frontiers | Frontiers in Built Environment

TYPE Original Research
PUBLISHED 30 August 2022
DOI 10.3389/fbuil.2022.899332

OPEN ACCESS

EDITED BY
Catherine Gorle,
Stanford University, United States

REVIEWED BY
Alessio Ricci,
Eindhoven University of Technology, Netherlands
Kavan Javanroodi,
Swiss Federal Institute of Technology Lausanne, Switzerland

*CORRESPONDENCE
Ivan Pađen,
i.paden@tudelft.nl

SPECIALTY SECTION
This article was submitted to Wind Engineering and Science, a section of the journal Frontiers in Built Environment

RECEIVED 18 March 2022
ACCEPTED 06 July 2022
PUBLISHED 30 August 2022

CITATION
Pađen I, García-Sánchez C and Ledoux H (2022), Towards automatic reconstruction of 3D city models tailored for urban flow simulations. *Front. Built Environ.* 8:899332. doi: 10.3389/fbuil.2022.899332

Towards automatic reconstruction of 3D city models tailored for urban flow simulations

Ivan Pađen*, Clara García-Sánchez and Hugo Ledoux

3D Geoinformation Research Group, Delft University of Technology, Delft, Netherlands

In the computational fluid dynamics simulation workflow, the geometry preparation step is often regarded as a tedious, time-consuming task. Many practitioners consider it one of the main bottlenecks in the simulation process. The more complex the geometry, the longer the necessary work, meaning this issue is amplified for urban flow simulations that cover large areas with complex building geometries. To address the issue of geometry preparation, we propose a workflow for automatically reconstructing simulation-ready 3D city models. The workflow combines 2D geographical datasets (e.g., cadastral data, topographic datasets) and aerial point cloud-based elevation data to reconstruct terrain, buildings, and imprint surface layers like water, low vegetation, and roads. Imprinted surface layers serve as different roughness surfaces for modeling the atmospheric boundary layer. Furthermore, the workflow is capable of automatically defining the influence region and domain size according to best practice guidelines. The resulting geometry aims to be error-free: without gaps, self-intersections, and non-manifold edges. The workflow was implemented into an open-source framework using modern, robust, and state-of-the-art libraries with the intent to be used for further developments. Our approach limits the geometry generation step to the order of hours (including input data retrieval and preparation), producing geometries that can be directly used for computational grid generation without additional preparation. The reconstruction done by the algorithm can last from a few seconds to a few minutes, depending on the size of the input data. We obtained and prepared the input data for our verification study in about 2 hours, while the reconstruction process lasted 1 minute. The unstructured computational meshes we created in an automatic mesh generator show satisfactory quality indicators and the subsequent numerical simulation exhibits good convergence behavior with the grid convergence index of observed variables less than 5%.

KEYWORDS

automatic city reconstruction, geometry preparation, pre-processing, computational fluid dynamics, semantic surfaces, 3D city modeling

1 Introduction

In 2014, in its CFD Vision 2030 study (Slotnick et al., 2014), NASA acknowledged that geometry pre-processing and grid generation are the most time-consuming part of computational fluid dynamics (CFD) workflows. Geometry preparation was labeled as one of the main bottlenecks, with authors urging for a much higher degree of automation.

Automatic geometry preparation is a wish for any CFD simulation, but it becomes of paramount importance for urban flows. First of all, the size of the domain can be very large, a few square kilometers is common. In that area, the number of complex geometrical features will also be substantial, one can think of hundreds of buildings. Furthermore, the data used for creating 3D city models usually come from different sources and in formats that are not standard in computer-aided design (CAD). This implies that the data must be converted and often modified (typically a semi-manual operation) and that in the process the user will introduce geometrical errors (e.g., overlapping buildings or sharp edges). Thus, it comes with no surprise that at the time of writing this article, urban flow simulations are still burdened with long pre-processing times (Blocken, 2021).

As further explained in Section 2, some recent developments in automatic building reconstruction show promising application in urban flows because they are scalable, produce error-free 3D geometries, and can generate buildings at different levels-of-detail (LoD). However, producing CFD-ready geometries for urban flow simulations necessitates more than automatic building reconstruction. First, we need to add the terrain to the domain, which needs to be done in a seamless interaction with buildings. This generally requires purpose-made algorithms or involves manual labor, which is why the terrain is often approximated as a flat surface (Liu et al., 2017; Dhunny et al., 2018; Toparlar et al., 2018; Zhang et al., 2021). However, in some cases, as demonstrated by Brozovsky et al. (2021), the inclusion of the real morphology of the terrain is indispensable. Second, we need multiple tools and format conversions to implicitly model ground features as roughness layers (García-Sánchez et al., 2021). Doing so on a complex terrain introduces further complications, and many applications ignore the effect of ground features. Third, we need to make engineering decisions that will critically affect the quality of our CFD solution, such as the definition of the area of interest (or influence region) (Blocken, 2015) and the outer boundary of the domain. Inside the area of interest, buildings and sometimes other features such as trees are explicitly reconstructed, while outside the area we parametrize any geometry. Works by Tominaga et al. (2008), Tong et al. (2016), Liu et al. (2018) give guidelines on the size of the area of interest, generally as a distance or number of building blocks relative to a building of interest. For the outer domain boundary, guidelines by Franke et al. (2007), Tominaga et al. (2008), Blocken (2015) prescribe the domain size relative to the highest building in the area of interest along with the maximum blockage ratio. While the geometry preparation step is more complex due to the three additional requirements identified above, these tasks are all well suited for automation.

In this paper, we address all these problems and provide an open-source tool that can automatically reconstruct 3D city models ready to run with CFD methods. This tool is the first version of an automated tool that reconstructs 3D city models at multiple levels of detail, with complete control over the reconstruction process. We first provide in Section 2 an overview of the relevant works related to geometry reconstruction, and we introduce the concepts of validity of geometries (in different fields) and of the LoD of buildings. We address in Section 3 all three above-mentioned problems through our workflow which automates the integration of buildings and terrain, automatically creates roughness surface layers, and automatically sets the area of interest and the domain size (see Figure 1 for an overview). To achieve this we use 2D geographic information systems (GIS) datasets (e.g., topographic or cadastral maps or volunteered geoinformation such as OpenStreetMap (Weber and Haklay, 2008)) and aerial point clouds (Mallet and Bretar, 2009) to reconstruct terrain, buildings, as well as to denote semantics (surfaces with attributes) later used to impose different roughness values through corresponding patches in generated computational meshes. Such datasets are nowadays readily available in many countries.

We have implemented our methodology in C++ using modern state-of-the-art libraries. The software is called *City4CFD* and its source code is available under an open-source library. In Section 4 we discuss its implementation and show experiments we ran with a test case from a real-world dataset in the Netherlands. The generated computational mesh shows satisfactory quality indicators and the subsequent numerical simulation exhibits good convergence behavior. Finally, in Section 5, we discuss the current capabilities and limitations of our code and point to future directions of research.

2 Related work

2.1 3D modeling of cities

When reconstructing 3D models of buildings, one has to decide which details and levels of abstraction are necessary. We use in this paper the level of detail (LoD) concept and classification by Biljecki et al. (2016b), Figure 2 shows the classes relevant to wind simulations. Mirzaei (2021) proposed the introduction of the same LoD classification to computational wind engineering's narrative. It should be stressed that higher LoDs are more complex to reconstruct, and also more and more accurate, data are necessary.

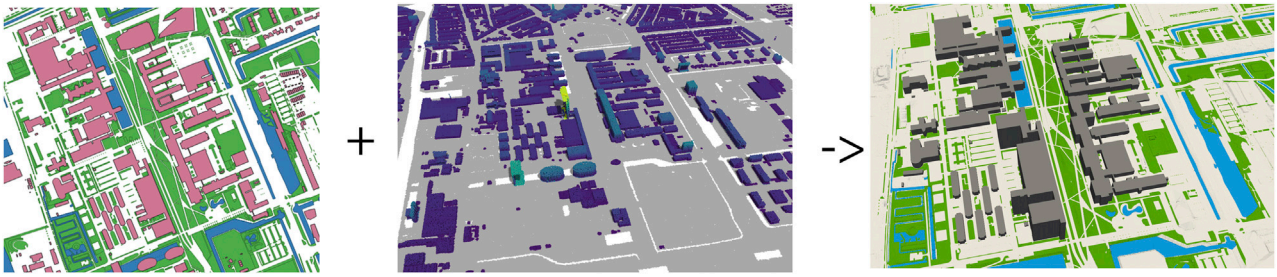

FIGURE 1
Schematic representation of the input data (2D polygons and a point cloud) and output geometry of the proposed workflow.

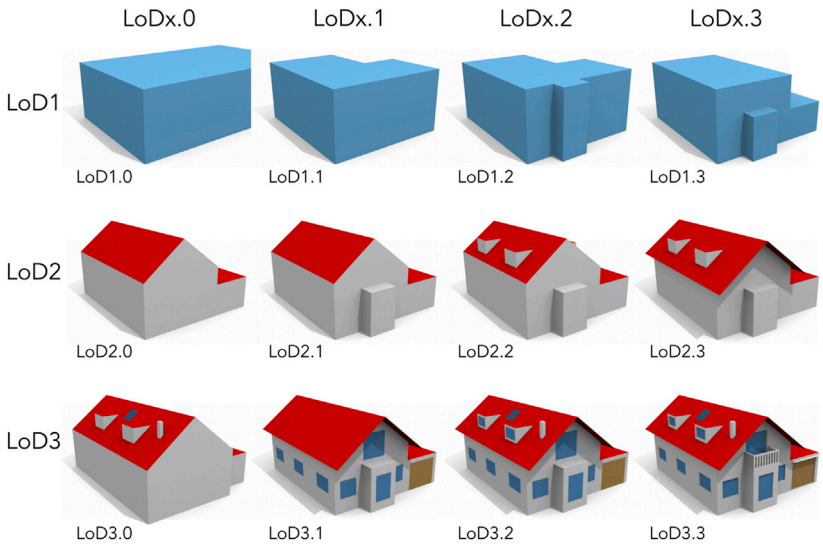

FIGURE 2
LoD definitions for buildings, adapted from Biljecki et al. (2016b).

