planes.

Next, the boundary and intersection lines are clustered depending on their orientation (Fig. 4(b)) and distance (Fig. 4(c)). One average

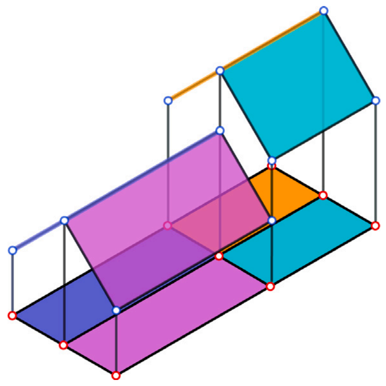

Fig. 3. An example of roof planes (containing blue vertices) projected onto the building footprint to form the roof partition (containing red vertices). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

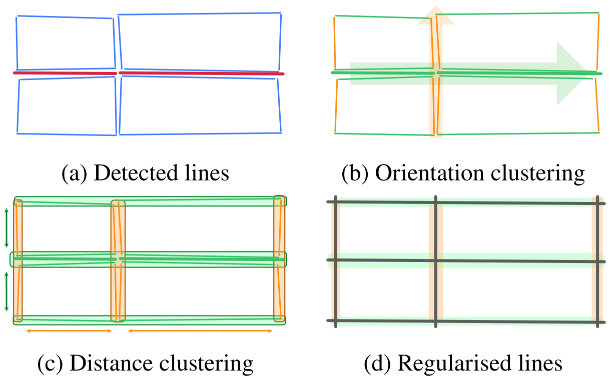

Fig. 4. Four steps of line regularisation through clustering.

line per cluster is derived, resulting in regularised lines as shown in Fig. 4(d).

At the same time, building footprints are also processed in this part of the algorithm. We utilise the Ramer–Douglas–Peucker algorithm [44, 45] to reduce the complexity of footprints, which we refer to as the "2D simplification". Additionally, during this part, the ground elevation is also approximated using a buffer of ground points surrounding the footprint.

3.1.2. *Roof partitions and optimisation*

In the second part, the initial roof partition is constructed from regularised lines and stored in a doubly connected edge list (DCEL) [46]. All line intersections are defined as vertices during the construction of DCEL, and there are no dangling edges.

Resulting partitions are typically overly complex — consequently, the reconstruction would result in too many small faces. To lower the number of resulting planes while keeping the model accuracy high, the algorithm performs a graph-cut optimisation based on the method of Zebedin et al. [47]. It is an energy minimisation equation defined as

$$E(f) = \lambda \sum_{p\in P} D_p\left(f_p\right) + (1-\lambda) \sum_{\{p,q\}\in N} S_{p,q}\left(f_p, f_q\right), \tag{1}$$

where $D_p\left(f_p\right)$ is called the *data term* which measures data fidelity obtained by assigning the label f_p to a roof partition p. The set of possible labels is formed by the set of planes that were earlier detected in the point cloud. The optimisation simplifies the roof partitions in such a way that adjacent roof partitions are merged by assigning the same label while simultaneously maintaining good data fidelity. The data fidelity is expressed as the sum of the height differences between the points in a point cloud bounded by candidate roof partitions, and the estimated elevation on a roof partition:

$$D_p\left(f_p\right) = \sum_{x\in p} \left[h_{\mathrm{PC}}\left(x_i\right) - h_{\mathrm{CRP}}\left(x_i\right)\right], \tag{2}$$

with $h_{\mathrm{PC}}\left(x_i\right)$ denoting a point cloud elevation at a location x_i, and $h_{\mathrm{CRP}}\left(x_i\right)$ representing interpolated elevation at the location x_i when label f_p would be assigned.

The second term in Eq. (1) is the *smoothness term* where

$$S_{p,q}\left(f_p, f_q\right) = \begin{cases} l_{\mathrm{b}}\left(f_p, f_q\right) & \text{if } f_p \neq f_q \\ 0 & \text{if } f_p = f_q \end{cases}. \tag{3}$$

In the equation above, $l_{\mathrm{b}}\left(f_p, f_q\right)$ is the border length between labelled partitions f_p and f_q. The smoothness term favours homogenised regions by penalising adjacent roof partitions with different labels.

In this paper, we introduce the parameter λ to the original formulation and call it the *complexity factor*. The lower the complexity factor, the more the optimisation favours fewer borders with lower data fidelity as a consequence. Simply put, by lowering λ, one can remove smaller details from the planar partition, and ultimately, de-feature the model. The opposite effect happens by increasing the factor.

Since it is normalised, λ with values approaching 1 results in a case that closely resembles the original point cloud, with the extreme case including all planes detected by the region-growing algorithm.

On the other hand, λ with the value of 0 will label all polygons as one group. With the cancelled out data term, the energy minimisation problem $\min\left(S_{p,q}\left(f_p, f_q\right)\right)$ equals 0 in a case where the total length of borders is 0 m, meaning there are no borders between groups. As a consequence, the outcome of this extreme is a model resembling LoD1.2 reconstruction. We say resembling because it can either be true LoD1.2, or LoD2.0 with one slanted surface. This is because footprint vertices can still be lifted to different elevations (while keeping the surface planar) if it better describes the input point cloud.

The graph-cut optimisation with the weighted complexity factor is very useful for computational wind engineering applications, as it introduces control over building complexity and it is simple to use for the end users. This functionality forms the second part of our simplification, which we named "3D simplification". The results of a changing complexity factor are shown in Fig. 5.

The output of the optimisation, $\min(E(f))$, contains roof partitions clustered in labelled groups. Edges between roof partitions within the same groups are removed to form the final planar partitioning of the footprint, now called *roof parts*.

3.1.3. *Downwards extrusion*

The final part of the reconstruction algorithm is creating a building geometry by extruding the roof parts downwards to terrain elevation. The created building faces can be separated into three categories depending on their reconstruction procedure:

1. The *ground face* edges are incident to the building footprint. Vertex elevations are obtained by conducting a natural neighbour interpolation [48,49] directly from the terrain (see Section 3.2 for terrain reconstruction).
2. *Wall faces* are vertical faces that connect the ground face with roof faces, as well as borders of incident roof parts. The faces are created in such a way to result in topologically correct geometries. This includes solving vertices of incident planes that have different heights.
3. *Roof faces* are reconstructed from the planes that were assigned to the roof parts during optimisation.

The order of vertices in all planes is set to counter-clockwise when viewed from the outside so that the normals are pointing outwards, as mandated by the ISO9107 [7].

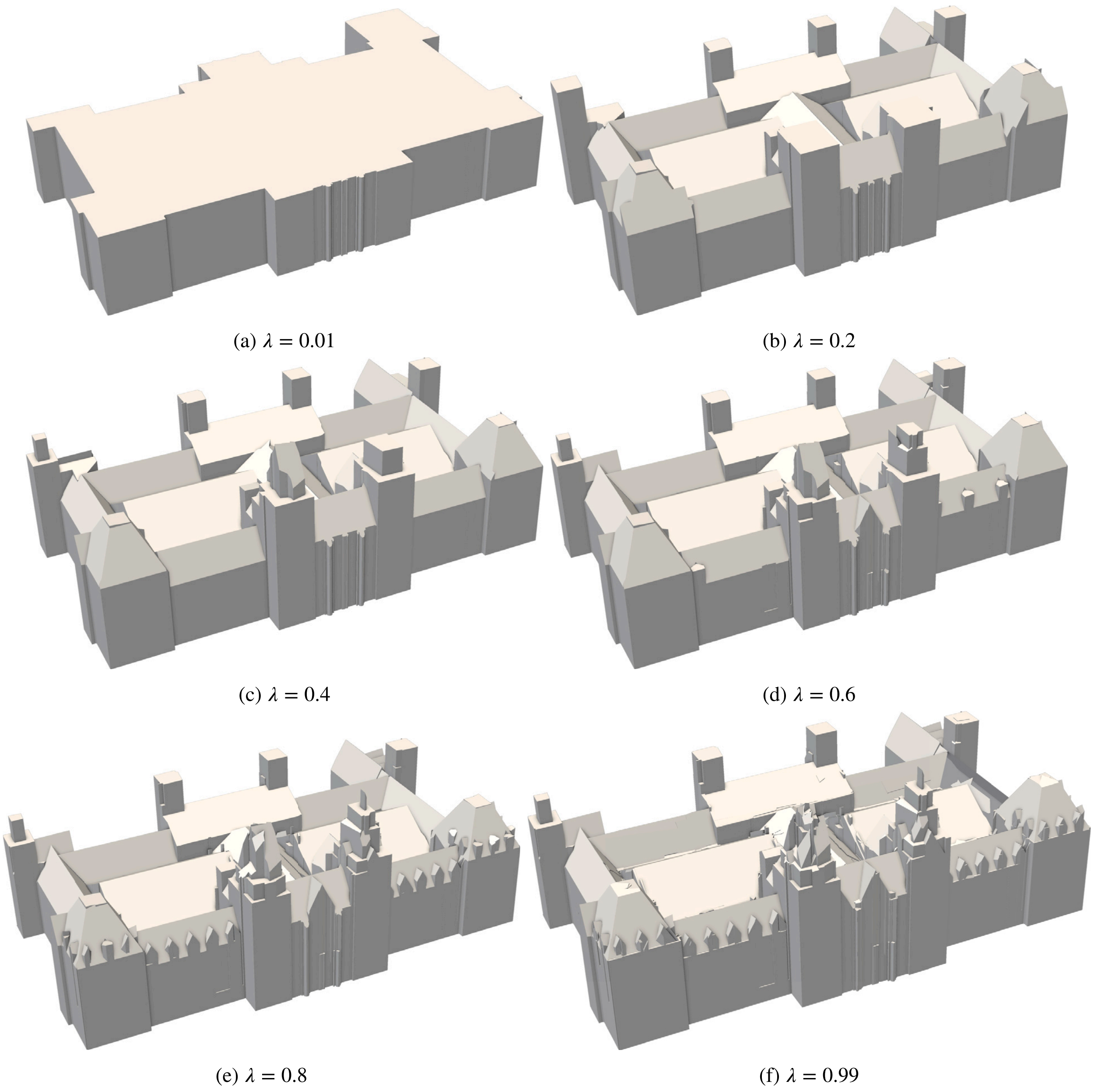

Fig. 5. LoD2.2 reconstruction with different complexity factors λ.

3.1.4. Differences between LoD1.3 and LoD2.2

To achieve the LoD1.3 reconstruction, vertices of individual roof parts are set to a predefined percentile calculated from elevations of all original points bound by the roof part, resulting in horizontal roof planes. This stage comes after the optimisation described in Section 3.1.2.

