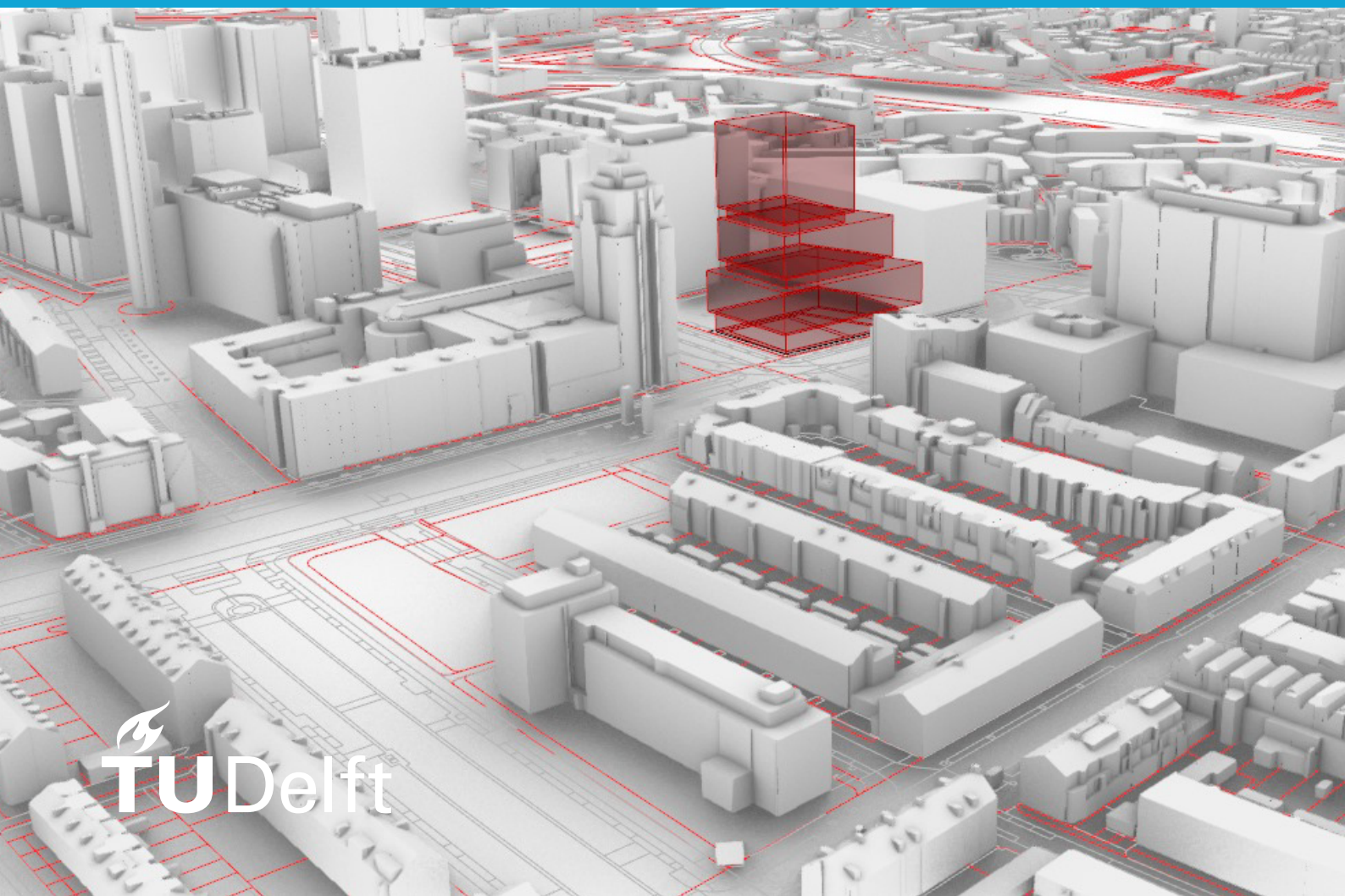
Synthesis Project Report for MCs Geomatics

(Co-)development of an open-data-based tool to perform preliminary environmental analyses at district scale in different European countries