The latest developments in automatic 3D city modeling strive towards creating high-detailed datasets which are valid according to the ISO 19107 standard (ISO, 2003), the international standard for geometries used in the GIS world. In short, individual buildings are watertight, without self-intersections, they do not contain duplicate faces, edges, vertices, or non-manifold edges or vertices—essentially, they do not require additional repair. Most recent works use the combination of point cloud elevation data and building footprints as input. Elevation data can typically be obtained from LiDAR (airborne, terrestrial) or photogrammetry. On the other hand, building footprints are common in most of the world, regularly as parts of 2D cadastral maps or volunteered geoinformation (Fan et al., 2014; Hecht et al., 2015).

An example of employing the combination of 2D topographical information and point cloud elevation for an automatic 3D city reconstruction is the LoD1.2 reconstruction framework called *3dfier* (Ledoux et al., 2021). The framework uses 2D geographical datasets and extrudes them to a height calculated from point cloud elevation data. It reconstructs buildings, but also other topographical data such as green surfaces, water, and bridges. Furthermore, it defines a set of rules on handling boundaries of features. Specifically, buildings are prismatic, water polygons are flat/horizontal, and roads are smooth surfaces.

Another example of an automated reconstruction from the two input datasets is Nys et al. (2020). The authors developed a workflow that segments roof surfaces from the airborne LiDAR

point cloud using a region-growing algorithm and creates LoD2.1 buildings from it. The authors praised the robustness of their algorithm, showing 92–96% valid buildings of tested datasets according to ISO19107 standard, checked using validator tool *val3dity* (Ledoux, 2018). However, the algorithm has yet to be tested on real-world applications, as reconstructed geometries and code are not publicly available.

Finally, Peters et al. (2021) also coupled airborne LiDAR point cloud and 2D GIS data to create an algorithm that can automatically reconstruct buildings in LoD1.2, LoD1.3, and LoD2.2. The authors successfully reconstructed 10 million buildings in the Netherlands and obtained around 98% of valid geometries (according to ISO 19107). The algorithm uses an optimization parameter that can include or exclude small features from rooftops, such as chimneys and ventilation units. The 3D BAG dataset (Dukai et al., 2021) created with this algorithm is, to our knowledge, the largest openly available LoD2 building dataset that was automatically reconstructed.

2.2 Repairing existing 3D building

Traditionally, CFD engineers obtain existing building models from a third party, mostly created for other purposes, such as visualization (Biljecki et al., 2015) or other use cases that require a lower degree of geometric, topological, and semantic consistency than computer-aided engineering (CAE) does (Piepereit et al., 2016). Most existing 3D city datasets are not perfectly valid; this means they contain errors such as missing surfaces, self-intersections, and non-manifold edges (Biljecki et al., 2016a), requiring geometry repair (also referred to as *geometry cleanup* (Simões and Estanqueiro, 2016)) that, to the best of our knowledge, is not a fully automated process (Deininger et al., 2020).

Nevertheless, noteworthy attempts have been made to reduce the necessary times for geometry preparation. For instance, Lu et al. (2011) developed a framework that generates polygon layers along with the building height, individually repairs those layers, and stitches them afterward to get the repaired geometry. The main methods used were the k-way Boolean and the Minkowski sum operations. Notwithstanding the computational efficiency and robustness of this approach, the output mesh still showed sharp features and sliver triangles, meaning some manual processing was necessary afterward.

Several other software packages that offer some degree of geometry reparation were reviewed by Saeedrashed and Benim (2019). The software packages they analyzed, specifically checking their application for urban wind simulations, were CityDoctor, FreeCAD, GeomagicWrap, MeshLab, NetFabb, and Ansys SpaceClaim. The main issues some of those packages reportedly experienced were lack of optimization for large geometries, and the introduction of new errors to geometries such as holes, gaps, and disconnections, especially in the case of self-intersections. They concluded that none of the investigated software could automatically repair geometry to the point that it could be considered simulation-ready.

2.3 Automatic methods for computational fluid dynamics-ready models

All three issues mentioned in the introduction—integrating buildings with terrain, implicitly modeling features, and setting up of the area of interest—are susceptible to automation. At this moment, *3dfier* is the only tool that comes close to solving the problems of missing terrain and roughness layers, as it reconstructs the terrain and enables watertight integration of buildings. Furthermore, it creates different surface layers such as roads, water, and vegetation-covered regions. However, the output geometry still requires manual editing, mainly due to complex modeling of boundaries between different surface types (see Ledoux et al. (2017) and Deininger et al. (2020) for two concrete examples).

Other works, such as Kawaguchi and Oguni (2016), integrate existing buildings into the terrain reconstructed as triangulated irregular network (TIN). Their workflow creates a TIN of the terrain and then imprints building footprints into the terrain. TIN triangles that are overlapping with the footprint are removed, and the terrain is re-meshed around the footprint, having footprints defined as holes. Similarly, Camelli et al. (2012) developed an approach that integrates building polygons with elevation data into a triangulated digital elevation model (DEM). The same approach of imprinting cadastral footprints into a DEM was employed by Gargallo-Peiró et al. (2016). The authors expanded the methodology by including LoD1.2 reconstruction using LiDAR elevation data.

Unfortunately, the usability of all previous frameworks can not be tested because, to the best of our knowledge, none of them are publicly available.

2.4 The influence of buildings' levels-of-detail in urban flows

As seen in the previous section, general-purpose LoD2 reconstruction algorithms are scarce. Unsurprisingly, relatively few projects have used LoD2 geometries to simulate urban flows on a larger scale, i.e., a city district or larger. An example of that is Toja-Silva et al. (2018) and Toja-Silva et al. (2017), where they used a LoD2.1 model of a district in Munich, Germany. They noted it took them a long time and a lot of manual effort to create the geometries that are adequate for a numerical simulation.

Ricci et al. (2017) published one of the few works that investigate the influence of building details (LoDs) on the wind flow field. They compared geometries of a city block reconstructed in what we would classify as LoD2.1, LoD1.3, and LoD1.1. They observed significant differences between LoD1.3 and LoD1.1, but also a satisfactory agreement between LoD2.1 and LoD1.3. The subsequent validation study on a city district level between LoD1.3 and LoD1.1 confirmed that on average the more detailed geometry offers substantially better results. The authors also noted that the time required to pre-process the LoD2.1 case took five times longer than LoD1.3 for only one building block. Even though it was just one case study, this research has shown that LoD1.1 might not be adequate to describe the airflow in the area of interest accurately, whereas LoD1.3 indicates a good trade-off between accuracy and time cost.

Hågbo et al. (2021) created detailed geometries (which we would classify as LoD2.1 by visual inspection) from photogrammetry and LiDAR point clouds. They manually processed the data, i.e. cleaned the geometries to be acceptable for wind flow simulations, and referred to the step as "time-consuming". The authors also reconstructed buildings from polygon extrusion to LoD1.2 and investigated the fourth model—an already available national dataset of buildings, in unknown LoD. The comparison of the results showed that there is a notable difference between LoD1.2 and LoD2.1 for pedestrian wind comfort simulations.

While we can see that there are a few works that addressed the influence of the LoD on urban flow simulations, there is still a need for a systematic investigation of the effect of geometry. This can be done by quantifying the geometric uncertainties. By systematic, we refer to the level of investigation comparable to uncertainty quantification of the turbulence model (Gorlé and Iaccarino, 2013), inflow conditions (García-Sánchez et al., 2014), and dispersion (García-Sánchez et al., 2017). We believe the first step towards this goal is to develop a tool that has control over the reconstruction process.

2.5 Modeling roughness layers

A very common way to parameterize features in the computational domain is through the application of aerodynamic roughness length and sand-grain roughness height (Parente et al. (2011); Blocken et al. (2012); Toparlar et al. (2015); Liu et al. (2018); Ricci et al. (2020), and many others). Roughness parameters are applied in the area of influence to account for smaller features that are not explicitly modeled (benches, sidewalks, poles, façades), as well as to model the region between boundaries and the area of interest. As indicated by Blocken and Persoon (2009), Ricci et al. (2017), and García-Sánchez et al. (2021), the influence of roughness zones on the local flow field is noticeable, especially close to the ground, which is of paramount importance for pedestrian comfort.

Most examples, including those already mentioned, integrate different surface roughnesses into a flat terrain. Very few expand this to non-flat terrains, two examples being Piroozmand et al. (2020) and Brozovsky et al. (2021).

While there are works that deal with automatic roughness estimation for atmospheric flows on larger scales, such as mesoscale (Macdonald et al., 1998), we were not able to find information on the automated reconstruction of roughness surfaces in urban flow simulations. What we observe is that the data used to determine roughness height values generally comes from satellite imaging (Liu et al., 2018; Ricci and Blocken, 2020) or 2D GIS datasets (Brozovsky et al., 2021; García-Sánchez et al., 2021). The lack of automation (or information thereof) in handling different surface layers motivated us to create a workflow that automatically imprints 2D GIS topographical information into a non-flat terrain—a method we describe in Section 3.3.

3 Methodology

In this section, we provide an overview of our automatic reconstruction and preparation methodology for obtaining CFD-ready 3D city models at different LoDs. We give a detailed overview of every code feature, our engineering decisions, and the reasoning behind them.

Our methodology can be divided into four steps:

1) Building reconstruction and/or import of existing building models,
2) Terrain triangulation,
3) Imprinting of surface layers,
4) Definition of the area of interest and domain boundaries.

3.1 Building reconstruction and/or import

As seen in Section 2, elevation data from point clouds obtained by LiDAR or photogrammetry are typical input data for automatic building reconstruction algorithms. However, point clouds are not always available as they are expensive and time-consuming to acquire (Biljecki et al., 2017). Additionally, as their capturing, processing, and subsequent update takes a span of a few years, datasets can become obsolete. On the other hand, 2D building datasets (i.e., footprints) are widely available as part of national cadastral and volunteer databases, but also fairly easy to keep up-to-date compared to point clouds (Biljecki et al., 2017). What is useful for urban flow applications is that many 2D building datasets accommodate attributes such as explicitly defined height (Gao et al., 2018), or the number of floors (Guney et al., 2012;

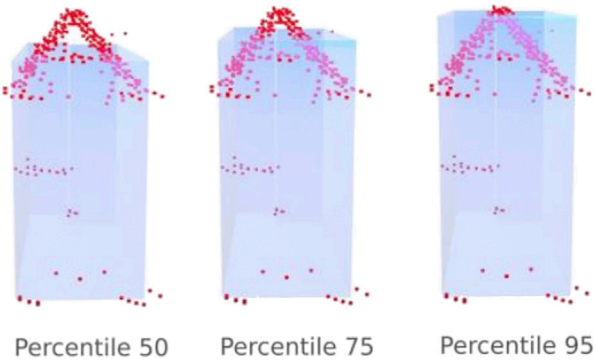

FIGURE 3
Calculated building height depending on percentiles and points belonging to that building, figure adapted from Dukai et al. (2019).