Additional functionality is added that enforces a minimum jump in heights between planes (i.e. the step size), with checks starting from the plane with the highest elevation. If, for instance, the targeted mesh resolution is 2 m, any two roof parts whose elevation difference is smaller than this threshold will be merged to the predefined percentile of all elevations belonging to those parts. Fig. 6 shows an example of LoD1.3 reconstruction with different step sizes. The extreme instance of this functionality is when the step size is sufficiently large (matching or larger than the building height); then, the resulting building will be reconstructed in LoD1.2, as seen in Fig. 6(e).

LoD2.2 reconstruction skips these two steps.

3.1.5. Geometric validity

Even though the algorithm targets valid reconstruction, problems can still arise due to different reasons, e.g. issues in plane detection, overlapping planes, regularisation gone wrong, numerical issues, etc. This is why our workflow can conduct validity inspection with the workflow of Ledoux [18], implemented in the library *val3dity*.

Whether the error is acceptable or not depends on the numerical method and capabilities of the grid generation software. This is why we added optional fallbacks in case the validity is of paramount importance for a numerical simulation.

The first fallback is the alpha wrapping algorithm by Portaneri et al. [19] which refines and carves a 3D Delaunay triangulation with empty balls of radius alpha on an offset surface of the input. It is a surface approximation approach where the approximation (and thus fidelity) can be tuned with user-defined parameters. The algorithm is robust and efficient, guaranteed to terminate and to produce valid geometries.

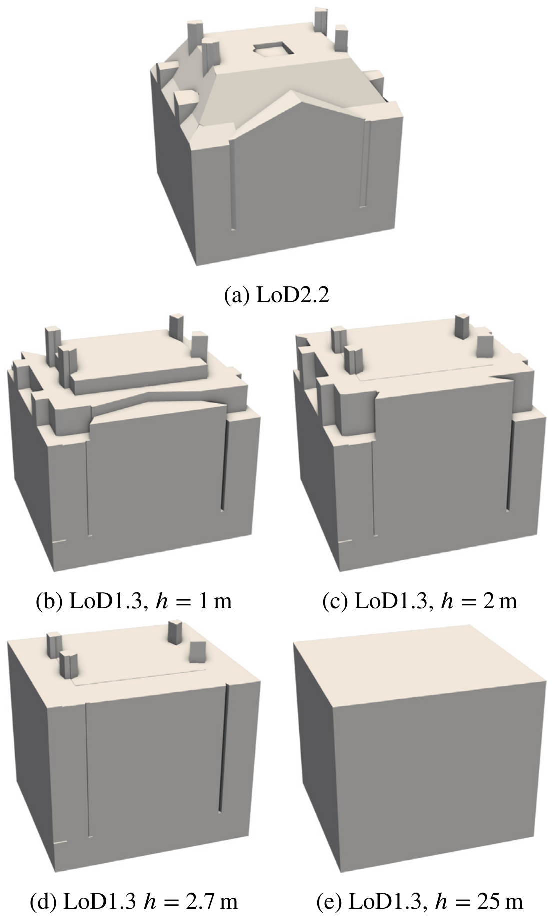

Fig. 6. Distinction between LoD2.2 and LoD1.3 reconstruction with different step sizes.

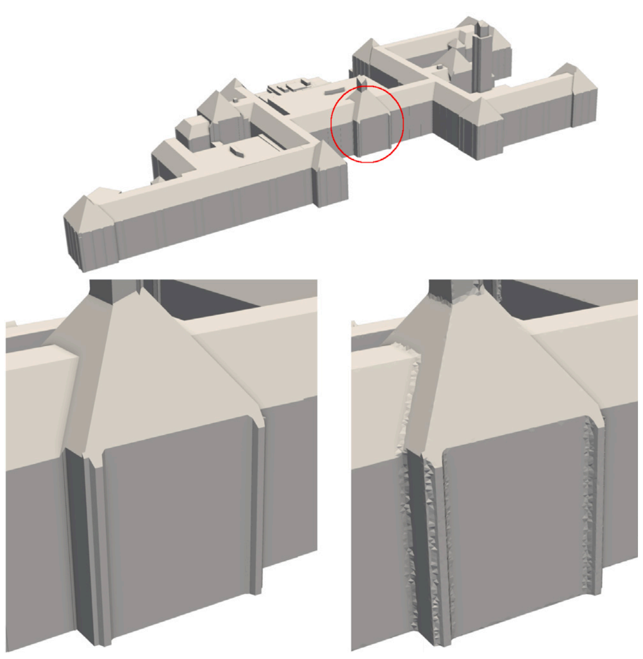

Fig. 7. Edge details of a building model before (left) and after (right) alpha wrapping.

The drawback of the algorithm is that it is an approximation of the original geometry and as such, it has a disadvantage in capturing sharp edges. Tighter tolerances (e.g. smaller carving ball) make an approximation better, however, capturing convex edges is a limitation. Such results can create issues in some aspects of computational mesh generation, e.g. when enforcing feature edges. An example of a resulting geometry from an alpha wrap is shown in Fig. 7.

The second fallback is a reconstruction of invalid geometries to the lower LoD1.2 since the implementation of LoD1.2 reconstruction ensures valid buildings. The two fallbacks are mutually exclusive.

3.2. *Workflow for urban microscale simulations*

We expanded the LoD1.3 and LoD2.2 reconstruction algorithm of Peters et al. [17] with the normalised complexity factor λ and geometric validity fallbacks into an automatic urban scenario reconstruction workflow specifically targeting urban microscale simulations. The workflow can be split into the following features: (1) buildings, (2) terrain, (3) surface layers, (4) influence regions, (5) domain boundaries.

First, as a part of the workflow, building geometries can also be reconstructed in LoD1.2 by gathering all building points encompassed by a footprint and calculating an arbitrary percentile to define building elevation. Additionally, LoD1.2 reconstruction is also available using polygon attributes, either through a building height attribute or an attribute containing the number of floors paired with a floor height. Building geometries can also be isotropically remeshed to target equilateral triangles with uniform edge lengths.

Second, we create the terrain geometry from ground points as a triangulated irregular network (TIN) using the constrained Delaunay triangulation (CDT). Smooth terrain surface is obtained in three steps: first, we apply the weighted locally optimal projection WLOP operator [50] on the input points; second, we construct an initial surface with the triangulation; third, we conduct the shape smoothing from Loriot et al. [51]. Building footprints are imprinted into the terrain as constraints, leading to good alignment of building and terrain surfaces.

Third, surface layers are polygons encompassing different types of surfaces such as water, low vegetation, roads, etc. We imprint those 2D polygons into the triangulated 3D terrain as constraints using the natural neighbour interpolation for elevation calculation. The algorithm removes duplicated edges during the insert (in case of adjacent polygons), and in the case of overlaps, it inserts additional vertices at intersections of edges. We then tag surface layers by traversing through constrained regions and exporting them as triangulated geometries, perfectly cutting them out from the terrain. As such, they can be used to impose roughness values associated with the type of the surface, typically by using the classification from Wieringa [52], as done in [53–56].

Fourth, the workflow includes an automatic definition of the influence region according to best practice guidelines (BPGs) by Liu et al. [57]. Furthermore, multiple LoDs can be combined in the influence region — one can, for example, reconstruct the very centre of the domain in LoD2.2, then define the second perimeter for LoD1.2 reconstruction, and then extract the third LoD1.2 region where the buildings can be parameterised (i.e. used as virtual buildings with drag terms).

Fifth, we can automatically define the domain scope using the BPGs from Tominaga et al. [58] and Franke et al. [59]. The rectangular and circular domains are available as the two representative domain types in urban microscale simulations [60]. A typical rectangular domain extends 5 times the tallest building height $H_{\max}$ from the influence region in the upwind, lateral, and vertical direction, as well as $15H_{\max}$ in the downwind direction [61]. Similarly, the circular domain extends $15H_{\max}$ radially from the influence region and $5H_{\max}$ from the highest building in the vertical direction, as indicated by Hågbo and Giljarhus [62]. Finally, a domain can be closed off from sides and top, depending on the requirements of grid generating software.

3.3. Implementation

We used our open-source automatic geometry reconstruction framework *City4CFD* [31] as a basis for the implementation of this workflow. The workflow is implemented in C++ using the Computational Geometry Algorithms Library (CGAL) [63] for low-level functionalities such as triangulations [64], spatial searching [65], polygon operations [66,67], and polygon mesh processing [51]. The reconstruction of buildings is performed concurrently, utilising *OpenMP* [68] library for parallel execution.

4. Test cases

We evaluated the implementation of our workflow in three steps:

1. Reconstruction of urban scenarios: in this step, we assessed the quality of the outputted building geometries, focusing on their adherence to the ISO 19107 standard for the validity of 3D primitives.
2. Computational grid generation for urban flow simulations: we used the reconstructed geometries without any manual modifications as input for an automatic computational grid generator. Through this process, we generated computational grids and reported on the grid quality indicators.
3. Running numerical simulations: finally, we conducted numerical simulations to calculate grid convergence indices (GCI) and to evaluate differences between reconstructions with low and high point cloud densities.

Two different locations were analysed:

1. The location of the Delft University of Technology's campus in Delft, The Netherlands (Fig. 8(a)). The three datasets used for reconstruction were the Dutch national lidar dataset AHN [69], the building footprint dataset BAG [70], and the polygon dataset BGT [71], the latter used to extract water and low vegetation surfaces. All datasets are open and freely available. The initial point cloud density was 17.4 pts/m^2 which we then thinned to 6.11 pts/m^2 to test whether the algorithm is suitable for low-density airborne lidar-based point clouds (based on classification by Stanley and Laefer [72]). We will refer to it as the "TU Delft case".
2. The location of the Science and Engineering Quad at Stanford University in Stanford, CA, USA (Fig. 8(b)). The two datasets used for reconstruction were open data footprints from OpenStreetMap [73] and the Santa Clara county point cloud from U.S. Geological Survey [74], also open and freely available. The input point cloud density in this case is 23.76 pts/m^2, classifying it as a high-density point cloud by Stanley and Laefer [72]. We will denote it as the "Stanford case".

We deliberately chose different point cloud resolutions to investigate the impacts of a low-density point cloud reconstruction. The impacts were investigated by comparing the results of numerical simulations using geometries reconstructed from high-density and low-density point clouds.

Even though our workflow can reconstruct multiple levels of detail, we focused on LoD2.2 as it represents the most complex scenario both for the reconstruction itself and the subsequent grid generation.

4.1. Geometry reconstruction

The two reconstructed areas are 3 km^2 (TU Delft) and 1.6 km^2 (Stanford), both representing a neighbourhood-scale case [75]. The TU Delft case was reconstructed as a rectangular domain, portraying a typical urban flow simulation with an inlet, outlet, and side domains [61]. The total number of buildings was 76 and 8213 other polygons represented water and low vegetation.

Table 2
Data related to geometry reconstruction.