Bing-Shiuan Tsai
Lars Huizer
Michele Giampaolo
Sérénic Monté
Sicong Gong

November 2023



Synthesis Project Report for MSc Geomatics

(Co-)development of an open-data-based tool to perform preliminary environmental analyses at district scale in different (European) countries

Bing-Shiuan Tsai
Lars Huizer
Michele Giampaolo
Sérénic Monté
Sicong Gong

November 2023

Bing-Shiuan Tsai, Lars Huizer, Michele Giampaolo, Sérénic Monté, Sicong Gong:
(Co-)development of an open-data-based tool to perform preliminary environmental analyses at district scale in different (European) countries (2023)

The work is supported by:

Royal HaskoningDHV
Enhancing Society Together

Supervisors of this project

TU Delft:	Giorgio Agugiaro
Royal HaskoningDHV:	Gabriel Garcia

Contents

List of Figures

List of Tables

Acronyms

1 Introduction

1.1 Background and Motivation

With the high speed and scale of urbanisation, built environment practitioners are facing challenges such as meeting accelerated demand for affordable housing, viable infrastructure including transport systems, and many more. An efficient and accurate spatial analysis is of vital importance to sustain this rapidly growing construction, not only by providing comprehensive insights into the site for planning but also by mitigating the impact of the construction on its surroundings.

Recently, the advance of spatial data acquisition and processing technologies has brought a considerable yield of 3D data of the built environment. The primitive data collected in Point Cloud or Mesh formats from Lidar or Photogrammetry techniques can be transformed into specialised data formats such as CityGML (OGC [2023]) or CityJSON (Ledoux et al. [2019]) and further processed by Geographical Information System (GIS) software. These abundant spatial datasets are made available on the Internet as open data, which shows great potential to support spatial analysis.

In the realm of Geomatics, the main focus is on the acquisition, modelling, and management of spatial data from the built environment. On the other hand, the Architecture, Environment and Construction (AEC) practitioners may pay more attention to the tools that allow direct operation and process of 3D data. The existing powerful tools such as Quantum Geographic Information System (QGIS), Auto Computer Aided Design (AutoCAD), Rhinoceros and its related extensions all allow users to perform spatial data analysis, visualisation and simulation of the environmental contexts, facilitating the decision-making while planning and designing.

Despite the prevailing adoption of 3D data applications across various analytical tools, the process of acquiring the available and processable input data remains a challenging hurdle in the field of AEC. These difficulties often pose significant obstacles to the seamless progress of AEC projects, impacting their overall efficiency and effectiveness.

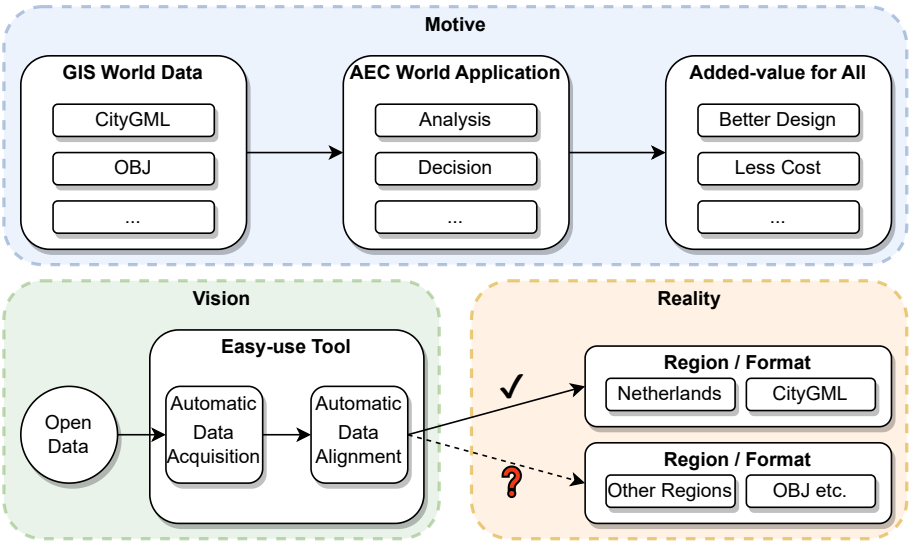


Figure 1.1: Motivation, Vision and Reality

Therefore, establishing an efficient interface that supports data collection and process to generate a readable format for AEC applications has become the ongoing goal of the GIS industry, satisfying the increasing need for data from the AEC industry. The value of spatial data will only be maximised with high accessibility and processability for further analysis and application. This added value of spatial data will facilitate the AEC industry in rapidly adapting to the ever-changing environment, enabling them to provide more reasonable and research-based solutions.