Agugiaro, 2016) which can be used to derive a building height (Zhang et al., 2021).

In addition, CFD simulations are often used to investigate the environmental impact of new buildings (Toja-Silva et al., 2016; Thordal et al., 2020), modifications of buildings (Guo et al., 2015), or detailed investigations of existing buildings (Blocken and Persoon, 2009), meaning that the integration of additional geometries, like projected developments, into existing datasets is essential for the urban flow community.

Therefore, in our methodology, we combine three methods that CFD could benefit from:

1) 3D reconstruction from footprint polygons and point cloud,
2) 3D reconstruction from footprint polygons with attributes, such as the heights or number of floors,
3) Importing pre-reconstructed geometries.

3.1.1. 3D reconstruction using footprint polygons and point cloud

For the first method, we implemented LoD1.2 reconstruction, i.e. footprint extrusion. The height of a building is calculated as a certain percentile of all points falling with the footprint polygon, e.g. 50, 75, or 90 percentile as shown in Figure 3.

As we saw in Section 2.1, automatic LoD2 (and also LoD1.3) reconstruction algorithms are still in the development phase and we did not implement any at this point. However, some of the above-mentioned reconstruction algorithms are publications in progress and we expect them to become publicly available in the near future. We aim here to provide a framework that enables quick adaptation of said algorithms to urban flows simulations, therefore incorporating the capability to reconstruct up to LoD2 within this code relatively soon.

3.1.2. 3D reconstruction from footprint polygons with attributes

The second method enables us to use either the height attribute or the number of floors to create LoD1.2 geometry (with extrusion).

3.1.3. Importing pre-reconstructed geometries

The third option includes importing geometries stored in either Wavefront OBJ, STL, or CityJSON (Ledoux et al., 2019). CityJSON is an international standard of the *Open Geospatial Consortium* for representing 3D city models, it can store geometries, semantics (what individual objects are, and also what each surface represents, e.g., the ground, the roof, a window, etc.), and attributes in the same file.

With the first two methods, footprints of buildings are seamlessly imprinted into the terrain using a method described in Section 3.3. We assume that 2D building footprints are valid, implying that individual polygons are valid according to ISO 19107 (ISO, 2003), as well as they do not overlap with each other. In case there is an overlap between an imported building and a 2D footprint, one of the two is left out, according to user-defined priority.

Our workflow allows for the combination of individual methods. For example, should the point cloud-based reconstruction fail due to missing points, the reconstruction falls back to utilizing the height attribute.

3.2 Terrain triangulation

The terrain is created as a triangulated irregular network (TIN) with a Delaunay triangulation using elevation points. This means the terrain is a 2.5D surface, i.e., every x-y coordinate has one height. Consequently, all triangulation operations can be conducted in 2D, with vertices lifted to their height afterward. This makes the terrain triangulation fast, robust, and suitable for polygon imprinting, described in the next section.

Elevation points can be obtained directly from a point cloud, but also converted from another GIS data model such as raster. Points can be randomly thinned to increase triangulation performance. While Delaunay triangulation is not a prerequisite for TIN, we use it to avoid long and skinny edges, which is again advantageous for the ensuing step of polygon imprinting.

3.3 Imprinting surface polygons

The polygon imprinting step allows for seamless integration of different surfaces into the terrain, as well as defining the ground polygon for building reconstruction. Implicitly modeling features with a roughness height has become an indispensable part of urban flow simulations, as demonstrated by works of

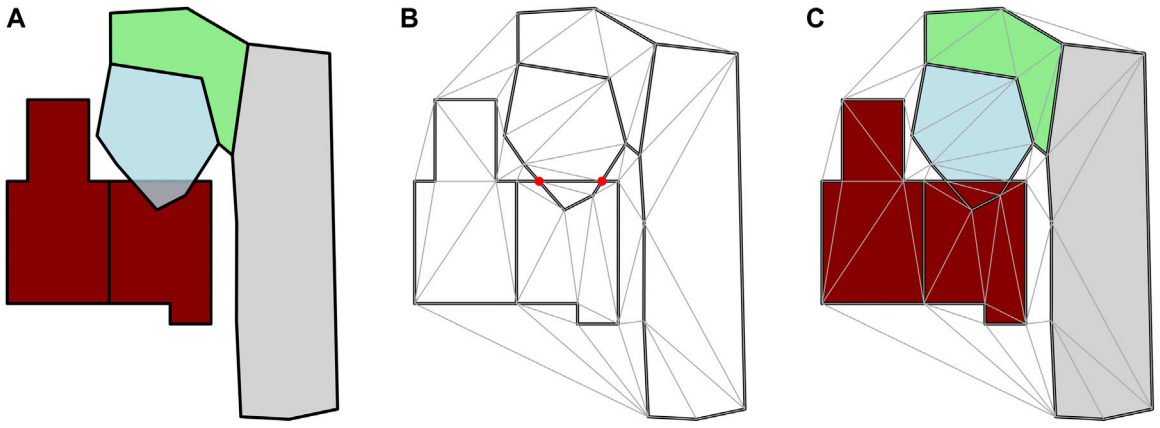

FIGURE 4
(A) 5 polygons: 2 buildings, 1 pond, 1 grass field, and 1 road. **(B)** Resulting CDT. Observe that 2 polygons are overlapping (pond and one building), their edges were split at the intersections and new vertices added (in red). **(C)** Triangulation with labels from the input polygons (with overlap fixed).

Blocken et al. (2012); Toparlar et al. (2015); Ricci and Blocken (2020); Brozovsky et al. (2021), and many others. Different types of surfaces, such as vegetation-covered areas, roads, or water are an integral part of GIS datasets and can be utilized to designate roughness parameters. We have previously used 2D GIS dataset to define roughness regions on a flat surface in (García-Sánchez et al., 2021), but the process was carried out manually. Here we present the method to automatically define multiple surface roughnesses using 2D GIS data on non-flat terrain. Our method uses constrained Delaunay triangulation (CDT) as the supporting structure, as it was used successfully to handle and repair invalid GIS datasets (Arroyo Ohori et al., 2012). The method is as follows:

1) Polygons edges are shortened to a predefined length to avoid abrupt jumps when lifting the terrain,
2) Heights of polygon vertices are interpolated from the terrain,
3) Input polygon segments are added as constraints to the terrain triangulation,
4) Every triangle in the CDT is labeled with a polygon set identifier,
5) Polygon sets are exported as separate files.

The process of creating the CDT and subsequent labeling is shown in Figure 4 for a simple case.

3.3.1 Height interpolation

As polygons are imprinted into the 2.5D terrain triangulation, the elevations of polygon vertices must be interpolated. We use the natural neighbor interpolation (NNI) to estimate the elevation (Sibson, 1981; Gold, 1988). In building polygons, those interpolated values represent ground points for reconstruction.

Alternatively, the height of polygon vertices and the height of all triangulation vertices bounded by the polygon can be averaged to a certain elevation. This method is useful for specific situations when the point density belonging to the polygon is excessively low, as is often the case with water (Mallet and Bretar, 2009). Figure 5 shows the example of a case before and after averaging.

3.3.2 Adding constraints

There are several advantages to using constrained triangulation for polygon partitioning (Arroyo Ohori et al., 2012). First, it is a planar partition by definition, so it has no trouble handling overlaps and adjacency. In case of overlaps, additional points are added on intersections, and two polygons form a new constrained region shared between them (Figure 4B). Overlapping is finally resolved in the triangle labeling step, described in Section 3.3.3. Adjacent segments and points are handled swiftly with the CDT, i.e., they are not added multiple times. Second, changes to the triangulation are local, making the triangulation fast. Third, several robust and maintained triangulation libraries are available (as is the one we use, see Section 4.1), therefore the approach is interesting from an implementation standpoint.

The constrained triangulation phase is where the effect of the first step (i.e. polygon edge shortening) becomes apparent. 2D polygons can be hundreds of meters long—without additional vertices to interpolate in between, constrained edges would create sharp triangles, as shown in Figure 6. One could argue that we would benefit from the conforming Delaunay triangulation because it creates additional points on constrained edges to adhere to the Delaunay criterion (Shewchuk, 1997). However, we aimed for the user to have more control over the constraining process, thus polygon edges can be shortened to a predefined length, and the triangulation is performed as CDT.

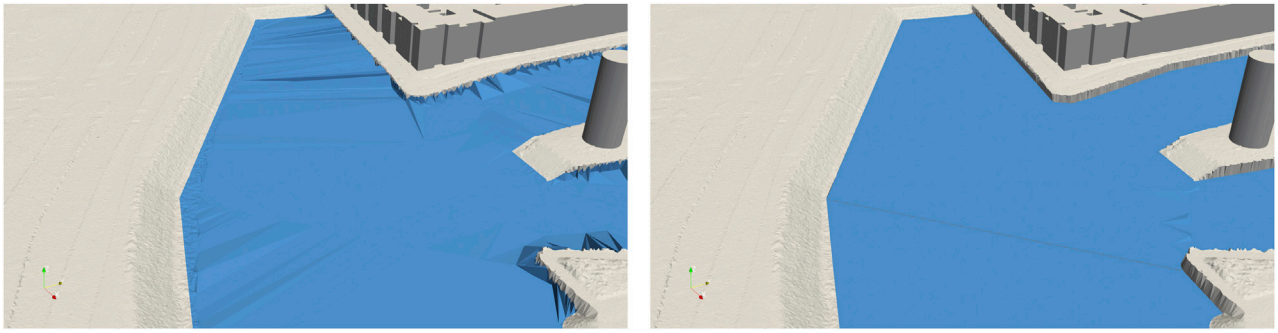

FIGURE 5
Water polygons before (left) and after (right) averaging.

3.3.3 Semantic labeling of triangles

For the semantic labeling step, we exploit the data on constrained edges of the CT. The algorithm first checks whether the centroid of the triangle adjacent to a constrained edge belongs to any polygon. Once the information on polygon ID (or no polygon) is retrieved, the algorithm proceeds to visit and mark triangles that are within the constrained region (those that do not cross the constrained edge) with a graph-based algorithm (i.e., dept-first search).

The label for every constrained region is given on a first-come-first-served basis. This explains how potential overlaps are handled. Our method of labeling triangles from constrained triangulation is similar to the polygon repair step of the workflow by Arroyo Ohori et al. (2012). The exception to our method is overlapping between two or more building polygons. Since overlapping building polygons require modifications to buildings in 3D, we plan to handle those issues in the future, as a part of a broader building generalization process.