	TU Delft	Stanford
Area	3 km^2	1.6 km^2
Point density	6.11 pts/m^2	23.76 pts/m^2
No. buildings	76	167
No. other polygons	8213	5583
No. invalid	4	18
No. failed recon.	6	0
Recon. time	136 s	385 s

The Stanford case was reconstructed as a round domain for a cylindrical grid, a validated configuration [60] that allows for easier setup of multiple wind directions, as demonstrated in [20,76,77]. The total number of buildings was 167, along with 5583 additional low vegetation polygons. Reconstructed geometries are shown in Fig. 8.

Table 2 shows the reconstruction details for the both cases. We can see that there are six failed reconstructions in the TU Delft case and zero failed reconstructions in the Stanford case. All failed reconstructions are due to insufficient points in the point cloud for LoD2.2 reconstruction. The algorithm recovered one building with LoD1.2 reconstruction as there were enough points for height estimation. Our workflow would be able to mitigate the issue with LoD1.2 reconstruction of other buildings if either of the footprint databases contained attributes for building height or number of floors.

The geometric validity report showed 5% and 10% of invalid buildings for the TU Delft and Stanford cases, respectively. In detail, the invalid buildings contained self-intersections and non-manifold edges. With a subsequent geometry repair with alpha wrapping (done automatically through City4CFD), the investigated scenarios resulted in 100% valid buildings.

The reconstruction time took in total 136 s for the TU Delft case and 385 s for the Stanford case, using the AMD Threadripper 3970X processor.

4.2. Computational grids

We conducted our numerical investigations in OpenFOAM [78] version 7. We used *blockMesh* and *snappyHexMesh* utilities to create our numerical grids. Building geometries were used as they were automatically reconstructed, without manual modifications. To facilitate a grid convergence study, we constructed three distinct meshes for both cases.

The background mesh was refined through an octree-based refinement strategy, ensuring a 2:1 ratio in cell edge length between consecutive refinement levels. The increase in mesh density was targeted to adhere to BPG's [58,59,61] in areas of interest, including a minimum of ten grid cells across building heights and sides, and within passages between buildings.

The two finer meshes were obtained by isotropically refining the background mesh of the coarsest case by a factor of 2, resulting in mesh resolutions in the vicinity of buildings from 0.33 m to 0.19 m for the TU Delft case and 0.36 m to 0.2 m for the Stanford case. In total, the grid generation produced grids from 9.2 million to 35 million cells for the TU Delft case and 14 million to 55.3 million for the Stanford case. Nominal meshes are shown in Fig. 9 and the basic information on computational grids is presented in Table 3.

4.3. Numerical simulations

Simulations were conducted as steady-state with the Reynolds-Averaged Navier Stokes (RANS) approach and the standard $k - \epsilon$ turbulence model. The atmospheric boundary layer (ABL) was modelled as neutral, with the formulation:

$$U = \frac{u_*}{\kappa} \ln\left(\frac{z + z_0}{z_0}\right), \tag{4}$$

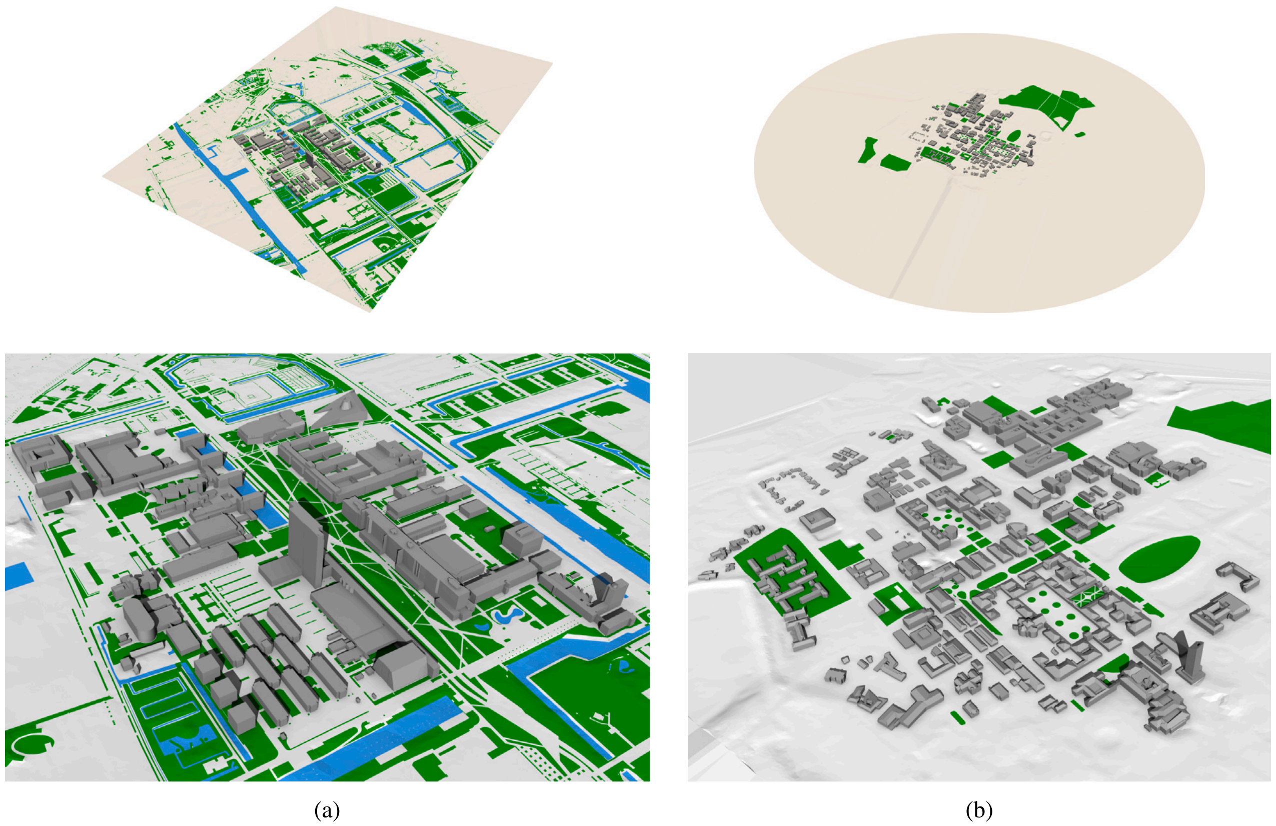

Fig. 8. Reconstructed urban scenarios of (a) TU Delft and (b) Stanford campuses. Low vegetation surfaces are coloured green, whereas water surfaces are coloured blue.

Table 3
Computational grids and first cell heights in the influence region.

		Grid size	h_0 [m]
TU Delft	Coarse	9 224 141	0.33
	Nominal	17 670 626	0.26
	Fine	35 337 584	0.19
Stanford	Coarse	13 989 641	0.36
	Nominal	27 347 383	0.28
	Fine	55 313 175	0.20

where u_* represents the friction velocity, κ is the von Karman constant with the value 0.41, z is the height above ground level (AGL) and z_0 is the aerodynamic roughness length.

Both simulated cases are located in urban areas; therefore, the inlet aerodynamic roughness length was chosen as 0.2 m, corresponding to the value of "very rough" area by the specification of Wieringa [52].

For the TU Delft case, the inlet was modelled with the 45° north direction and 4.9 m/s inlet velocity at 10 m. These conditions correspond to averaged meteorological data for the period of January to October of 2019 at the Rotterdam Station [79]. Boundary conditions included symmetry for the side and top domains, and a prescribed pressure/zero velocity gradient at the outlet.

The Stanford case utilised *inletOutlet* type boundary condition, assigning the ABL inlet for inflow and zero gradient for outflow depending on the velocity gradient. The inflow parameters were calculated using the weather station data located around 800 m east of the centre of Stanford's Science and Engineering Quad [80]. The inlet was modelled as a northwest wind with 3.06 m/s velocity at 10 m height. The time chosen to calculate the inlet velocity was between 3 am and 4 am on the 12th of October 2017, as it was reported in [80] that the minimum variability had been observed in measurements, indicating conditions close to neutral.

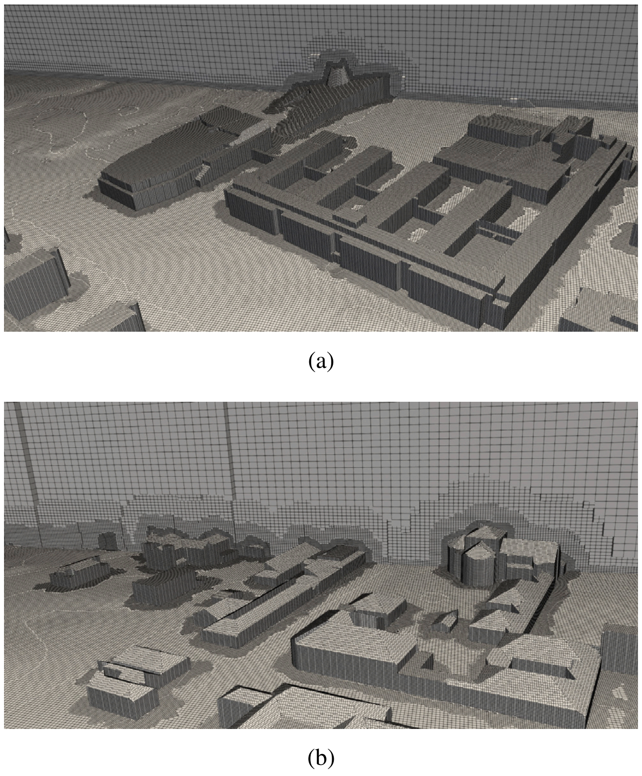

Fig. 9. Computational grids of nominal meshes for (a) TU Delft and (b) Stanford cases.