Royal HaskoningDHV is an independent consultancy firm which integrates 140 years of engineering expertise with digital technologies and software solutions. It has developed a prototype such as an

interface extension tool on Rhino/Grasshopper to facilitate their architects and engineers to retrieve the target datasets necessary for environmental analysis in the built environment. The tool allows users to directly import the most up-to-date geodata in Rhino, bypassing its traditional cumbersome acquisition through different portals and Application Programming Interfaces (APIs).

However, this tool is currently limited to the Netherlands where the required spatial datasets are highly accessible and of high quality, and it is still unsupported in other countries/regions in which the company operates due to the heterogeneous data formats and their inaccessibility. To enhance the applicability of the tool, this research takes on the role of establishing a generic framework to enrich the existing interface while using the data portals of different countries/regions.

The research project will first focus on seeking the corresponding local datasets that are necessary to offer the current function. Secondly, it will delve into the process of adapting the collected data from the local APIs to the extension tool, which finally enables the tool to support global projects. Thus comes the project title: **(Co-)development of an open-data-based tool to perform preliminary environmental analyses at district scale in different European countries**.

1.2 Client Requirements and Use Cases

The current Rhino-based toolkit developed by Royal HaskoningDHV[1] offers an interface for the architects to perform preliminary analyses on a 3D model. This model is composed of a Digital Terrain Model (DTM), building geometries with a certain Level of Detail (LoD), and other additional city element geometries obtained via open spatial data portals. The main concept is to allow the user to seamlessly establish a 3D simulation environment of their project area from open data.

To achieve the target analyses in terms of visibility, energy and wind, the core data of the project must include 3D building and terrain modelling, which provide the volumetric information to perform energy consumption analysis of buildings, basic shadowing impact assessment and the view simulation of new construction. For more supplementary details, it is suggested to have land cover and usage data, which provides specific surface types of information for facilitating simulation like wind dispersion and heat analysis, resulting in a more accurate outcome. Additional optional datasets could include the orthophoto, road, vegetation and cadastre data. This additional information could enable users to perform an inspection on a deeper level, offering a more in-depth context analysis related to the project.

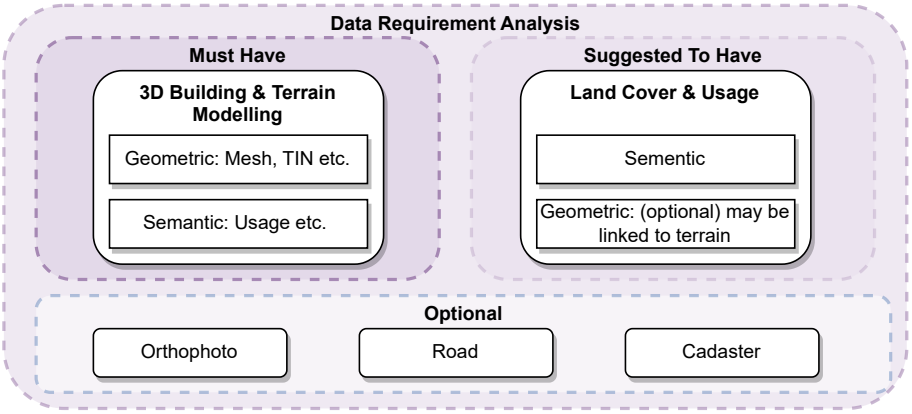


Figure 1.2: Data Requirement Analysis

As part of the requirements from the client, the main workflow needs to be carried out in Rhino, more specifically with the use of Grasshopper, for the geometry generations and with the Ladybug suite for the simulations. Another requirement was to limit the use of other plugins and external processes, preferably to none. This was requested by the client to increase the ease of script sharing as well as to reduce the complexity in the hands of users who might not be experienced in geodata processing. A brief introduction of the tool and its related extensions and functions for data processing and analysis is shown as follows.