After the labeling step, the resulting terrain ends up being divided into non-overlapping regions based on polygon sets. Those regions can now be used to mark different patches in a computational mesh. The roughness parameters are left for a user to specify because they vary between CFD software (Blocken et al., 2007).

3.4 Area of interest and domain boundaries

3.4.1 Area of interest

The last feature of our method is the modeling of the area of interest and the definition of domain boundaries. As it is ineffective to model all buildings in the computational domain, an engineering decision needs to be made to explicitly model a certain group of buildings, while others are parameterized in some way. Our framework gives an option to define the area of interest in three different ways:

1) Manually, using a prescribed point of interest and a radius,
2) Manually, by defining the area of interest with a polygon,
3) Automatically, employing the best practice guideline (BPG) by Liu et al. (2018).

FIGURE 6
Terrain before (left) and after (right) shortening of polygon edges.

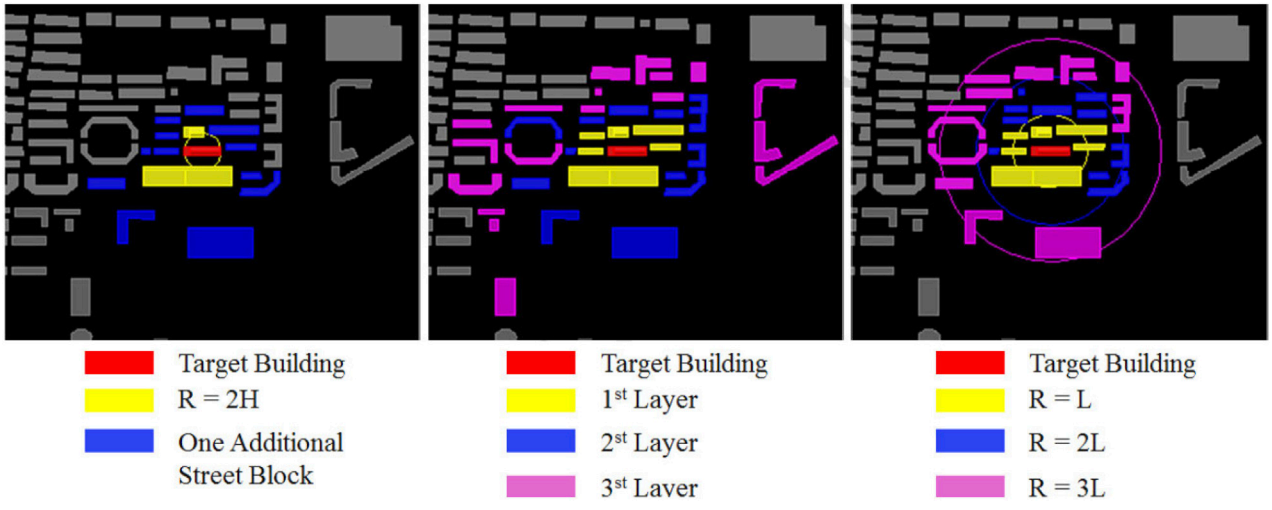

FIGURE 7
The definition of the influence zone by Tominaga et al. (2008) (left), Tong et al. (2016) (middle) and Liu et al. (2018) (right), from Liu et al. (2018).

The first two options are self-explanatory, thus we will focus on the automatic method. In the Introduction, we mentioned three common BPG's in the area of interest. Tominaga et al. (2008) suggest that the area of interest should have a radius as large as the height of the building of interest, plus one building block. Guidelines by Tong et al. (2016) define the area of interest with a number of building blocks around a building/point of interest. Our opinion is that those two methods are not suitable for automation, since building blocks are not clearly divided in some urban environments. Liu et al. (2018) defines the area of interest as a multiplier of the largest building dimension (as opposed to building height from Tominaga et al. (2008)). The authors propose the factor of three; we use this factor in our framework as well. All three guidelines are shown in Figure 7.

Our algorithm initially reconstructs the building of interest, designated with a point bounded by the building footprint. Afterward, it determines which buildings belong to the area of interest and which do not. The ones in the influence region are reconstructed, whereas the ones outside can be imprinted as polygons (and given roughness value afterward), or they can be ignored.

3.4.2 Domain boundaries

Defining domain boundaries is another step that requires engineering decisions. Our framework enables the same approaches to define domain boundary as the area of interest. Two manual approaches (point and radius, polygon) and one automatic using BPG's. The BPG's we implemented are typically recommended across multiple BPG-related works (Franke et al., 2004, 2007; Tominaga et al., 2008; Blocken, 2015). They consist of a combination of distance from explicitly modeled buildings and the maximum allowed blockage ratio. The former guideline uses the information on the highest building in the area of interest and extends the domain away from the furthest building in each direction, depending on the type of domain (Figure 8A). We added three domain types—rectangular, round, and oval. The three domain types are shown in Figure 9.

The second guideline projects building edges onto a plane normal to the free stream vector, constructs the CDT using projected edges as constraints, marks constrained regions, and divides the area of the constrained region by the domain area normal to the free stream vector. The guideline is schematically shown in Figure 8B. If the resulting blockage ratio is smaller than the prescribed percentage, the domain is proportionally expanded to fulfill the condition. We can recommend using commonly prescribed values (Figure 8), but we leave the option to change the default values as well.

The last step in domain setup is the creation of buffer zones between the terrain and domain boundaries. Ground surface near the domain edge should be free from perturbations to avoid numerical stability issues derived from zero-gradient requirements (An et al., 2020). Therefore, we included buffer zones in our framework that can extend either inwards or outwards, replacing the terrain with a flat surface. Figure 10 shows an example of a buffer zone.

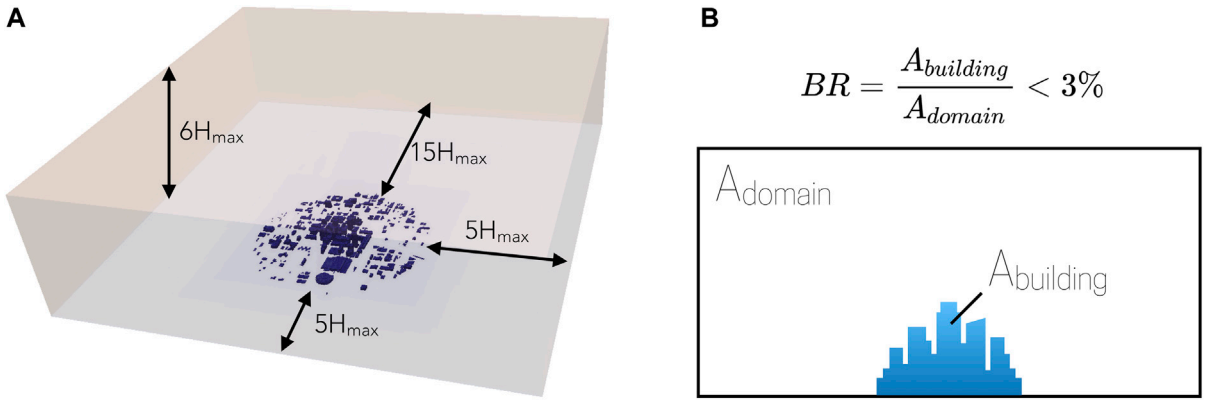

FIGURE 8
Typical guidelines for a domain size, adapted from Blocken (2015).

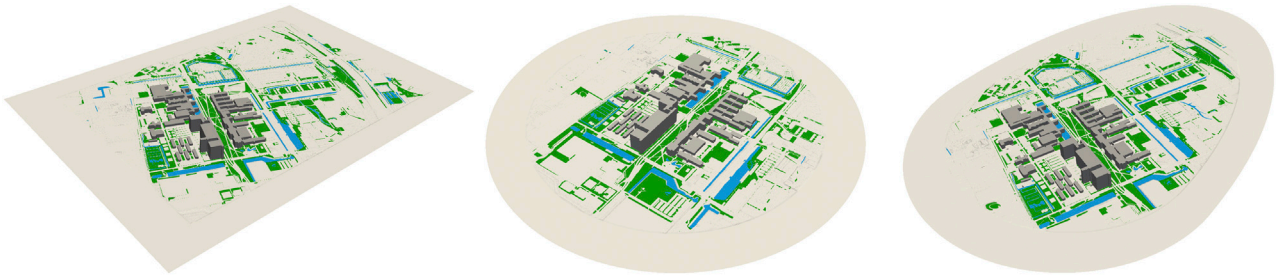

FIGURE 9
Different types of domain calculated with BPG's: rectangular (left), round (middle), and oval (right).

4 Implementation and test case

4.1 Implementation

The workflow presented in Section 3 was implemented in C++, using the Computational Geometry Algorithms Library (CGAL[1]) for low-level functionalities such as triangulations, polygon operations, and mesh processing.

Our prototype framework, which we named *City4CFD*, is open-source under GNU GPLv3 licence[2] and can be freely accessed through the link provided at the end of this paper. It uses the following formats for input:

- GeoJSON for 2D polygons[3],
- LAS, PLY, and XYZ formats for point clouds,
- Wavefront OBJ, STL, and CityJSON for importing existing building geometries.

The output geometries can be exported in Wavefront OBJ, STL, and CityJSON formats. We have tested our prototype on publicly available datasets in the Netherlands and we show the results in the following section.

4.2 Geometry test case

We tested our implementation on the campus of the Delft University of Technology (TUD) (Figure 11). For the locations we used the following datasets:

- AHN3 for LiDAR-based point cloud,
- BAG for building footprints,
- BGT for topographic layers of water and green areas (lawns, meadows).

1 CGAL, https://www.cgal.org/.
2 GNU GPLv3 license, https://www.gnu.org/licenses/gpl-3.0.html.
3 http://geojson.org.

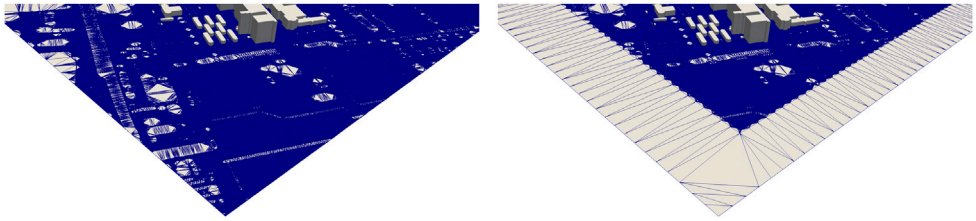

FIGURE 10
Adding a buffer zone to the edge of domain.