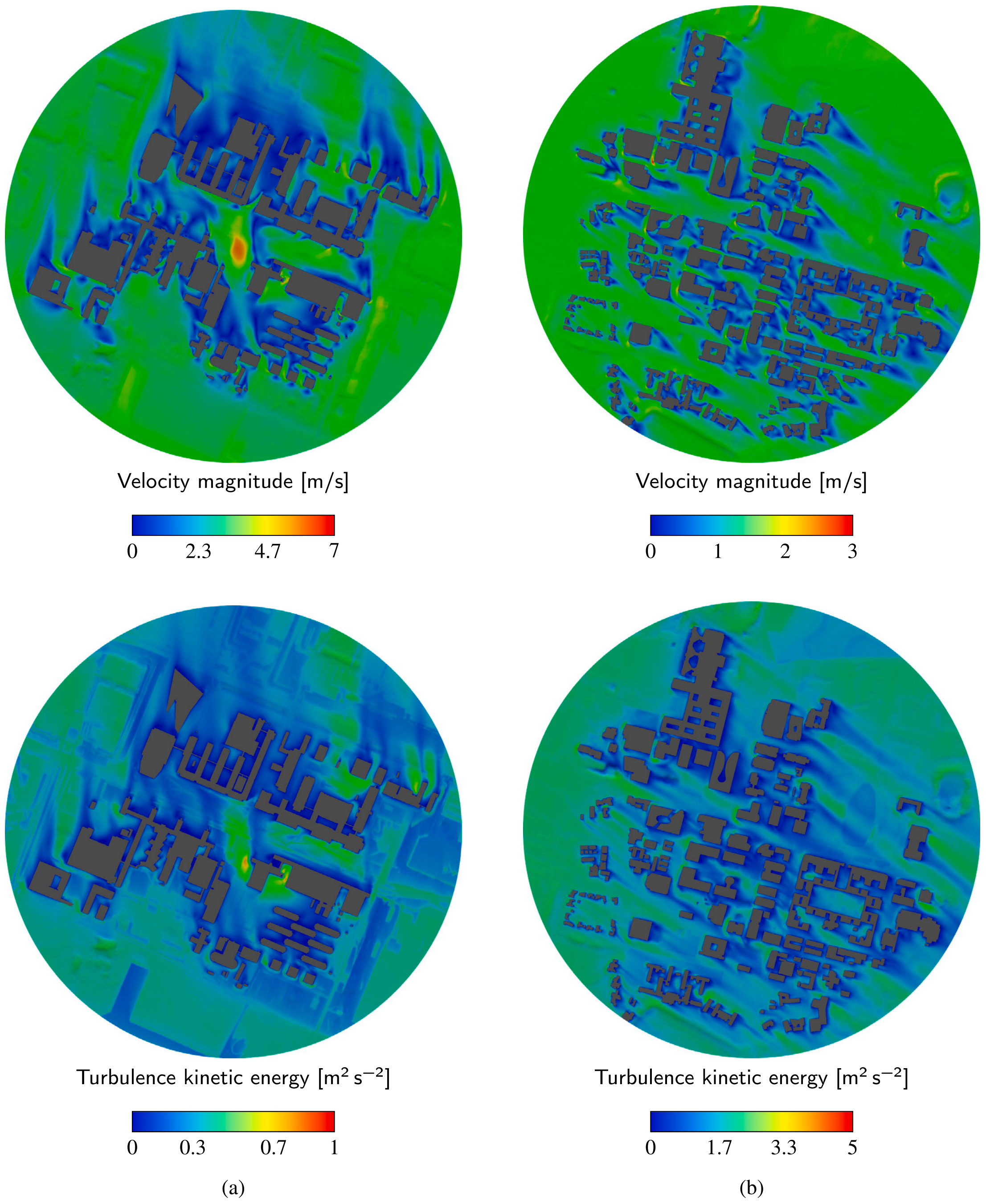

Fig. 10. Velocity magnitude and TKE fields at 2 m AGL for the (a) TU Delft and (b) Stanford cases.

We used the rough wall function based on z_0 by Parente et al. [81] to model the near-wall behaviour in the rough terrain. The water surfaces' aerodynamic roughness length was set to 0.0002 m, whereas for the low vegetation surfaces it was set to 0.03 m. The rest of the terrain, not covered by any other surface layers, was modelled with an aerodynamic roughness length with the value of 0.2 m.

We used the central differencing scheme for pressure discretisation, the linear upwind-biased scheme for the momentum variables, and the limited linear scheme for turbulence quantities.

4.3.1. Grid convergence analysis

All the simulations yielded stable and consistent outcomes. Fig. 10 illustrates the velocity magnitude and turbulence kinetic energy (TKE) fields at the 2 m AGL for both the TU Delft and Stanford cases. In the TU Delft scenario, we can see the notable hot spot located next to the high-rise building in the campus [25,31]. On the other hand, the Stanford case exhibits patterns that qualitatively match the observations reported in [80], reinforcing the validity of our simulation results.

We conducted the grid convergence study to verify the results using the method outlined in [82]. The solved variables of velocity, pressure, and turbulence quantities were measured at 32 random locations within the area of interest. Median grid convergence values were computed for all measured quantities, with the results presented in Table 4. From the grid convergence values, e_a^{21} denotes the approximate relative error between the fine and the medium mesh, e_{ext}^{21} is the extrapolated relative error, and $\mathrm{GCI}_{\mathrm{ext}}^{21}$ stands for the fine grid convergence index. The GCI values, with a maximum of 3.8% for velocity and pressure fields and up

Table 4
Approximate relative error, extrapolated relative error, and GCI values between medium and fine meshes across simulation results.

	TU Delft			Stanford		
	e_a^{21}	e_{ext}^{21}	GCI_{ext}^{21}	e_a^{21}	e_{ext}^{21}	GCI_{ext}^{21}
U_{mag}	1.4%	2.1%	2.5%	2.3%	2.3%	2.9%
p	0.48%	0.21%	0.26%	3.7%	3.0%	3.8%
k	0.84%	0.39%	0.49%	5.4%	5.9%	7.8%
ϵ	2.3%	2.1%	2.6%	4.5%	4.1%	5.1%

to 7.8% for turbulence quantities, confirm the simulations' reliability. Notably, the consistency in convergence towards asymptotic values, with relative errors below 6% across all variables, underscores the grids' suitability for conducted simulations. This finding reinforces the applicability of our automatic building reconstruction algorithm for urban flow simulations.

4.3.2. *Comparison between low and high density point clouds*

We have conducted an additional set of simulations to investigate the differences between using lower-density point clouds and using higher-density point clouds as input. To achieve that, we reconstructed the same scenarios as in Section 4.1, but with the original point cloud density for the TU Delft case (17.4 pts/m^2) and we thinned the original point cloud of Stanford (23 pts/m^2) to the same low-resolution (6.11 pts/m^2) as that of the TUDelft case.

We observed that some buildings reconstructed from lower-density point clouds lost finer details, whereas other buildings exhibited sharp directional changes in rooflines (a 'zigzag' pattern). Furthermore, one of the failed geometries in the TU Delft case could be reconstructed in LoD2.2 (denoted as 'B44'). See Fig. 11 for a few examples showing geometry differences in the TU Delft case. Finally, the resulting computational grids are nearly identical in size and quality, containing 18 million cells in the high-density TU Delft case and 27.4 million cells in the low-density Stanford case.

Fig. 12 shows the difference in simulation results between geometries created from high-density and low-density point clouds. The differences are calculated with the expression

$$U_{\mathrm{diff}} = \frac{|U_{\mathrm{HD}} - U_{\mathrm{LD}}|}{U_{\mathrm{ref}}} \tag{5}$$

for the absolute non-dimensional velocity magnitude difference, and

$$k_{\mathrm{diff}} = \frac{|k_{\mathrm{HD}} - k_{\mathrm{LD}}|}{U_{\mathrm{ref}}^2} \tag{6}$$

for the absolute non-dimensional TKE difference, where HD and LD represent field values for the cases created with high-density and low-density point clouds as an input respectively, and U_{ref} is the reference velocity of 4.97 m/s for the TU Delft case and 3.06 m/s for the Stanford case.

In Fig. 12, we show results for the TU Delft case at three distinct levels: 2 m AGL, 10 m elevation, and 20 m elevation. We notice that differences in velocity magnitude and TKE are localised around certain buildings, with maximum values being 138% of U_{ref} for the velocity magnitude and 15% of U_{ref}^2 for the TKE. As the differences are localised, the average differences do not exceed 5.3% of U_{ref} for the velocity magnitude and 0.39% of U_{ref}^2 for the TKE. The differences are smaller at the pedestrian level but tend to increase with height. This is caused by higher wind velocities but also intensified by larger differences in geometries at higher AGLs. The one building that stands out the most as the local maxima for all cases is the building 'B44' which was reconstructed in LoD1.2 in the low-density case due to a failed LoD2.2 reconstruction.

We also compared the difference between two input geometries by calculating chamfer and Hausdorff distances. The chamfer distance can be described as an average value of all the distances from a point in one set to the nearest point in the other set — see [83] for more details. On the other hand, Hausdorff distance is the maximum value of all the distances from a point in one set to the nearest point in the other set [84]. Point sets required to calculate distances were obtained by sampling building surfaces with 70 pts/m^2.

Fig. 13 shows the chamfer and Hausdorff distances for buildings in the TU Delft case. Buildings that cause visible differences in simulation results are marked red. We can see that the differences between two reconstructions are not negligible, especially in Hausdorff distances where more than half the buildings have differences larger than one meter. Moreover, 30% of buildings have Hausdorff distances over 3 m, meaning a considerable percentage of buildings can be differently reconstructed when using low-density point clouds.

In terms of how these distances translate to simulation results, we can observe that under a certain threshold, which is around 0.027 m chamfer distance and 1.17 m Hausdorff distance, no differences can be observed in simulation results. From the threshold, with the increasing distances, there is a high probability that the differences in building geometries will influence the flow around them. However, some buildings above the threshold show no observable differences in the flow field around them. We can assume that other factors, such as wind orientation, building height, and placement in the domain contribute to this outcome. Detailed investigation of such phenomena is outside of the scope of this paper, and we believe it deserves research on its own.

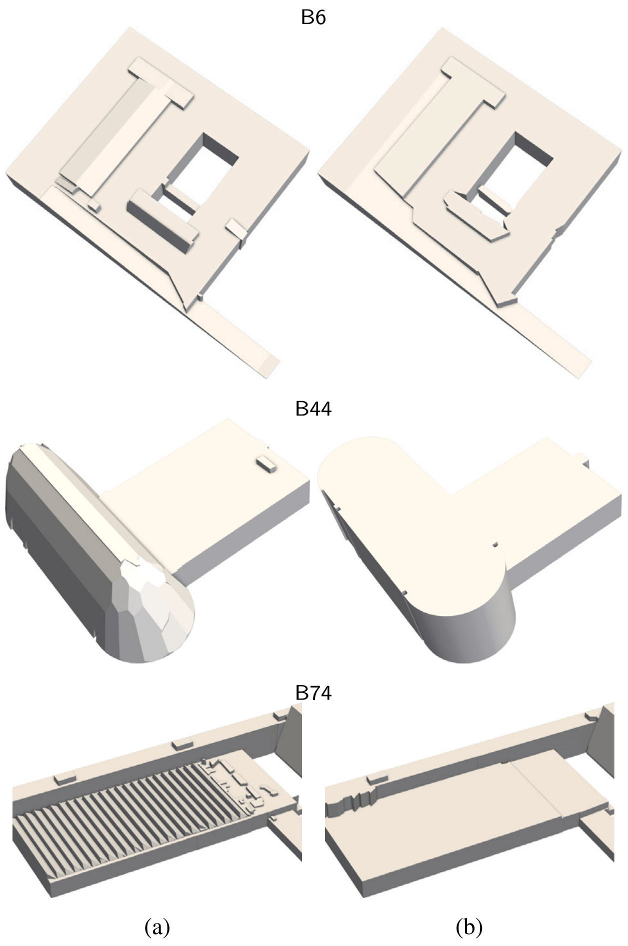

Fig. 11. Selected examples of buildings from the TU Delft scenario reconstructed in LoD2.2 using the (a) high-density point cloud (17.4 pts/m^2) and (b) low-density point cloud (6.11 pts/m^2).