[1] https://www.royalhaskoningdhv.com/en/about-us

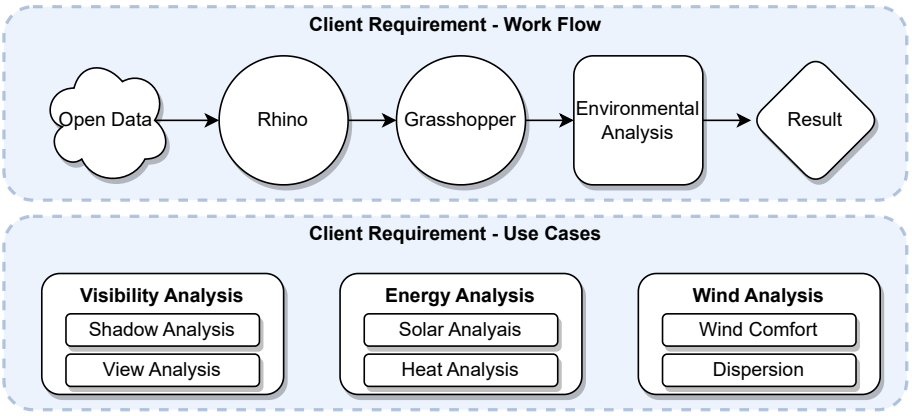


Figure 1.3: Client Requirement Analysis

Rhino[2] is a CAD package that can be used to create 3D geometries. Objects can be modified and transformed in various ways, allowing for various applications, among which detailed architectural and city models. In this project, Rhino is used as a viewer so that the collected data and the generated geometries can be analysed and visualised.

Grasshopper[3] is a visual programming editor that allows to create and alter parametrically geometries in Rhino. It offers various inbuilt components, among which ones which allow scripts to be run in other programming languages, such as Python and C#. Grasshopper is used in this project to load and transform the collected data to be visualised Rhino.

Ladybug[4] is a tool used to add environmental analysis functionalities to Grasshopper. In this project, Ladybug is used to perform the environmental experiments in the second phase as a validation for the first phase. It can be integrated with additional components that amplify its application, such as in the case of computational fluid dynamics and solar analysis.

Table 1.1: The Components of Ladybug

Ladybug	Honeybee	Butterfly	Dragonfly	Spider
Climate Visualisation + Analysis	Building Energy, Daylight + Comfort Modelling	Airflow Modelling (CFD)	Urban Modelling (urban energy, heat island, custom epw)	Web Visualisation (sun path, shadows, gbxml viewing/editing)

The required data will be imported in Rhino via the aid of Grasshopper, and then the elaboration of data alignment on two method types will be discussed, which will focus on the standard post-process to build up the database to support tools such as Ladybug, Honeybee and Butterfly etc in Rhino. Finally, an experiment of the target analyses in terms of visibility, energy and wind will be conducted by these Rhino tools, and a summary of the result will be provided to suggest further development.

[2] https://www.rhino3d.com/features/
[3] https://en.wikipedia.org/wiki/Grasshopper_3D
[4] https://www.ladybug.tools/

2 Data Evaluation

As indicated in the introduction, this project mainly focuses on preliminary analyses which are made available by using the built-in tools in the Rhino and Grasshopper software. Therefore, the datasets used for establishing 3D environments should be general and easily accessible to support core analysing functionalities. Since the tool is tailor-made for architectural analysis, the scale of the datasets is set to the district level, which is suitable for the assessment of new construction on the urban scale.

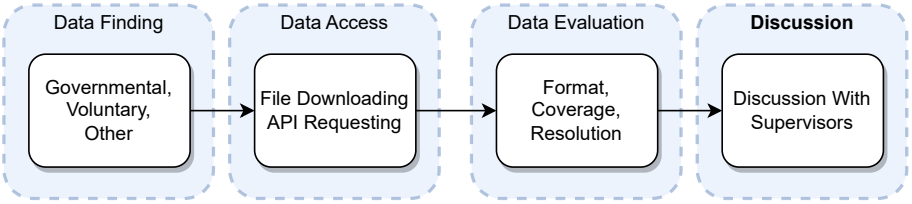


Figure 2.1: Methodology for Data Finding and Evaluation

In the initial research phase, the target datasets were searched in ten countries/regions in the world. After the exploration of different data portals, it was decided along with the client to specifically delve into the countries/regions of Germany, Hong Kong, Italy, Spain and the United Kingdom (UK). The following tables show the results of the target data accessibility among all the preferred countries/regions.