All three datasets are openly available on the Dutch central distribution platform for deploying geographical datasets (PDOK, 2022). Some basic information about the test case is given in Table 1.

The TUD campus location was reconstructed with a rectangular domain according to BPG's, using a predefined polygon to define area of interest. The resulting geometry (shown in Figure 12) is 1.86 km wide and 2.54 km long. A total of 86 buildings were reconstructed, with an execution time of 60 s on an AMD Ryzen Threadripper 3970X workstation.

Table 1 indicates the number of failed reconstructions. Building reconstruction can fail due to lack of point cloud points belonging to a building, or in case the calculated height is lower than a prescribed minimum which was 1.5 m in this

FIGURE 11
Google Earth imagery (Google, 2022) of the TUD Campus.

TABLE 1 Test case geometry and reconstruction information.

Case	TUD Campus
Size [km^2]	4.72
# points	2 million
# buildings	86
# other polygons	4,426
# failed reconstructions	7
Reconstruction time [s]	60

case—less than the targeted mesh size. In our case, the point cloud was acquired around 6 years ago, and those buildings have been built since then. Figure 13 shows an example of building footprints that are unable to be reconstructed due to lack of any elevation points belonging to that building. Our framework can alleviate the issue by attaching height attributes to building footprints, as described in Section 3.1.

The LoD1.2 reconstruction is supposed to create 100% valid buildings by design. We verified the implementation with *val3dity* (Ledoux, 2018) and got the perfectly valid score.

4.3 Computational fluid dynamics simulation

We used *snappyHexMesh*, an automatic unstructured hex-dominated finite volume mesh generator, to create the computational domains for simulations. It is important to highlight that we used the reconstructed geometry *as is*—meaning we have not done any modifications to the geometry whatsoever following the reconstruction by our workflow. We made three computational grids: coarse with 14 million, medium with 20 million, and fine with 31.2 million control volumes. All three meshes have the maximum non-orthogonality less than 70° and maximum skewness less than 4. Figure 14 visualizes the medium mesh.

Simulations were run in an open-source CFD package *OpenFOAM* (Weller et al., 1998) with the *simpleFoam* solver, i.e., as viscous, incompressible, and turbulent. We used the Reynolds-averaged Navier-Stokes (RANS) approach with the

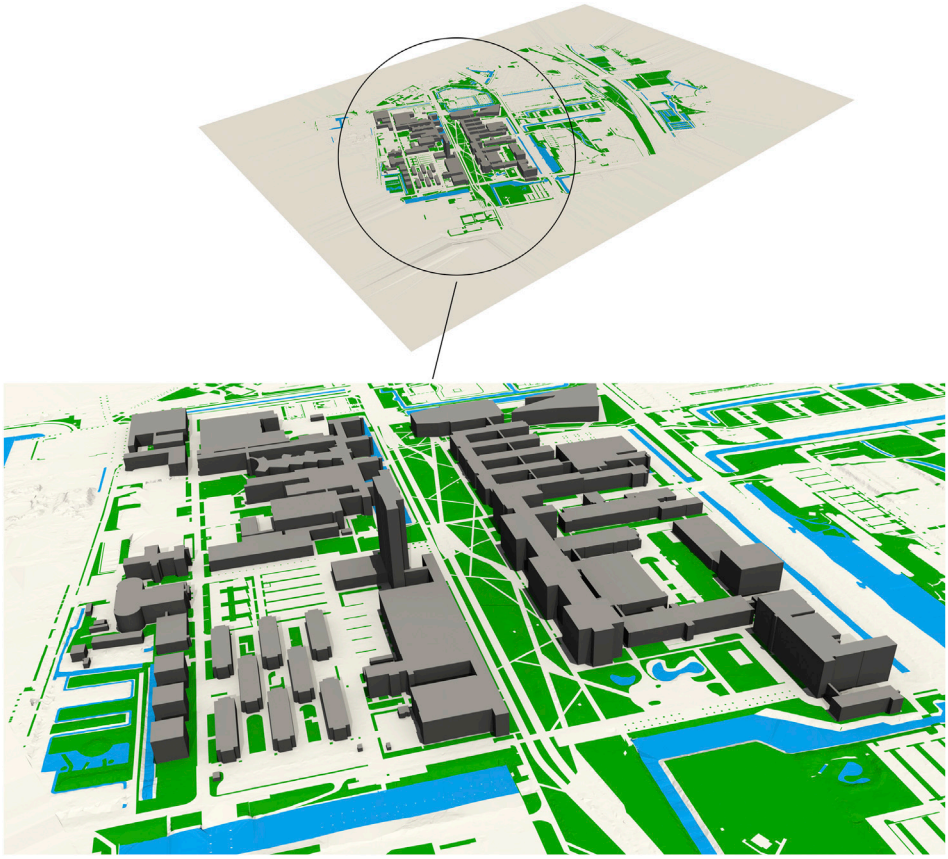

FIGURE 12
Reconstructed geometry of TUD campus with water and vegetation surface layers.

FIGURE 13
Failed reconstructions (in red) due to lack of elevation data.

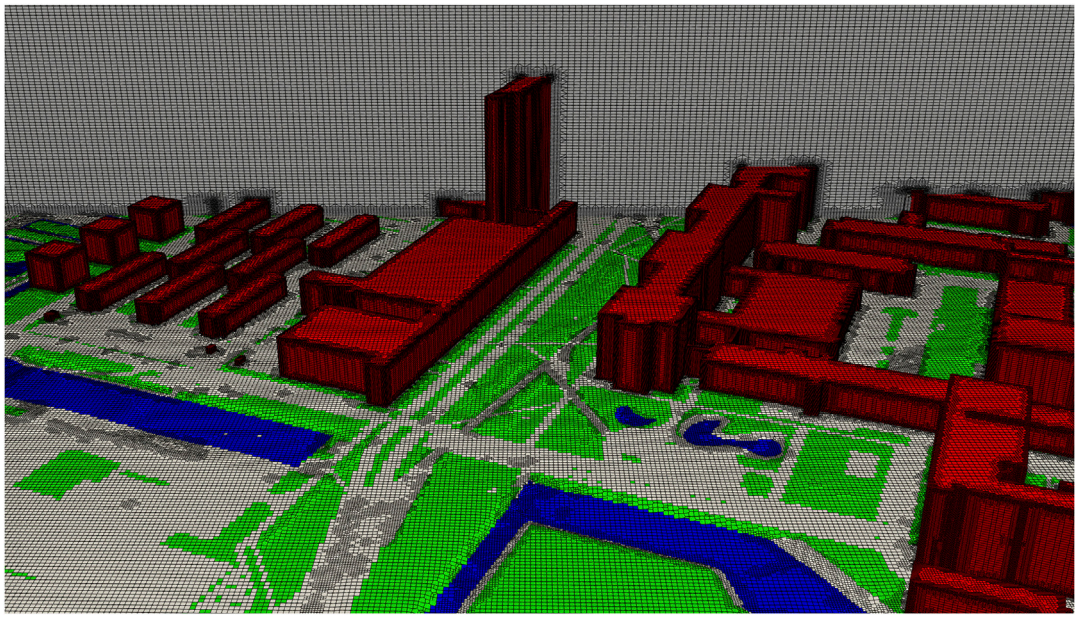

FIGURE 14
Computational mesh (medium) of the TUD campus. Individual patches are shown in different colors.

standard $k-\epsilon$ model to describe turbulence. We used the same boundary conditions as in García-Sánchez et al. (2021), more specifically, neutral atmospheric boundary layer (ABL) wind flow of 45° north direction with 4.9 m/s inlet velocity at 10 m, with the wind profile calculated according to the following formulation:

$$U = \frac{u_*}{\kappa}\ln\left(\frac{z+z_0}{z_0}\right), \tag{1}$$

where u_* is the friction velocity, κ is the von Karman constant with the value 0.41, z is the height above ground

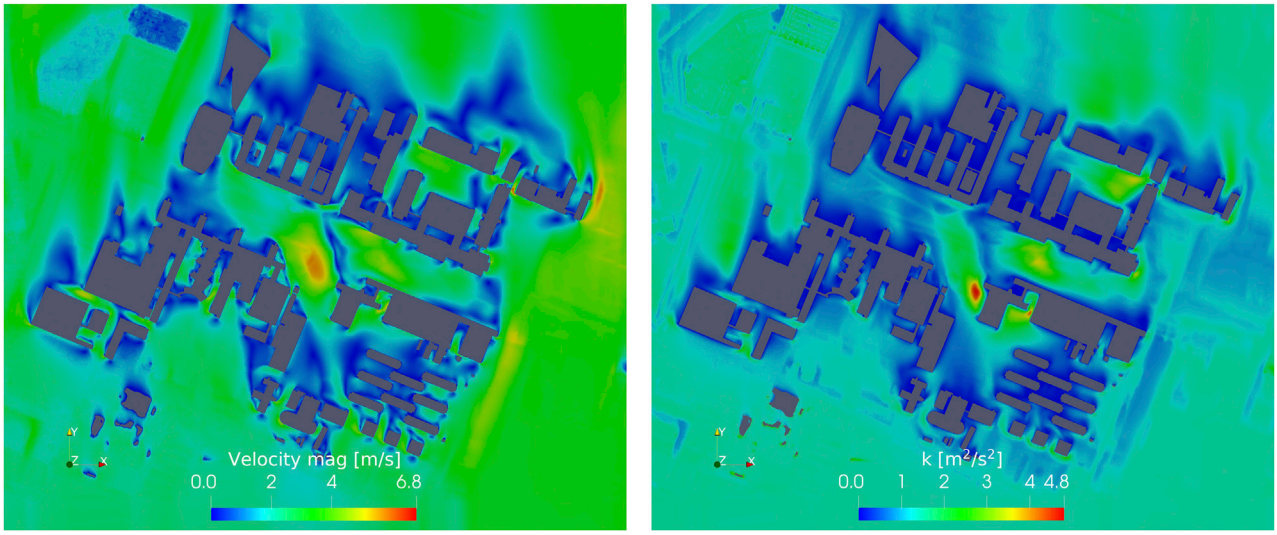

FIGURE 15
Velocity magnitude [m/s] and TKE [m^2/s^2] fields, observed at 2 m height (medium mesh).

level and z_0 is the aerodynamic roughness length. We set the roughness length as a "very rough" area with scattered buildings according to Wieringa (1992) for the inlet condition and terrain patch. The roughness length for water and vegetation patches correspond to the values of 0.0002 and 0.03 m, respectively, and are used together with the terrain in the rough wall function based on z_0 by Parente et al. (2011). Building walls are modeled as smooth. We used the central differencing scheme for pressure discretization, the bounded linear-upwind scheme for velocity discretization, and the limited-linear scheme for turbulence quantities.