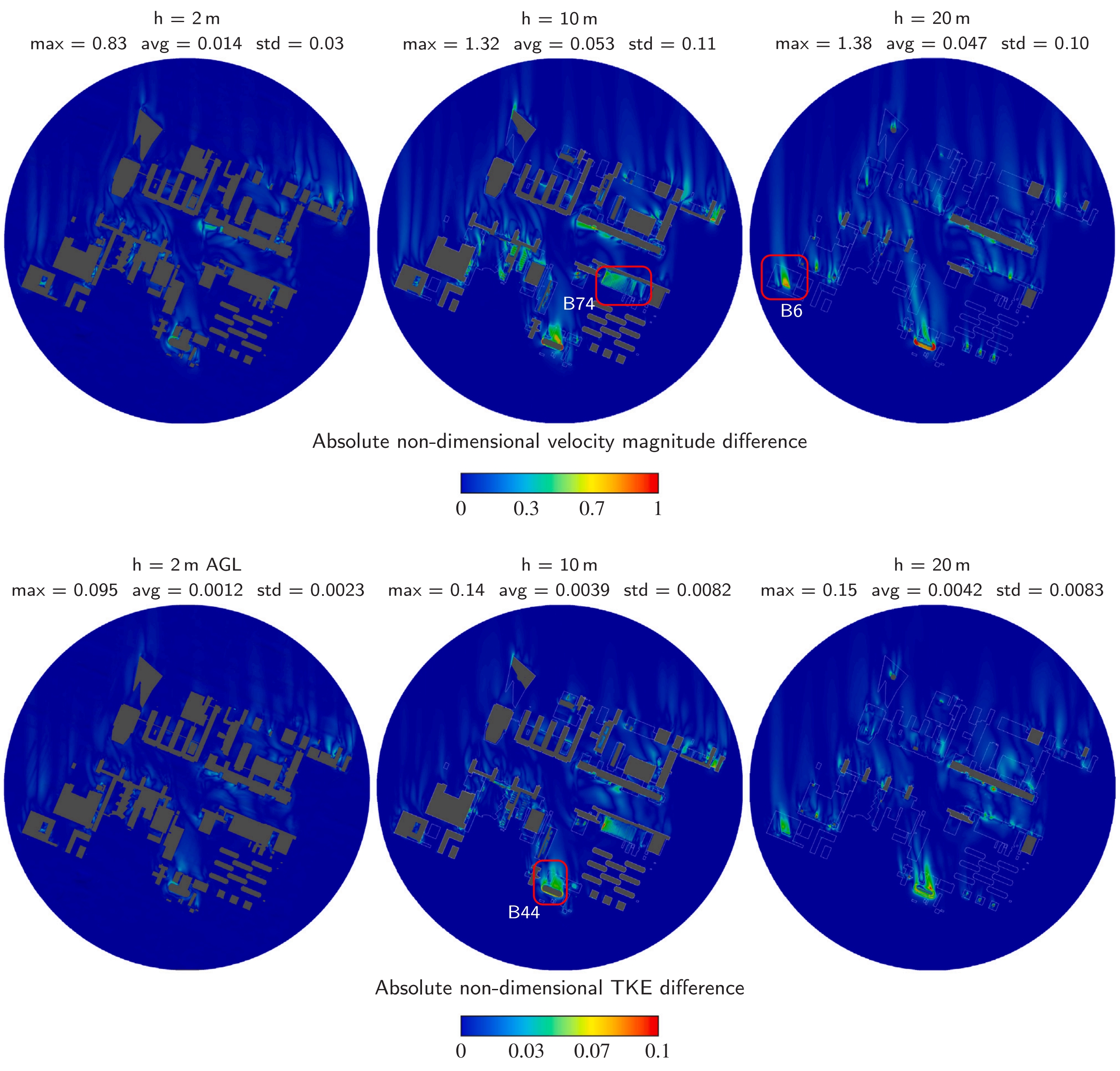

Fig. 12. Differences in simulation results between high-density and low-density input point clouds for the TU Delft case.

Similarly, the qualitative and quantitative results of the Stanford case (see Appendix, Fig. A.2) support our conclusions. Maximum and average U_{diff} and k_{diff} correspond to the TU Delft case, while in terms of geometry, results for the Stanford case (see Appendix, Fig. A.1) show a threshold of 0.0057 m for the chamfer distance and 0.75 m for the Hausdorff distance under which no differences in flow fields caused by building geometries are visible anymore.

We can conclude that the thinning done in the case of the TU Delft made an impact on the quality of the output. While the differences are not large enough to interfere with grid generation in our case, they can still cause failed reconstructions and have a localised effect on simulation results. This effect intensifies with increasing elevation due to higher wind velocities and larger discrepancies towards building tops. In other words, research aiming for results closer to the ground, such as pedestrian wind comfort studies, will be less affected by a lower-density point cloud. On the other hand, research aiming for detailed flow results at a higher AGL, such as urban air mobility, will be the most affected.

The noticeable difference between the original and thinned input point clouds means that to achieve the highest quality reconstruction, a high-density point cloud (i.e. 15 pts/m^2 or more) should be used.

5. Discussion and conclusions

In this study, we have introduced an automatic and high-detailed building reconstruction workflow for urban flow simulations. Our workflow targets applications such as pedestrian wind comfort studies, urban heat island effects, pollutant dispersion, and urban air mobility, or it can be used as a surrounding urban scenario of a very detailed and manually reconstructed building model. We have demonstrated the applicability of our workflow through the reconstruction of two realistic urban scenarios and a subsequent grid refinement study, which required no manual adjustments to the geometries. The algorithm has proved versatile, effectively processing various data sources and point cloud densities to produce LoD2.2 building models with minimal input.

The reconstructed models have achieved satisfactory quality levels, with geometric validity rates of 95% and 90% for the TU Delft and Stanford cases, respectively. Generated geometries were shown to be of satisfactory quality for the direct application with *snappyHexMesh*. However, we cannot foresee potential issues with different input datasets and different grid generators. Additionally, for more details regarding the quality assessment of generated building models on a larger sample of 10 million buildings, you can refer to Dukai et al. [40].

Furthermore, we have demonstrated that in the case of a low-density point cloud (according to the classification by Stanley and Laefer [72] our TU Delft case was low-density) we were still able to procure valid building models, generate computational grid, and successfully run a grid sensitivity analysis. However, it should be noted that a low-density point cloud (i.e. less than 15 m^2) can result in loss of building details or ultimately in failed reconstructions.

Despite its strengths, our method has limitations. First, the algorithm generates building models with 2.5D roof geometries and vertical walls. Simply put, features like balconies and overhangs cannot be reliably captured, which implies that reconstructing them is not feasible. This limitation is due to the state of the art in automatic reconstruction of 3D city models from airborne lidar. Our future efforts will investigate the reconstruction to include such features by using other datasets like terrestrial lidar and street-view images [85]. Second, the created planes are piecewise planar, so curved roof features such as domes cannot be reconstructed. However, if the point cloud density is high enough, a curved feature can be approximated by a number of small planar surfaces. Third, the output does not always guarantee valid reconstruction. We alleviate this problem by conducting a validity check [18] and automatically repairing geometries using alpha wrapping [19]. Fourth, the quality of the reconstruction is largely influenced by the input data quality (footprints, point clouds, and their alignment). Potential improvements include changes to the core parts of the algorithm (such as better planar partition regularisation [86]) or employing new techniques to solve missing data in point clouds [87].

This new versatility in simple albeit fine control over the building reconstruction process opens opportunities for practitioners to easily and quickly configure urban scenarios for urban microscale simulations that would otherwise take a very long time. As a part of our automatic reconstruction workflow *City4CFD*, one can now, for example, reconstruct the building of interest in LoD2.2 with a high complexity factor, the immediate vicinity in LoD2.2 with a lower complexity factor, the next row of buildings in LoD1.2, and have another extra region of buildings in LoD1.2 that can be used as virtual buildings. This can be done automatically using configuration settings of *City4CFD*.

We believe this method can noticeably speed up the geometry preparation step of the CFD workflow. The whole process of our reconstruction, given the input data source is known, takes around four hours in our subjective experience. On the other hand, manually reconstructing such a complex case would take days or even weeks.

Moreover, our contributions address the computer-aided design (CAD) challenges in CFD modelling of urban microclimate recognised by Mirzaei [15, p. 6] in their review paper, who wrote the following:

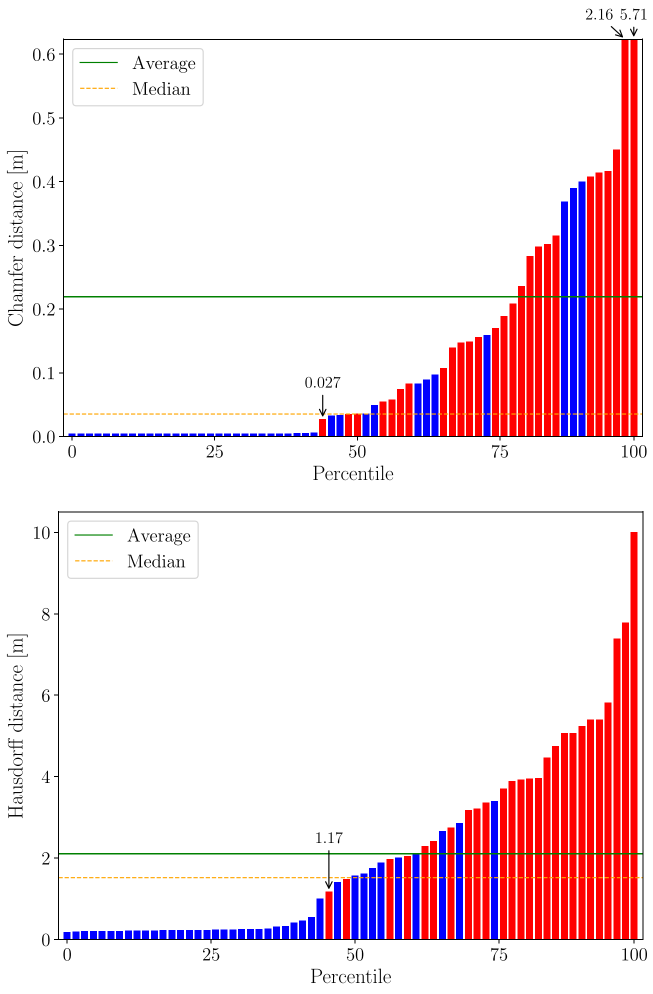

Fig. 13. Chamfer (above) and Hausdorff (below) distances between geometries generated by high-density and low-density input point clouds for the TU Delft case. Coloured red are the buildings that cause observable differences in flow fields when comparing original and thinned point clouds. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

> studies should be conducted to develop algorithms to synthesis such crude data to become suitable for CFD analysis [...] smart algorithms should make a balance between the geometry detail (flow field resolution) and mesh size (computational load). Definition of new concept such as Level of Details (LOD) [...] are handy practices and viable options in urban climate CFD simulations [...] each building, road, infrastructure, and tree has multiple surfaces that should be defined, labelled, and set as a suitable boundary condition in a CFD domain.