Among the five cases, Hong Kong has the highest accessibility of open data, all the target data is available through Application Programming Interface (API), which offers dynamic query to obtain the necessary data near the interest point while Italy and the UK are more conservative on open data. It is also important to note that Hong Kong is the only case that provides 3D LoD building data in OBJ format. As for the download-only data, it will be processed with the File-based method since each dataset will be saved separately before getting imported into Rhino. Both the API and File-based methods will be further explored in section Section 3.1.

2.1 Data Finding in Germany

The geodata available in Germany are generally split up according to the different states. This means that any end user looking to model cities in different states may end up needing to find individual data for separate locations. In addition, the method by which the data is delivered may vary per administrative region as well. For example, the LoD2 files for the region of Hamburg had to be downloaded in its entirety instead of per single tile, whereas the state of Bavaria requires the user to individually download tiles that divide the dataset. In addition, this data, although aggregated on geoportal.de, may sometimes require to be downloaded from the local Spatial Data Infrastructure (SDI) instead.

Table 2.1: Data finding in Germany

Region	Type	Name	Format	API	Detail	Coverage	Other
Germany	**Building**	3D-Gebäudemodell LoD2-DE Hamburg	CityGML	Download	LoD2	Fully (Hamburg)	Many separate files (entire region, tiled up)
	Terrain	Digitales Höhenmodell Hamburg	xyz	WMS / Download	Grid size of 1 metre	Fully (Hamburg)	Many separate files (entire region, tiled up)
	Land Use	Bodennutzung	image	WFS / Download		Fully (Hamburg)	
	Road	Straßen- und Wegenetz Hamburg	Various (e.g. CSV, GeoJSON)	WFS / Download	Line features	Only main roads Hamburg	
	Cadastral	Liegenschaftskataster	?	?	-	Fully (Hamburg)	Paid dataset
	Link	https://geoportal.de/					

For this exercise in data availability, the decision was made to analyse the city of Hamburg and its associated administrative state. For architectural contextual analysis, the necessary data in the form of buildings in LoD2 as well as digital terrain models were available. These were both delivered in a format consisting of many different files representing the entire administrative region of Hamburg, divided into different tiles. The finding of Germany is shown in table 2.1.

2.2 Data Finding in Hong Kong

Hong Kong's data is overall complete, valid, well-organised and mostly obtainable through API. These advantages may only come from the small area of Hong Kong, being only 1104 km² in area. However, its various types of data, especially three-dimensional OBJ data based on photogrammetry, have universal utilisation value and access channels, such as Google Earth. In this project, Hong Kong's data serves as a window to observe the situation outside Europe due to the market expansion needs of the clients. At the same time, it can compare the differences between the non-semantic model mainly based on OBJ and the semantic model mainly based on CityGML. The finding of Hong Kong is shown in table 2.2.

Table 2.2: Data finding in Hong Kong

Region	Type	Name	Format	API	Detail	Coverage	Other
Hong Kong	**Building**	3D Visualisation Map (2017)	OBJ	WFS/WMS	3D LoD2 Model	Fully	Tiling Indexes of the shapes; Files are separate for one tile
	Land-use	Digital Topographic Map iB1000	GML etc.	WFS/WMS	2D Vector	Partially	Tiling Indexes of the files
	Terrain	Digital Terrain Model (DTM)	GEOTIFF	WMS	2D Raster	Fully	All the data at once
	Orthophoto	Digital Orthophoto DOP5000	GEOTIFF	WFS/WMS	2D Raster	Fully	Tiling Indexes of the files
	Road	Road Network	GML etc.	WFS/WMS	2D Vector	Fully	Tiling Indexes of the files
	Cadastral	-	-	-	-	-	No data
	Link	CSDI Portal - Common Spatial Data Infrastructure (CSDI)					

2.3 Data Finding in Italy

The open data landscape in Italy exhibits significant heterogeneity across the country. While certain national datasets, such as the DTM, are available, most geospatial data is managed at a regional level. Each region operates its own Geoportal in which its catalogue of data can be found. However, a lack of standardisation leads to substantial variation between these platforms. Despite the presence of a national Geoportal where data from all regions is compiled, it still retains the fragmented nature of the initial datasets.