Figure 15 shows the results of the velocity magnitude and the turbulence kinetic energy (TKE) fields at 2 m height for the medium mesh. The wind flow direction is aligned with the y axis. We can observe the influence of rougness layers on the flow field near the ground, as we can see changes in velocity magnitude and TKE corresponding to surface layer silhouettes. Furthermore, we can see the characteristic flow acceleration in the middle of the campus, next to the tallest building. Those effects were previously investigated in Kenjereš and Ter Kuile (2013) and García-Sánchez et al. (2021). Figures A1 and A2 in Appendix show the results of velocity magnitude and TKE fields for all three meshes.

Figure 16 presents the convergence of residuals for three investigated meshes; we can see that after 3,000 iterations velocity and turbulence residuals fall below $1e^{-4}$, whereas pressure residual converges to values below $1e^{-3}$, for all three cases.

We further investigated the convergence of pressure in Figure 17. Here, we calculated the integral value of pressure over the surface of the largest building on campus. We can see that, in all cases, the calculated value converges to a stable solution after a couple of hundred iterations.

We verified the solution in space using the grid convergence procedure summarized in Celik et al. (2008). To conduct the study, we sampled flow characteristics (velocity, pressure, and turbulence quantities) at 32 different locations in domain. We calculated the mesh sensitivity for all quantities at every probe location and extracted the median value. The results are shown in Table 2. e_a^{21} is the approximate relative error between the fine and the medium mesh, e_{ext}^{21} is the extrapolated relative error, and $\text{GCI}_{\text{ext}}^{21}$ denotes the fine grid convergence index.

The results exhibit a monotonous convergence towards asymptotic values, and relatively low values of the grid convergence index (GCI) indicate that meshes used in simulations were adequate.

5 Discussion and perspectives

We have presented a new workflow for the automatic reconstruction of simulation-ready 3D city models and we have shown it is applicable to urban flow simulations without additional modifications to the geometry. The workflow addresses arguably the most time-consuming steps of geometry preparation for urban flow simulations. This includes the reconstruction of buildings and terrain and their seamless merger, the imprinting of different surface layers into terrain without overlapping, and the automatic definition of domain size according to best practice guidelines. It does all that while being efficient and robust, resulting in geometries that are adequate for computational grid generators: watertight,

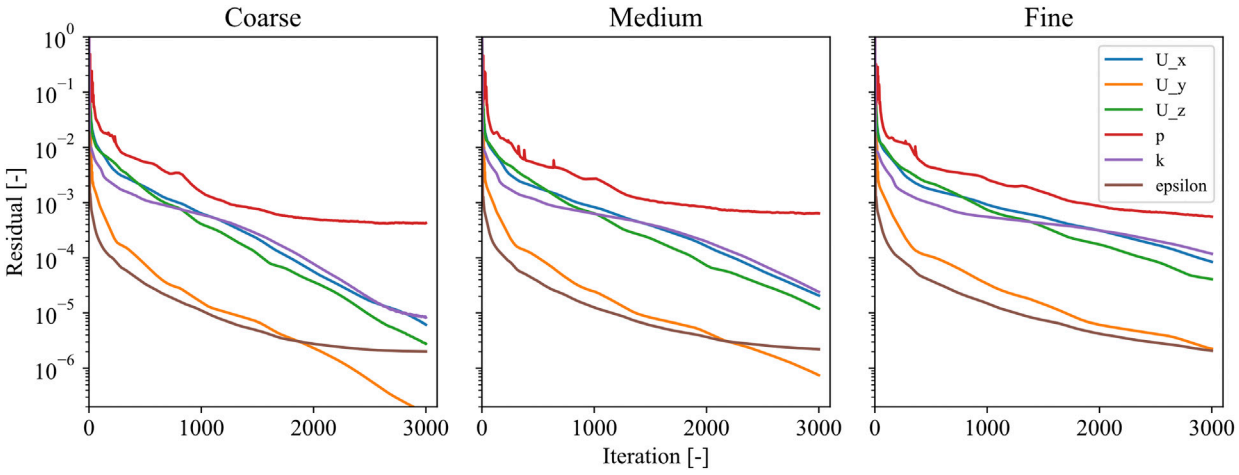

FIGURE 16
Convergence of residuals for investigated cases.

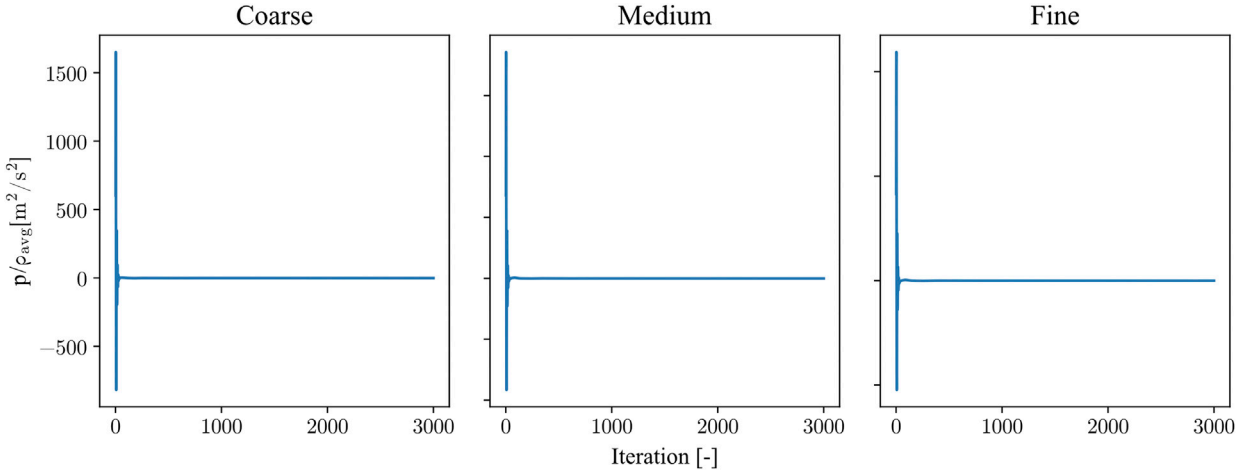

FIGURE 17
Convergence of the average pressure over one building.

TABLE 2 Grid convergence study, median values from 32 probes.

Variable	e_a^{21}	e_{ext}^{21}	GCI_{ext}^{21}
U_x	1.1%	0.4%	0.5%
U_y	1.5%	0.6%	0.2%
U_z	5.1%	4.1%	4.9%
p	1.6%	0.7%	0.1%
k	1.8%	1.5%	1.9%
ϵ	1.3%	1.5%	1.9%

without self-intersections and non-manifold edges. The resulting computational meshes showed satisfactory mesh quality indicators (non-orthogonality and skewness) and the subsequent CFD simulations displayed good convergence behavior.

We saw in the test case that the reconstruction time on a desktop workstation was around 1 minute. The total geometry preparation time is longer as it includes obtaining the input data and configuring the reconstruction parameters; by our estimate, the entire process took about two hours. We can compare this to the entirely manual process used in García-Sánchez et al. (2021) to prepare the same case, which

took a few days. The actual time savings are even greater, since in the manual process we approximated the terrain as a flat surface. Reconstructing the terrain and imprinting surface layers into it would further extend the preparation time. Given that most geometry preparation is still handled manually at present, we believe our tool can be beneficial to practitioners for modeling a wide range of wind engineering applications.

The workflow was implemented into an open-source framework using modern and state-of-the-art libraries. As such, it is intended to be used for further developments in geometry preparation for urban flow simulations. Our future work includes:

- Implementation of LoD1.3 and LoD2.2 reconstruction algorithm by Peters et al. (2021). Currently, to our knowledge, no purpose-built automatic reconstruction frameworks for urban flows exist for building LoDs greater than LoD1.2.
- Introducing generalization algorithms into the framework. While buildings created by our workflow are valid, we acknowledge that it is still possible for some computational grid generation packages to have issues creating the mesh due to sharp angles, short edges, and small distances between building walls. Geometry generalization algorithms such as Piepereit et al. (2018); Park et al. (2020) address those issues and also decrease the total size of the computational grid.
- Automatic reconstruction of trees from a point cloud. Large vegetation such as trees and hedges play an important role in urban climate studies (Hefny Salim et al., 2015; Buccolieri et al., 2018). We plan to implement algorithms that automatically reconstruct trees from a point cloud at different levels of detail, such as the work by de Groot (2020). Those reconstructed trees can be used to designate porous zones typically used in CFD simulations, as demonstrated by Gromke and Blocken (2015); Santiago et al. (2019); Kang et al. (2020) to name a few.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found below: https://github.com/tudelft3d/city4cfd.

Author contributions

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

Acknowledgments

CG-S would like to acknowledge the support provided by the Delft Technology Fellowship at Delft University of Technology.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Agugiaro, G. (2016). Energy planning tools and CityGML-based 3D virtual city models: Experiences from trento (Italy). *Appl. Geomat.* 8, 41–56. doi:10.1007/s12518-015-0163-2

An, K., Wong, S. M., Fung, J. C. H., and Ng, E. (2020). Revisit of prevailing practice guidelines and investigation of topographical treatment techniques in CFD-Based air ventilation assessments. *Build. Environ.* 169, 106580. doi:10.1016/j.buildenv.2019.106580

Arroyo Ohori, K., Ledoux, H., and Meijers, M. (2012). Validation and automatic repair of planar partitions using a constrained triangulation. *pfg.* 5, 613–630. doi:10.1127/1432-8364/2012/0143

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

Biljecki, F., Ledoux, H., and Stoter, J. (2016b). An improved LOD specification for 3D building models. *Comput. Environ. Urban Syst.* 59, 25–37. doi:10.1016/j.compenvurbsys.2016.04.005

Biljecki, F., Ledoux, H., and Stoter, J. (2017). Generating 3D city models without elevation data. *Comput. Environ. Urban Syst.* 64, 1–18. doi:10.1016/j.compenvurbsys.2017.01.001

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., and Çöltekin, A. (2015). Applications of 3D city models: State of the art review. *ISPRS Int. J. Geoinf.* 4, 2842–2889. doi:10.3390/ijgi4042842

Blocken, B. (2015). *Computational Fluid Dynamics for urban physics: Importance, scales, possibilities, limitations and ten tips and tricks towards accurate and reliable simulations. Building and Environment.* doi:10.1016/j.buildenv.2015.02.015

Blocken, B. (2021). "Introduction to the simulation of atmospheric flows," in CFD for atmospheric flows and wind engineering *(von Karman Institute Lecture Series)*.