In response to these challenges, our work introduces the concept of LoDs in wind engineering, and our workflow balances between geometrical details, generates LoD2 geometries, and adds definition and labelling of different surfaces that can be set as suitable boundary conditions in a CFD domain. It is, to the best of our knowledge, the first work that addresses all these challenges at once.

Finally, in alignment with the principles of open-source science, we are releasing the code publicly under the GNU Affero General Public License, hoping that others would contribute and benefit from it.

CRediT authorship contribution statement

Ivan Pađen: Writing – original draft, Validation, Software, Methodology, Conceptualization. **Ravi Peters:** Writing – review & editing, Software, Methodology, Conceptualization. **Clara García-Sánchez:** Writing – review & editing, Supervision, Conceptualization. **Hugo Ledoux:** Writing – review & editing, Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Our open-source implementation of the workflow used in this study is accessible at https://github.com/tudelft3d/city4cfd. The raw data used to reconstruct the TU Delft urban scenario can be accessed at https://www.pdok.nl/datasets (datasets BAG, BGT, and AHN). For the Stanford scenario, 2D polygons were extracted from OpenStreetMap https://www.openstreetmap.org, while the USGS 3DEP lidar data can be accessed at https://apps.nationalmap.gov.

Appendix

This appendix contains supplementary figures related to Section 4.3.2 and the Stanford case. In Fig. A.2, we show slices at 2 m AGL, 43 m elevation and 53 m elevation. The latter two elevations roughly correspond to 10 m AGL and 20 m AGL, respectively.

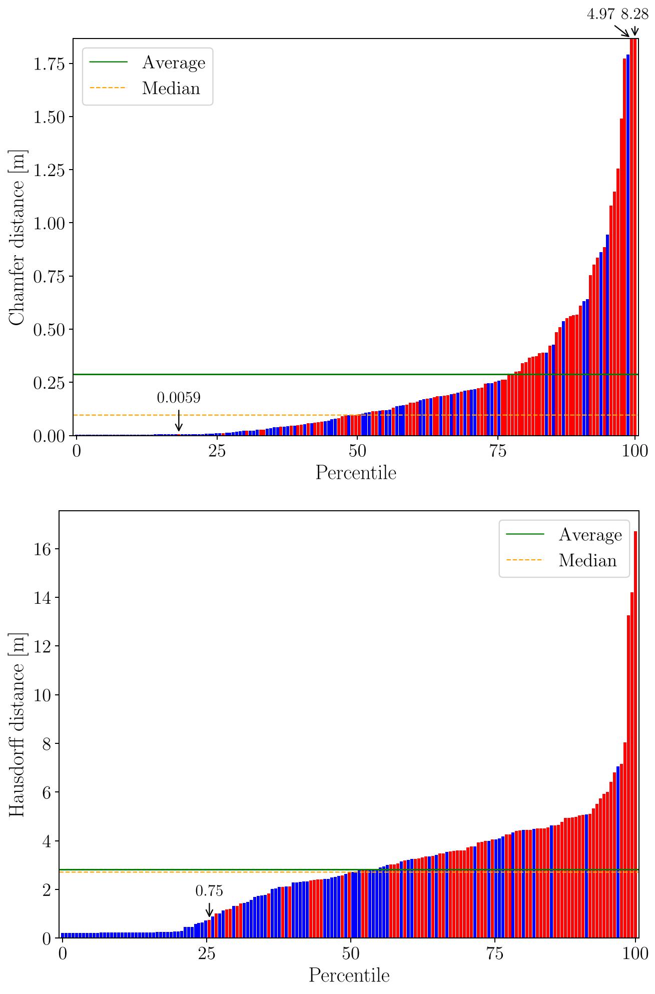

Fig. A.1. Chamfer (above) and Hausdorff (below) distances between geometries generated by high-density and low-density input point clouds for the Stanford case. Coloured red are the buildings that cause observable differences in flow fields when comparing original and thinned point clouds. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

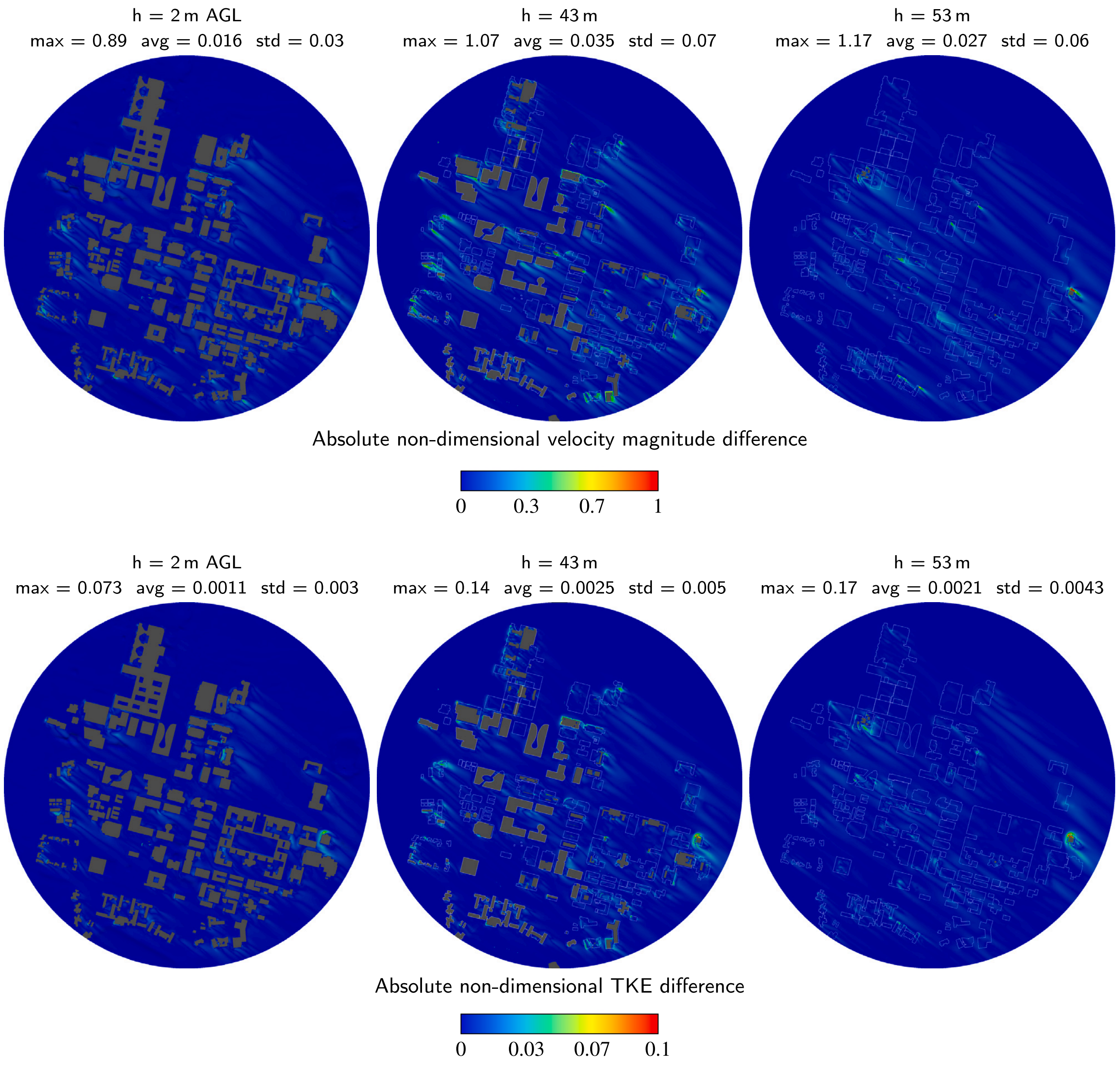
h = 2 m AGL
max = 0.89 avg = 0.016 std = 0.03
h = 43 m
max = 1.07 avg = 0.035 std = 0.07
h = 53 m
max = 1.17 avg = 0.027 std = 0.06
Absolute non-dimensional velocity magnitude difference
0 0.3 0.7 1
h = 2 m AGL
max = 0.073 avg = 0.0011 std = 0.003
h = 43 m
max = 0.14 avg = 0.0025 std = 0.005
h = 53 m
max = 0.17 avg = 0.0021 std = 0.0043
Absolute non-dimensional TKE difference
0 0.03 0.07 0.1

Fig. A.2. Differences in simulation results between high-density and low-density input point clouds for the Stanford case.

References

[1] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. Mavriplis, CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences, Technical Report NASA/CR-2014-218178, NASA, 2014.

[2] J.R. Chawner, J.F. Dannenhoffer, N.J. Taylor, Geometry, mesh generation, and the CFD 2030 vision, in: 46th AIAA Fluid Dynamics Conference, American Institute of Aeronautics and Astronautics Inc, AIAA, 2016, http://dx.doi.org/10.2514/6.2016-3485.

[3] F. Toja-Silva, C. Pregel-Hoderlein, J. Chen, On the urban geometry generalization for CFD simulation of gas dispersion from chimneys: Comparison with Gaussian plume model, J. Wind Eng. Ind. Aerodyn. 177 (2018) 1–18, http://dx.doi.org/10.1016/j.jweia.2018.04.003.

[4] T.O. Hågbo, K.E.T. Giljarhus, B.H. Hjertager, Influence of geometry acquisition method on pedestrian wind simulations, J. Wind Eng. Ind. Aerodyn. 215 (2021) http://dx.doi.org/10.1016/j.jweia.2021.104665.

[5] D.R. White, S. Saigal, S.J. Owen, Meshing complexity: Predicting meshing difficulty for single part CAD models, Eng. Comput. 21 (2005) 76–90, http://dx.doi.org/10.1007/s00366-005-0002-x.