Table 2.3: Data finding in Italy

Region	Type	Name	Format	API	Detail	Coverage	Other
Italy	Building	Building footprints (Piedmont)	SHP	Download	2D polygon	Fully	Has field for building height
	Land Use	Land use map (Piedmont)	SHP	Download	2D polygon	Fully	
	Transport	Roads (Piedmont)	SHP	Download	2D line	Fully	
		Train lines (Piedmont)	SHP	Download	2D line	Fully	
		Bike paths (Piedmont)	SHP	Download	2D line	Fully	
	Terrain	DTM 5m	GeoTIFF	Download	2.5D raster	Fully	Download per tile
		DSM 5m	GeoTIFF	Download	2.5D raster	Fully	Only available via email request
		Home - Geoportale Piemonte					

Accessing a single thematic dataset for the entire country can prove to be challenging, as the metadata details, field names, accuracy, coverage, scale, format and availability could all present differences based on the region of origin. Furthermore, the overall availability of WFS and WMS APIs is limited,

at times exclusively offered for certain datasets and often not available at all. For these reasons, and given the time restrictions and scope of the project, it was chosen to limit the research in Italy to one region. Piedmont was chosen to act as this case study.
Italy was chosen as one of the countries/regions for the project due to its contrast with the Netherlands in terms of geodata availability. The choice was further supported by the client company's existing operations in the country. The finding of Italy is shown in table 2.3.

2.4 Data Finding in Spain

Spain utilises a national Geoportal to collect its spatial open data datasets. The ones required for this project are available on a national level, and as such the required data is available and homogenised for the entire country. However, the available data of this national Geoportal is limited, especially in the case of thematic data. For instance, the transport network data sets only consist of an edge graph and no information on the size and type of road.

Table 2.4: Data finding in Spain

Region	Type	Name	Format	API	Detail	Coverage	Other
Spain	Building	Building outline from cadastre	GML	WFS	2D	Fully	
	Cadastre	Parcels from cadastre	GML	WFS	2D	Fully	
	Land Use	Land Cover Map 2018	GDB or Geopackage	Download	2D	Fully	Complete country
	Road	Transport networks	SHP	Download	2D	Fully	Per region
	Terrain	DSM of buildings and vegetation	ASCII (.asc) ESRI arrayfile	Download	2.5D raster	Fully	Download per tile
		DSM of buildings	ASCII (.asc) ESRI arrayfile	Download	2.5D raster	Fully	Download per tile
		http://centrodedescargas.cnig.es/CentroDescargas/index.jsp					

There are also regional Geoportals which vary greatly in terms of quality. Some provide higher quality thematic data, but the ones that don't often have worse quality than the national Geoportal. The available data may be incomplete or limited to an extremely small subregion. Since the national Geoportal provides the required data at a quality that meets the requirements for this project, datasets from regional Geoportals will not be used.

Another available source is the Spanish Cadastre, which provides the parcels and building outlines through a WFS for the entire country. The finding of Spain is shown in table 2.4.

2.5 Data Finding in the UK

The UK's open data sources are mainly provided by the Ordnance Survey, which is a free official portal offering nationwide primitive data for download only, such as DTM tiles in TIFF format, shapefiles of building footprints, etc. Several privately-owned data portals offer more advanced and integrated data such as 3D LoD3 building models but they are all offered at a fee based on the areal size to download selected by the user. Since the focus of this project is on the application of open data, the primitive data from the Ordnance Survey is used as the main input for the model reconstruction and further analyses. The finding of the UK is shown in table 2.5.