Blocken, B., Janssen, W. D., and van Hooff, T. (2012). 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, 15–34. doi:10.1016/j.envsoft.2011.11.009

Blocken, B., and Persoon, J. (2009). Pedestrian wind comfort around a large football stadium in an urban environment: CFD simulation, validation and application of the new Dutch wind nuisance standard. *J. Wind Eng. Industrial Aerodynamics* 97, 255–270. doi:10.1016/j.jweia.2009.06.007

Blocken, B., Stathopoulos, T., and Carmeliet, J. (2007). CFD simulation of the atmospheric boundary layer: Wall function problems. *Atmos. Environ.* 41, 238–252. doi:10.1016/j.atmosenv.2006.08.019

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

Buccolieri, R., Santiago, J. L., Rivas, E., and Sanchez, B. (2018). Review on urban tree modelling in CFD simulations: Aerodynamic, deposition and thermal effects. *Urban For. Urban Green.* 31, 212–220. doi:10.1016/j.ufug.2018.03.003

Camelli, F., Lien, J. M., Shen, D., Wong, D. W., Rice, M., Löhner, R., et al. (2012). Generating seamless surfaces for transport and dispersion modeling in GIS. *GeoInformatica* 16, 307–327. doi:10.1007/s10707-011-0138-3

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

de Groot, R. (2020). Automatic construction of 3D tree models in multiple levels of detail from airborne LiDAR data. Master's thesis. Delft, Netherlands: Delft University of Technology.

Deininger, M. E., von der Grün, M., Piepereit, R., Schneider, S., Santhanavanich, T., Coors, V., et al. (2020). A continuous, semi-automated workflow: From 3D city models with geometric optimization and CFD simulations to visualization of wind in an urban environment. *ISPRS Int. J. Geoinf.* 9, 657. doi:10.3390/ijgi9110657

Dhunny, A. Z., Samkhaniani, N., Lollchund, M. R., and Rughooputh, S. D. (2018). Investigation of multi-level wind flow characteristics and pedestrian comfort in a tropical city. *Urban Clim.* 24, 185–204. doi:10.1016/j.uclim.2018.03.002

Dukai, B., Ledoux, H., and Stoter, J. E. (2019). A multi-height LOD1 model of all buildings in The Netherlands. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* 4, 51–57. doi:10.5194/isprs-annals-IV-4-W8-51-2019

Dukai, B., van Liempt, J., Peters, R., Stoter, J., Vitalis, S., and Wu, T. (2021). 3D BAG. Availableat: https://3dbag.nl/ (Accessed February, 2022).

Fan, H., Zipf, A., Fu, Q., and Neis, P. (2014). Quality assessment for building footprints data on OpenStreetMap. *Int. J. Geogr. Inf. Sci.* 28, 700–719. doi:10.1080/13658816.2013.867495

Franke, J., Hellsten, A., Schlünzen, H., and Carissimo, B. (2007). *Best practice guideline for the CFD simulation of flows in the urban environment*, 44.

Franke, J., Hirsch, C., Jensen, A. G., Krus, H. W., Schatzmann, P. S., Miles, S. D., et al. (2004). "Recommendations on the use of CFD in wind engineering," in *COST action C14: Impact of wind and storm on city life and urban environment*.

Gao, Z., Bresson, R., Qu, Y., Milliez, M., de Munck, C., Carissimo, B., et al. (2018). High resolution unsteady RANS simulation of wind, thermal effects and pollution dispersion for studying urban renewal scenarios in a neighborhood of Toulouse. *Urban Clim.* 23, 114–130. doi:10.1016/j.uclim.2016.11.002

García-Sánchez, C., Philips, D. A., and Gorlé, C. (2014). Quantifying inflow uncertainties for CFD simulations of the flow in downtown Oklahoma City. *Build. Environ.* 78, 118–129. doi:10.1016/j.buildenv.2014.04.013

García-Sánchez, C., Van Tendeloo, G., and Gorlé, C. (2017). Quantifying inflow uncertainties in RANS simulations of urban pollutant dispersion. *Atmos. Environ.* 161, 263–273. doi:10.1016/j.atmosenv.2017.04.019

García-Sánchez, C., Vitalis, S., Paden, I., and Stoter, J. (2021). *International Archives of the Photogrammetry, Remote Sensing and Spatial Information*, 46. doi:10.5194/isprs-archives-XLVI-4-W4-2021-67-2021The impac of level of detail in 3D city models for CFD-based wind flow simuations*Sci. - ISPRS Arch.*

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

Gold, C. M. (1988). 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), 133–147.

Google (2022). Google Earth. Availableat: https://earth.google.com/web/ (Accessed February, 2022).

Gorlé, C., and Iaccarino, G. (2013). A framework for epistemic uncertainty quantification of turbulent scalar flux models for Reynolds-averaged Navier-Stokes simulations. *Phys. Fluids* 25, 055105. doi:10.1063/1.4807067

Gromke, C., and Blocken, B. (2015). Influence of avenue-trees on air quality at the urban neighborhood scale. Part I: Quality assurance studies and turbulent Schmidt number analysis for RANS CFD simulations. *Environ. Pollut.* 196, 214–223. doi:10.1016/j.envpol.2014.10.016

Guney, C., Akdag Girginkaya, S., Cagdas, G., and Yavuz, S. (2012). Tailoring a geomodel for analyzing an urban skyline. *Landsc. Urban Plan.* 105, 160–173. doi:10.1016/j.landurbplan.2011.12.016

Guo, W., Liu, X., and Yuan, X. (2015). A case study on optimization of building design based on CFD simulation Technology of wind environment. *Procedia Eng.* 121, 225–231. doi:10.1016/j.proeng.2015.08.1060

Hågbo, T. O., Giljarhus, K. E. T., and Hjertager, B. H. (2021). Influence of geometry acquisition method on pedestrian wind simulations. *J. Wind Eng. Industrial Aerodynamics* 215, 104665. doi:10.1016/j.jweia.2021.104665

Hecht, R., Meinel, G., and Buchroithner, M. (2015). Automatic identification of building types based on topographic databases–a comparison of different data sources. *Int. J. Cartogr.* 1, 18–31. doi:10.1080/23729333.2015.1055644

Hefny Salim, M., Heinke Schlünzen, K., and Grawe, D. (2015). Including trees in the numerical simulations of the wind flow in urban areas: Should we care? *J. Wind Eng. Industrial Aerodynamics* 144, 84–95. doi:10.1016/j.jweia.2015.05.004

ISO (2003). *ISO 19107:2003 (geographic information: Spatial schema)*. International Organization for Standardization.

Kang, G., Kim, J. J., and Choi, W. (2020). Computational fluid dynamics simulation of tree effects on pedestrian wind comfort in an urban area. *Sustain. Cities Soc.* 56, 102086. doi:10.1016/j.scs.2020.102086

Kawaguchi, T., and Oguni, K. (2016). Automatic conversion of visually consistent digital maps to conforming geometry for computational fluid dynamics. *J. Comput. Civ. Eng.* 30, 04015003. doi:10.1061/(asce)cp.1943-5487.0000473

Kenjereš, S., and Ter Kuile, B. (2013). Modelling and simulations of turbulent flows in urban areas with vegetation. *J. Wind Eng. Industrial Aerodynamics* 123, 43–55. doi:10.1016/j.jweia.2013.09.007

Ledoux, H., Arroyo Ohori, K., Kumar, K., Dukai, B., Labetski, A., Vitalis, S., et al. (2019). CityJSON: A compact and easy-to-use encoding of the CityGML data model. *Open geospatial data Softw. stand.* 4. doi:10.1186/s40965-019-0064-0

Ledoux, H., Biljecki, F., Dukai, B., Kumar, K., Peters, R., Stoter, J., et al. (2021). 3dfier: Automatic reconstruction of 3D city models. *J. Open Source Softw.* 6, 2866. doi:10.21105/joss.02866

Ledoux, H., Commandeur, T., Nagel, C., Trometer, S., and Vager, S. (2017). Automatic reconstruction of simulation-ready 3D city models. Tech. rep., Delft University of Technology, CADFEM and virtualcitySYSTEM.

Ledoux, H. (2018). val3dity: validation of 3D GIS primitives according to the international standards. *Open geospatial data Softw. stand.* 3, 1. doi:10.1186/s40965-018-0043-x

Liu, S., Pan, W., Zhang, H., Cheng, X., Long, Z., Chen, Q., et al. (2017). CFD simulations of wind distribution in an urban community with a full-scale geometrical model. *Build. Environ.* 117, 11–23. doi:10.1016/j.buildenv.2017.02.021

Liu, S., Pan, W., Zhao, X., Zhang, H., Cheng, X., Long, Z., et al. (2018). Influence of surrounding buildings on wind flow around a building predicted by CFD simulations. *Build. Environ.* 140, 1–10. doi:10.1016/j.buildenv.2018.05.011

Lu, Y., Behar, E., Donnelly, S., Lien, J. M., Camelli, F., Wong, D., et al. (2011). Fast and robust generation of city-scale seamless 3D urban models. *Computer-Aided Des.* 43, 1380–1390. doi:10.1016/j.cad.2011.08.029

Macdonald, R. W., Griffiths, R. F., and Hall, D. J. (1998). An improved method for the estimation of surface roughness of obstacle arrays. *Atmos. Environ.* 32, 1857–1864. doi:10.1016/S1352-2310(97)00403-2

Mallet, C., and Bretar, F. (2009). Full-waveform topographic lidar: State-of-the-art. *ISPRS J. Photogrammetry Remote Sens.* 64, 1–16. doi:10.1016/j.isprsjprs.2008.09.007

Mirzaei, P. A. (2021). CFD modeling of micro and urban climates: Problems to be solved in the new decade. *Sustain. Cities Soc.* 69, 102839. doi:10.1016/j.scs.2021.102839

Nys, G. A., Poux, F., and Billen, R. (2020). City json building generation from airborne LiDAR 3D point clouds. *ISPRS Int. J. Geoinf.* 9, 521. doi:10.3390/ijgi9090521

Parente, A., Gorlé, C., van Beeck, J., and Benocci, C. (2011). *Boundary layer meteorology.* A comprehensive modelling approach for the neutral atmospheric boundary layer: Consistent inflow conditions, wall function and turbulence model *140*.