[6] Y.S. Saeedrashed, A.C. Benim, Validation methods of geometric 3D-CityGML data for urban wind simulations, E3S Web Conf. 128 (2019) http://dx.doi.org/10.1051/e3sconf/201912810006.

[7] ISO, ISO 19107:2019 (Geographic Information: Spatial Schema), International Organization for Standardization, 2019.

[8] F. Biljecki, H. Ledoux, J. Stoter, J. Zhao, Formalisation of the level of detail in 3D city modelling, Comput. Environ. Urban Syst. 48 (2014) http://dx.doi.org/10.1016/j.compenvurbsys.2014.05.004.

[9] M. Gammon, H. Bucklow, R. Fairey, A review of common geometry issues affecting mesh generation, in: AIAA Aerospace Sciences Meeting, 2018, American Institute of Aeronautics and Astronautics Inc, AIAA, 2018, http://dx.doi.org/10.2514/6.2018-1402.

[10] J. Zheng, J. Chen, Y. Zheng, Y. Yao, S. Li, Z. Xiao, An improved local remeshing algorithm for moving boundary problems, Eng. Appl. Comput. Fluid Mech. 10 (2016) http://dx.doi.org/10.1080/19942060.2016.1174888.

[11] A. Lefieux, S. Bridio, D. Molony, M. Piccinelli, C. Chiastra, H. Samady, F. Migliavacca, A. Veneziani, Semi-automatic reconstruction of patient-specific stented coronaries based on data assimilation and computer aided design, Cardiovasc. Eng. Technol. 13 (2022) http://dx.doi.org/10.1007/s13239-021-00570-7.

[12] F. Biljecki, H. Ledoux, X. Du, J. Stoter, K.H. Soon, V.H.S. Khoo, The most common geometric and semantic errors in CityGML datasets, ISPRS Ann. Photogram. Remote Sens. Spat. Inf. Sci. IV-2/W1 (2016) 13–22, http://dx.doi.org/10.5194/isprs-annals-IV-2-W1-13-2016.

[13] F. Biljecki, J. Stoter, H. Ledoux, S. Zlatanova, A. Çöltekin, Applications of 3D city models: State of the art review, ISPRS Int. J. Geo-Inf. 4 (4) (2015) 2842–2889, http://dx.doi.org/10.3390/ijgi4042842.

[14] A. Ricci, I. Kalkman, B. Blocken, M. Burlando, A. Freda, M.P. Repetto, Local-scale forcing effects on wind flows in an urban environment: Impact of geometrical simplifications, J. Wind Eng. Ind. Aerodyn. 170 (2017) 238–255, http://dx.doi.org/10.1016/j.jweia.2017.08.001.

[15] P.A. Mirzaei, CFD modeling of micro and urban climates: Problems to be solved in the new decade, Sustainable Cities Soc. 69 (2021) http://dx.doi.org/10.1016/j.scs.2021.102839.

[16] F. Biljecki, H. Ledoux, J. Stoter, An improved LOD specification for 3D building models, Comput. Environ. Urban Syst. 59 (2016) 25–37, http://dx.doi.org/10.1016/j.compenvurbsys.2016.04.005.

[17] R. Peters, B. Dukai, S. Vitalis, J. van Liempt, J. Stoter, Automated 3D reconstruction of LoD2 and LoD1 models for all 10 million buildings of the Netherlands, Photogramm. Eng. Remote Sens. 88 (2022) http://dx.doi.org/10.14358/PERS.21-00032R2.

[18] H. Ledoux, val3dity: validation of 3D GIS primitives according to the international standards, Open Geospat. Data Softw. Stand. 3 (1) (2018) 1–12, http://dx.doi.org/10.1186/s40965-018-0043-x.

[19] C. Portaneri, M. Rouxel-Labbé, M. Hemmer, D. Cohen-Steiner, P. Alliez, Alpha wrapping with an offset, ACM Trans. Graph. 41 (4) (2022) 1–22, http://dx.doi.org/10.1145/3528223.3530152.

[20] T.O. Hågbo, K.E.T. Giljarhus, Pedestrian wind comfort assessment using computational fluid dynamics simulations with varying number of wind directions, Front. Built Environ. 8 (2022) http://dx.doi.org/10.3389/fbuil.2022.858067.

[21] P. Piroozmand, G. Mussetti, J. Allegrini, M.H. Mohammadi, E. Akrami, J. Carmeliet, Coupled CFD framework with mesoscale urban climate model: Application to microscale urban flows with weak synoptic forcing, J. Wind Eng. Ind. Aerodyn. 197 (2020) 104059, URL: https://doi.org/10.1016/j.jweia.2019.104059.

[22] F. Martín, S. Janssen, V. Rodrigues, J. Sousa, J.L. Santiago, E. Rivas, J. Stocker, R. Jackson, F. Russo, M.G. Villani, G. Tinarelli, D. Barbero, R.S. José, J.L. Pérez-Camanyo, G.S. Santos, J. Bartzis, I. Sakellaris, Z. Horváth, L. Környei, B. Liszkai, Kovács, X. Jurado, N. Reiminger, P. Thunis, C. Cuvelier, Using dispersion models at microscale to assess long-term air pollution in urban hot spots: A FAIRMODE joint intercomparison exercise for a case study in Antwerp, Sci. Total Environ. 925 (2024) http://dx.doi.org/10.1016/j.scitotenv.2024.171761.

[23] F. Yunus, D. Casalino, F. Avallone, D. Ragni, Efficient prediction of urban air mobility noise in a vertiport environment, Aerosp. Sci. Technol. 139 (2023) http://dx.doi.org/10.1016/j.ast.2023.108410.

[24] J. Hochschild, C. Gorlé, Comparison of measured and LES-predicted wind pressures on the space needle, J. Wind Eng. Ind. Aerodyn. 249 (2024) 105749, http://dx.doi.org/10.1016/j.jweia.2024.105749.

[25] C. García-Sánchez, S. Vitalis, I. Paden, J. Stoter, The impact of level of detail in 3D city models for CFD-based wind flow simuations, in: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, Vol. 46, 2021, http://dx.doi.org/10.5194/isprs-archives-XLVI-4-W4-2021-67-2021.

[26] A. Gargallo-Peiró, A. Folch, X. Roca, Representing urban geometries for unstructured mesh generation, Procedia Eng. 163 (2016) 175–185, http://dx.doi.org/10.1016/j.proeng.2016.11.044.

[27] H. Si, TetGen, a Delaunay-based quality tetrahedral mesh generator, ACM Trans. Math. Software 41 (2015) http://dx.doi.org/10.1145/2629697.

[28] F. Camelli, J.M. Lien, D. Shen, D.W. Wong, M. Rice, R. Löhner, C. Yang, Generating seamless surfaces for transport and dispersion modeling in GIS, GeoInformatica 16 (2) (2012) http://dx.doi.org/10.1007/s10707-011-0138-3.

[29] C. Sun, F. Zhang, P. Zhao, X. Zhao, Y. Huang, X. Lu, Automated simulation framework for urban wind environments based on aerial point clouds and deep learning, Remote Sens. 13 (2021) http://dx.doi.org/10.3390/rs13122383.

[30] V. Naserentin, A. Logg, Digital twins for city simulation: Automatic, efficient, and robust mesh generation for large-scale city modeling and simulation, 2022, http://dx.doi.org/10.48550/arXiv.2210.05250.

[31] I. Pađen, C. García-Sánchez, H. Ledoux, Towards automatic reconstruction of 3D city models tailored for urban flow simulations, Front. Built Environ. 8 (2022) http://dx.doi.org/10.3389/fbuil.2022.899332.

[32] T.F. Alemayehu, G.T. Bitsuamlak, Autonomous urban topology generation for urban flow modelling, Sustainable Cities Soc. 87 (2022) http://dx.doi.org/10.1016/j.scs.2022.104181.

[33] Q.Y. Zhou, U. Neumann, Complete residential urban area reconstruction from dense aerial LiDAR point clouds, Graph. Models 75 (2013) 118–125, http://dx.doi.org/10.1016/j.gmod.2012.09.001.

[34] S. Gui, R. Qin, Automated LoD-2 model reconstruction from very-high-resolution satellite-derived digital surface model and orthophoto, ISPRS J. Photogramm. Remote Sens. 181 (2021) 1–19, http://dx.doi.org/10.1016/j.isprsjprs.2021.08.025.

[35] J. Huang, J. Stoter, R. Peters, L. Nan, City3D: Large-scale building reconstruction from airborne LiDAR point clouds, Remote Sens. 14 (9) (2022) http://dx.doi.org/10.3390/rs14092254.

[36] Q.-Y. Zhou, U. Neumann, 2.5D dual contouring: A robust approach to creating building models from aerial LiDAR point clouds, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), Computer Vision – ECCV 2010, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 115–128, http://dx.doi.org/10.1007/978-3-642-15558-1_9.

[37] S. Gui, R. Qin, Y. Tang, SAT2LoD2: A software for automated LoD-2 building reconstruction from satellite-derived orthophoto and digital surface model, Int. Archiv. Photogram. Remote Sens. Spat. Inf. Sci. XLIII-B2-2022 (2022) 379–386, http://dx.doi.org/10.5194/isprs-archives-XLIII-B2-2022-379-2022.

[38] L. Nan, P. Wonka, PolyFit: Polygonal surface reconstruction from point clouds, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, http://dx.doi.org/10.1109/ICCV.2017.258.

[39] B. Dukai, J. van Liempt, R. Peters, J. Stoter, S. Vitalis, T. Wu, 3D BAG, 2021, https://3dbag.nl/. (Accessed 21 February 2024).

[40] B. Dukai, R. Peters, S. Vitalis, J.V. Liempt, J. Stoter, Quality assessment of a nationwide data set containing automatically reconstructed 3D building models, 46, International Society for Photogrammetry and Remote Sensing, 2021, pp. 17–24, http://dx.doi.org/10.5194/isprs-archives-XLVI-4-W4-2021-17-2021,

[41] F. Tarsha-Kurdi, T. Landes, P. Grussenmeyer, M. Koehl, Model-driven and data-driven approaches using LIDAR data: Analysis and comparison, in: ISPRS Workshop, Photogrammetric Image Analysis, PIA07, 2007, pp. 87–92.

[42] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, G. Yan, An easy-to-use airborne LiDAR data filtering method based on cloth simulation, Remote Sens. 8 (6) (2016) http://dx.doi.org/10.3390/rs8060501.

[43] H. Edelsbrunner, D. Kirkpatrick, R. Seidel, On the shape of a set of points in the plane, IEEE Trans. Inform. Theory 29 (4) (1983) 551–559, http://dx.doi.org/10.1109/TIT.1983.1056714.

[44] U. Ramer, An iterative procedure for the polygonal approximation of plane curves, Comput. Graph. Image Process. 1 (1972) http://dx.doi.org/10.1016/S0146-664X(72)80017-0.