Table 2.5: Data finding in the UK

Region	Type	Name	Format	API	Detail	Coverage	Other
United Kingdom	**Building**	3D Model	3DS/DWG/ 4CS/SKP	Download	3D LoD1-3 Model	Fully	Tiling Indexes of the shapes; Various licensed data providers and all data is chargeable
	Land-use	osfeatures:Zoomstack_(District/Local)Buildings	SHP(exportable to other formats)	Download	2D Vector	Fully	All the data at once; With shape area attribute but no height ; Too large to process
	Terrain	Land Cover Map 2021	PNG	Download	2D Raster	Fully	All the data at once;
	Orthophoto	LIDAR DTM-2022	TIF/laz	Download	2D Raster/Point cloud	Partially	Tiling Indexes of the shapes;
	Road	OS Terrain® 50	SHP/GML/ Geopackage/ Vector tile(MBtile)	Download	2D Vector	Fully	All the data at once; Contour line only
	Cadastral	Vertical Aerial Photography Tiles RGB	ECW	Download	2D Raster	Partially	Tiling Indexes of the shapes; quite old
	Link	https://environment.data.gov.uk/DefraDataDownload/?Mode=survey					

2.6 Data Finding in Other Regions

2.6.1 Data Finding in Finland

The geodata found in Finland are generally similar to other European countries that actively participate in the Infrastructure for Spatial Information in Europe (INSPIRE) project, and as such the integrity and availability of data can be guaranteed. In particular, it offers a three-dimensional urban semantic model that is mainly based on CityGML. One of the main shortcomings is that Finland's data cannot be obtained directly through the API. Instead, users need to apply via email on the web page and then download the data from the response. On the one hand, the inability to obtain it through API does not meet the customer's needs for real-time and automation. Due to this inaccessibility of the data and similarity to other countries/regions chosen, Finland was not chosen as one of the countries/regions analysed in this project.

2.6.2 Data Finding in Ireland

The Irish organisation Tailte Éireann (formerly known as the Ordnance Survey Ireland, or OSI) are responsible for national mapping and surveying infrastructure within the country. They offer paid data, which is unfit for this study. Certain datasets are made accessible as open data, however, very little relevant data is available here. In some cases more local datasets are available, but these are often of low quality or insufficient for the project. For these reasons, Ireland was deemed inappropriate for this project.

2.6.3 Data Finding in Taiwan

The open datasets like the building footprint and DTM are available on the "National Land Surveying and Mapping Centre" official data portal. Recently, the portal began to support 3D building information surveying, however, the map service is limited to inspecting only. Since the backend data are provided by the local governments across the nation, the processable data is only available via individual applications, which limits the data accessibility and does not meet our requirement for open data application. Thus, the result of data finding in Taiwan can only be seen as a reference.

2.6.4 Data Finding in France

The French national Geoportal contains the spatial open data sets required by the project. The quality of the data is also satisfactory and similar to some of the countries/regions chosen, such as Spain. Although the data quality was found to be sufficient for the research, France was in the end not chosen as one of the countries/regions to analyse, due to its similarity with other EU countries as well as due to the client expressing more interest in the other chosen countries/regions.

2.6.5 Data Finding in Australia

Australia's open data access is divided based on its states. Each offers its data on a separate geoportal which allows the viewing and in certain cases direct access to the datasets. 3D building data is available in certain states with varying coverage and quality. The state of Victoria currently offers the most complete open 3D building dataset. It covers the entire state and offers LoD2 buildings within Melbourne and LoD1 outside the capital. It is divided based on municipalities and each dataset is available via email request. The state of Queensland offers mostly LoD1 building data for certain sections of Brisbane, but not for the entire city. The state of New South Wales offers models of small areas created as tests for a future more widespread implementation.

Overall, it appears that the 3D data in Australia is currently in a phase of growth and might see significant improvement in the next few years. While the data was available and in certain parts of the country of high quality, it was ultimately excluded for the rest of the study to maintain the main focus on the requested analysis of European countries. However, in the future, it shows potential as a possible case with which to continue this study, both for the availability of the required datasets as well as for the current presence of the client company's operations.

2.7 Conclusion

Overall, every location that has been the subject of the study has had differing methods in which the relevant geographic data is delivered, ranging from something that would be very easy to implement into a tool (e.g. Hong Kong where all of its data is available per API) to ones that require more manual user interaction to make the data workable (Italy, where data often needs to be manually downloaded).