Park, G., Kim, C., Lee, M., and Choi, C. (2020). Building geometry simplification for improving mesh quality of numerical analysis model. *Appl. Sci. Switz.* 10, 5425. doi:10.3390/APP10165425

PDOK (2022). *Pdok*. Availableat: https://www.pdok.nl/datasets/(Accessed February, 2022).

Peters, R., Dukai, B., Vitalis, S., van Liempt, J., and Stoter, J. (2021). *Automated 3D reconstruction of LoD2 and LoD1 models for all 10 million buildings of The Netherlands manuscript submitted for publication to journal of photogrammetric engineering & remote sensing*.

Piepereit, R., Deininger, M., Kada, M., Pries, M., and Voß, U. (2018). "A sweep-plane algorithm for the simplification of 3D building models in the application scenario of wind simulations," in *In int. Arch. Photogramm. Remote sens. Spatial inf. Sci*, XLII-4/W10. doi:10.5194/isprs-archives-XLII-4-W10-151-2018

Piepereit, R., Schilling, A., Alam, N., Wewetzer, M., Pries, M., Coors, V., et al. (2016). Towards automatic processing of virtual city models for simulations. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* 4, 39–45. doi:10.5194/isprs-annals-IV-2-W1-39-2016

Piroozmand, P., Mussetti, G., Allegrini, J., Mohammadi, M. H., Akrami, E., Carmeliet, J., et al. (2020). Coupled CFD framework with mesoscale urban climate model: Application to microscale urban flows with weak synoptic forcing. *J. Wind Eng. Industrial Aerodynamics* 197, 104059. doi:10.1016/j.jweia.2019.104059

Ricci, A., and Blocken, B. (2020). On the reliability of the 3D steady RANS approach in predicting microscale wind conditions in seaport areas: The case of the IJmuiden sea lock. *J. Wind Eng. Industrial Aerodynamics* 207, 104437. doi:10.1016/j.jweia.2020.104437

Ricci, A., Kalkman, I., Blocken, B., Burlando, M., Freda, A., Repetto, M. P., et al. (2017). Local-scale forcing effects on wind flows in an urban environment: Impact of geometrical simplifications. *J. Wind Eng. Industrial Aerodynamics* 170, 238–255. doi:10.1016/j.jweia.2017.08.001

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

Saeedrashed, Y. S., and Benim, A. C. (2019). Validation methods of geometric 3D-CityGML data for urban wind simulations. *E3S Web Conf.* 128, 10006. doi:10.1051/e3sconf/201912810006

Santiago, J. L., Buccolieri, R., Rivas, E., Calvete-Sogo, H., Sanchez, B., Martilli, A., et al. (2019). CFD modelling of vegetation barrier effects on the reduction of traffic-related pollutant concentration in an avenue of Pamplona, Spain. *Sustain. Cities Soc.* 48, 101559. doi:10.1016/j.scs.2019.101559

Shewchuk, J. R. (1997). *Delaunay refinement mesh generation*. Pittsburg, USA: Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Sibson, R. (1981). "A brief description of natural neighbour interpolation," in *Interpreting multivariate data*. Editor V. Barnett (New York, USA: Wiley), 21–36.

Simões, T., and Estanqueiro, A. (2016). A new methodology for urban wind resource assessment. *Renew. Energy* 89, 598–605. doi:10.1016/j.renene.2015.12.008

Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., et al. (2014). *CFD vision 2030 study: A Path to revolutionary computational aerosciences*. Tech. Rep. NASA/CR-2014-218178, NASA.

Thordal, M. S., Bennetsen, J. C., Capra, S., Kragh, A. K., and Koss, H. H. H. (2020). Towards a standard CFD setup for wind load assessment of high-rise buildings: Part 2 – blind test of chamfered and rounded corner high-rise buildings. *J. Wind Eng. Industrial Aerodynamics* 205, 104282. doi:10.1016/j.jweia.2020.104282

Toja-Silva, F., Chen, J., Hachinger, S., and Hase, F. (2017). CFD simulation of CO2 dispersion from urban thermal power plant: Analysis of turbulent Schmidt number and comparison with Gaussian plume model and measurements. *J. Wind Eng. Industrial Aerodynamics* 169, 177–193. doi:10.1016/j.jweia.2017.07.015

Toja-Silva, F., Lopez-Garcia, O., Peralta, C., Navarro, J., and Cruz, I. (2016). An empirical-heuristic optimization of the building-roof geometry for urban wind energy exploitation on high-rise buildings. *Appl. Energy* 164, 769–794. doi:10.1016/j.apenergy.2015.11.095

Toja-Silva, F., Pregel-Hoderlein, C., and Chen, J. (2018). On the urban geometry generalization for CFD simulation of gas dispersion from chimneys: Comparison with Gaussian plume model. *J. Wind Eng. Industrial Aerodynamics* 177, 1–18. doi:10.1016/j.jweia.2018.04.003

Tominaga, Y., Mochida, A., Yoshie, R., Kataoka, H., Nozu, T., Yoshikawa, M., et al. (2008). AIJ guidelines for practical applications of CFD to pedestrian wind environment around buildings. *J. Wind Eng. Industrial Aerodynamics* 96, 1749–1761. doi:10.1016/j.jweia.2008.02.058

Tong, Z., Chen, Y., and Malkawi, A. (2016). Defining the Influence Region in neighborhood-scale CFD simulations for natural ventilation design. *Appl. Energy* 182, 625–633. doi:10.1016/j.apenergy.2016.08.098

Toparlar, Y., Blocken, B., Maiheu, B., and van Heijst, G. J. (2018). The effect of an urban park on the microclimate in its vicinity: A case study for antwerp, Belgium. *Int. J. Climatol.* 38, e303–e322. doi:10.1002/joc.5371

Toparlar, Y., Blocken, B., Vos, P., Van Heijst, G. J., Janssen, W. D., van Hooff, T., et al. (2015). CFD simulation and validation of urban microclimate: A case study for bergpolder zuid, RotterdamBuilding and environment, *Rotterdam*. 83, 79–90. doi:10.1016/j.buildenv.2014.08.004

Weber, P., and Haklay, M. (2008). OpenStreetMap: User-generated street maps. *IEEE Pervasive Comput.* 7, 12–18. doi:10.1109/mprv.2008.80

Weller, H. G., Tabor, G., Jasak, H., and Fureby, C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Comput. Phys.* 12, 620. doi:10.1063/1.168744

Wieringa, J. (1992). Updating the Davenport roughness classification. *J. Wind Eng. Industrial Aerodynamics* 41, 357–368. doi:10.1016/0167-6105(92)90434-C

Zhang, S., Kwok, K. C., Liu, H., Jiang, Y., Dong, K., Wang, B., et al. (2021). A CFD study of wind assessment in urban topology with complex wind flow. *Sustain. Cities Soc.* 71, 103006. doi:10.1016/j.scs.2021.103006

10.3389/fbuil.2022.899332

Appendix

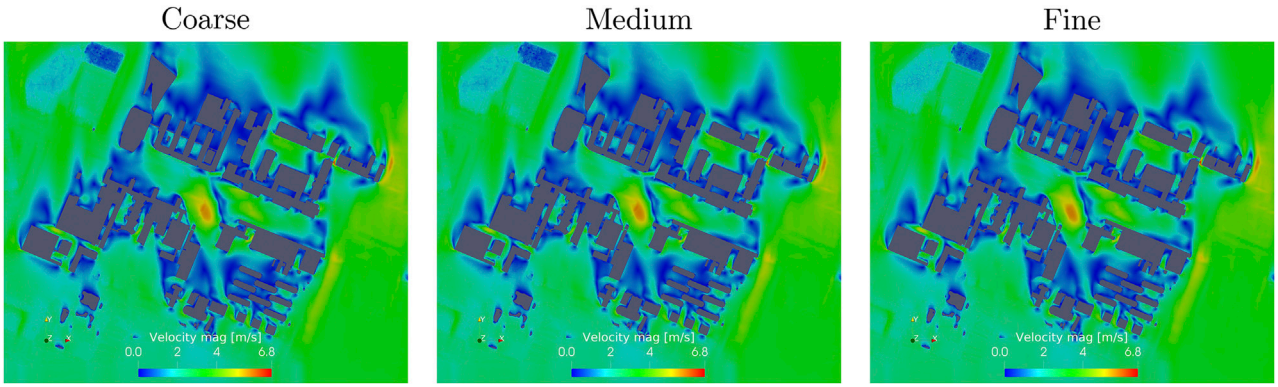

FIGURE A1
Velocity magnitude fields for all three investigated meshes.

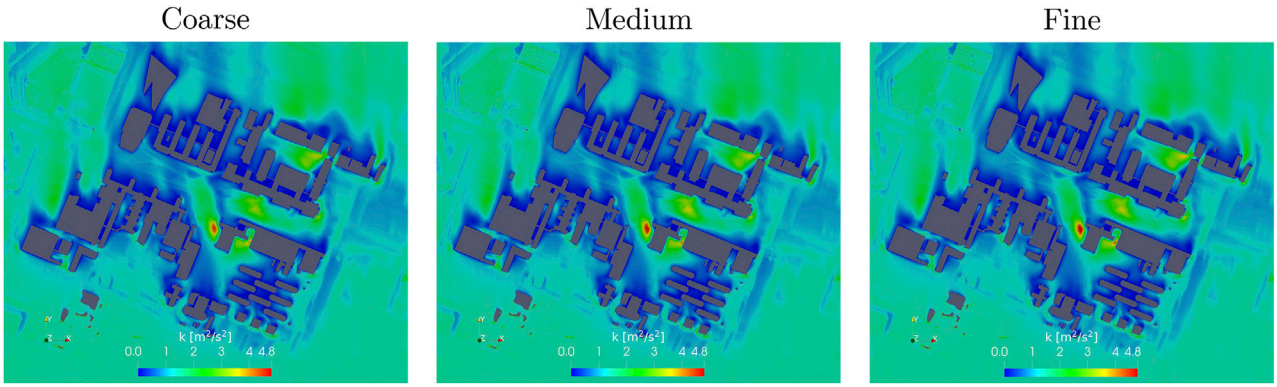

FIGURE A2
TKE fields for all three investigated meshes.