[45] D.H. Douglas, T.K. Peucker, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature, Cartographica 10 (1973) http://dx.doi.org/10.3138/fm57-6770-u75u-7727.

[46] D. Muller, F. Preparata, Finding the intersection of two convex polyhedra, Theoret. Comput. Sci. 7 (2) (1978) 217–236, http://dx.doi.org/10.1016/0304-3975(78)90051-8.

[47] L. Zebedin, J. Bauer, K. Karner, H. Bischof, Fusion of feature-and area-based information for urban buildings modeling from aerial imagery, in: Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10, Springer, 2008, pp. 873–886, http://dx.doi.org/10.1007/978-3-540-88693-8_64.

[48] C.M. Gold, Point and area interpolation and the digital terrain model, in: Proceedings 2nd Annual International Symposium in Trends and Concerns of Spatial Sciences, Fredericton, NB, Canada, 1988, pp. 133–147.

[49] R. Sibson, A brief description of natural neighbour interpolation, in: V. Barnett (Ed.), Interpreting Multivariate Data, Wiley, New York, USA, 1981, pp. 21–36.

[50] H. Huang, D. Li, U. Ascher, H. Zhang, D. Cohen-Or, Consolidation of unorganized point clouds for surface reconstruction, ACM Trans. Graph. 28 (2009) http://dx.doi.org/10.1145/1618452.1618522.

[51] S. Loriot, M. Rouxel-Labbé, J. Tournois, I.O. Yaz, Polygon mesh processing, in: CGAL User and Reference Manual, five.six ed., CGAL Editorial Board, 2023, URL: https://doc.cgal.org/5.6/Manual/packages.html#PkgPolygonMeshProcessing.

[52] J. Wieringa, Updating the davenport roughness classification, J. Wind Eng. Ind. Aerodyn. 41 (1–3) (1992) 357–368, http://dx.doi.org/10.1016/0167-6105(92)90434-C.

[53] B. Blocken, W.D. Janssen, T. van Hooff, CFD simulation for pedestrian wind comfort and wind safety in urban areas: General decision framework and case study for the eindhoven university campus, Environ. Model. Softw. 30 (2012) 15–34, http://dx.doi.org/10.1016/j.envsoft.2011.11.009.

[54] Y. Toparlar, B. Blocken, P. Vos, G.J. Van Heijst, W.D. Janssen, T. van Hooff, H. Montazeri, H.J. Timmermans, CFD simulation and validation of urban microclimate: A case study for Bergpolder Zuid, Rotterdam, Build. Environ. 83 (2015) 79–90, http://dx.doi.org/10.1016/j.buildenv.2014.08.004.

[55] A. Ricci, I. Kalkman, B. Blocken, M. Burlando, M.P. Repetto, Impact of turbulence models and roughness height in 3D steady RANS simulations of wind flow in an urban environment, Build. Environ. 171 (2020) 106617, http://dx.doi.org/10.1016/j.buildenv.2019.106617.

[56] J. Brozovsky, A. Simonsen, N. Gaitani, Validation of a CFD model for the evaluation of urban microclimate at high latitudes: A case study in Trondheim, Norway, Build. Environ. 205 (2021) http://dx.doi.org/10.1016/j.buildenv.2021.108175.

[57] S. Liu, W. Pan, X. Zhao, H. Zhang, X. Cheng, Z. Long, Q. Chen, Influence of surrounding buildings on wind flow around a building predicted by CFD simulations, Build. Environ. 140 (2018) 1–10, http://dx.doi.org/10.1016/j.buildenv.2018.05.011.

[58] Y. Tominaga, A. Mochida, R. Yoshie, H. Kataoka, T. Nozu, M. Yoshikawa, T. Shirasawa, AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings, J. Wind Eng. Ind. Aerodyn. 96 (10–11) (2008) http://dx.doi.org/10.1016/j.jweia.2008.02.058.

[59] J. Franke, A. Hellsten, H. Schlünzen, B. Carissimo, Best practice guideline for the CFD simulation of flows in the urban environment, COST Action, Vol. 44, 2007.

[60] P. Kastner, T. Dogan, A cylindrical meshing methodology for annual urban computational fluid dynamics simulations, J. Build. Perform. Simul. 13 (2020) 59–68, http://dx.doi.org/10.1080/19401493.2019.1692906.

[61] B. Blocken, Computational fluid dynamics for urban physics: Importance, scales, possibilities, limitations and ten tips and tricks towards accurate and reliable simulations, Build. Environ. (2015) http://dx.doi.org/10.1016/j.buildenv.2015.02.015.

[62] T.-O. Hågbo, K.E.T. Giljarhus, Sensitivity of urban morphology and the number of CFD simulated wind directions on pedestrian wind comfort and safety assessments, Build. Environ. 253 (2024) http://dx.doi.org/10.1016/j.buildenv.2024.111310.

[63] The CGAL Project, CGAL User and Reference Manual, five.five.one ed., CGAL Editorial Board, 2022, URL: https://doc.cgal.org/5.5.1/Manual/packages.html.

[64] M. Yvinec, 2D triangulations, in: CGAL User and Reference Manual, five.five.one ed., CGAL Editorial Board, 2022, URL: https://doc.cgal.org/5.5.1/Manual/packages.html#PkgTriangulation2.

[65] H. Tangelder, A. Fabri, dD spatial searching, in: CGAL User and Reference Manual, five.five.one ed., CGAL Editorial Board, 2022, URL: https://doc.cgal.org/5.5.1/Manual/packages.html#PkgSpatialSearchingD.

[66] F. Cacciola, 2D straight skeleton and polygon offsetting, in: CGAL User and Reference Manual, five.five.one ed., CGAL Editorial Board, 2022, URL: https://doc.cgal.org/5.5.1/Manual/packages.html#PkgStraightSkeleton2.

[67] E. Fogel, O. Setter, R. Wein, G. Zucker, B. Zukerman, D. Halperin, 2D regularized boolean set-operations, in: CGAL User and Reference Manual, five.five.one ed., CGAL Editorial Board, 2022, URL: https://doc.cgal.org/5.5.1/Manual/packages.html#PkgBooleanSetOperations2.

[68] L. Dagum, R. Menon, OpenMP: an industry standard API for shared-memory programming, Comput. Sci. Eng. IEEE 5 (1) (1998) 46–55, http://dx.doi.org/10.1109/99.660313.

[69] PDOK, PDOK actueel hoogtebestand nederland (AHN) dataset, 2024, https://www.pdok.nl/introductie/-/article/actueel-hoogtebestand-nederland-ahn. (Accessed 01 February 2024).

[70] PDOK, PDOK basisregistratie adressen en gebouwen (BAG) dataset, 2024, https://www.pdok.nl/introductie/-/article/basisregistratie-adressen-en-gebouwen-ba-1. (Accessed 01 February 2024).

[71] PDOK, PDOK basisregistratie grootschalige topografie (BGT) dataset, 2024, https://www.pdok.nl/introductie/-/article/basisregistratie-grootschalige-topografie-bgt-. (Accessed 01 February 2024).

[72] M.H. Stanley, D.F. Laefer, Metrics for aerial, urban lidar point clouds, ISPRS J. Photogramm. Remote Sens. 175 (2021) 268–281, http://dx.doi.org/10.1016/j.isprsjprs.2021.01.010.

[73] OpenStreetMap contributors, Planet dump retrieved from , 2022, https://www.openstreetmap.org.

[74] U.S. Geological Survey, CA SantaClaraCounty 2020. Distributed by OpenTopography, 2021, https://portal.opentopography.org/usgsDataset?dsid=CA_SantaClaraCounty_2020. (Accessed 01 February 2024).

[75] R.E. Britter, S.R. Hanna, Flow and dispersion in urban areas, Annu. Rev. Fluid Mech. 35 (2003) 469–496, http://dx.doi.org/10.1146/annurev.fluid.35.101101.161147.

[76] J. Natanian, P. Kastner, T. Dogan, T. Auer, From energy performative to livable mediterranean cities: An annual outdoor thermal comfort and energy balance cross-climatic typological study, Energy Build. 224 (2020) 110283, http://dx.doi.org/10.1016/j.enbuild.2020.110283.

[77] G. Chen, L. Rong, G. Zhang, Unsteady-state CFD simulations on the impacts of urban geometry on outdoor thermal comfort within idealized building arrays, Sustainable Cities Soc. 74 (2021) http://dx.doi.org/10.1016/j.scs.2021.103187.

[78] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Comput. Phys. 12 (6) (1998) http://dx.doi.org/10.1063/1.168744.

[79] Koninklijk Nederlands Meteorologisch Instituut (KNMI), Climatological data from KNMI, 2019, https://www.knmi.nl/nederland-nu/klimatologie/uurgegevens. (Accessed 01 February 2023).

[80] J. Sousa, C. Gorlé, Computational urban flow predictions with Bayesian inference: Validation with field data, Build. Environ. 154 (2019) 13–22, http://dx.doi.org/10.1016/j.buildenv.2019.02.028.

[81] A. Parente, C. Gorlé, J. van Beeck, C. Benocci, Boundary layer meteorology, in: A Comprehensive Modelling Approach for the Neutral Atmospheric Boundary Layer: Consistent Inflow Conditions, Wall Function and Turbulence Model, Vol. 140, 2011, http://dx.doi.org/10.1007/s10546-011-9621-5.

[82] I.B. Celik, U. Ghia, P.J. Roache, C.J. Freitas, H. Coleman, P.E. Raad, Procedure for estimation and reporting of uncertainty due to discretization in CFD applications, Trans. ASME 130 (7) (2008) http://dx.doi.org/10.1115/1.2960953.

[83] H. Fan, H. Su, L. Guibas, A point set generation network for 3D object reconstruction from a single image, in: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Vol. 2017-January, 2017, http://dx.doi.org/10.1109/CVPR.2017.264.

[84] G. Rote, Computing the minimum hausdorff distance between two point sets on a line under translation, Inform. Process. Lett. 38 (1991) http://dx.doi.org/10.1016/0020-0190(91)90233-8.

[85] F. Biljecki, K. Ito, Street view imagery in urban analytics and GIS: A review, Landsc. Urban Plan. 215 (2021) http://dx.doi.org/10.1016/j.landurbplan.2021.104217.

[86] R. Sulzer, F. Lafarge, Concise plane arrangements for low-poly surface and volume modelling, 2024, http://dx.doi.org/10.48550/arXiv.2404.06154.

[87] Z. Chen, H. Ledoux, S. Khademi, L. Nan, Reconstructing compact building models from point clouds using deep implicit fields, ISPRS J. Photogramm. Remote Sens. 194 (2022) 58–73, http://dx.doi.org/10.1016/j.isprsjprs.2022.09.017.