The heterogeneity in the provision of all these different data sources indicates that it is unlikely that a single solution will be a fit for all of the countries/regions, and as such different countries/regions will require separate efforts if they were to be integrated into a tool. The same consideration applies to the different formats in which the data is delivered, although more overlap is present which should provide some more united implementations.

3 Methodology

In this chapter, the methodologies of this project will be introduced. Firstly, the overview of the technique workflow will be described. Then, in Sections 1, 2 and 3, the data acquisition, data alignment and simulation experiment will be introduced. Finally, a conclusion of this chapter will be given as the "prelude" to the implementation part.

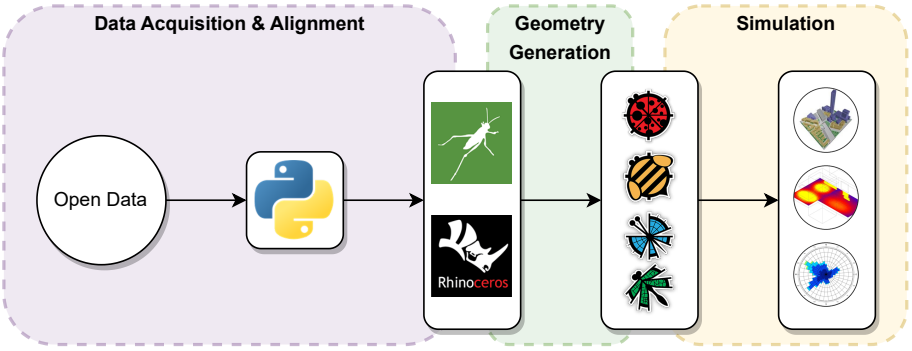


Figure 3.1: Methodology Overview

As mentioned in the Introduction chapter, the main goal of this project is to develop a tool that could help the AEC practitioners to do environmental analysis, which could be further divided into the following three sub-goals.

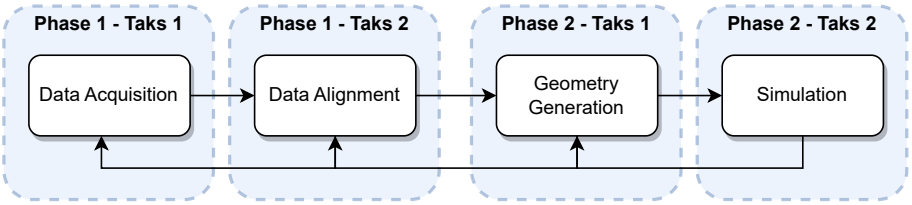


Figure 3.2: Iterative Process

1. Data Acquisition and Alignment: Automatically acquire the built-environment open data from the Internet.
2. Geometry Generation: Automatically align and separate the geometry of buildings and context.
3. Simulation: Automatically prepare the models and parameters for the wind simulation and experiment with the effectiveness.

3.1 Data Acquisition and Alignment

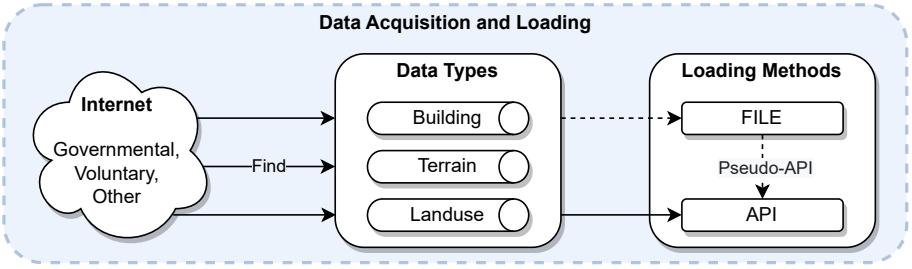


Figure 3.3: Data Acquisition

There are several methods for users to acquire built-environment data such as obtaining data from the mobile physical storage device (hard drive or USB flash drive) or receiving data by email (usually

after registering and applying from the platforms). However, those methods are heavy, non-timely, inconvenient, and, most importantly, they require manual participation and cannot be automated.

Compared to the "traditional" methods mentioned above, the Internet could provide users with alternatives. For example, they could directly download the data file from the platforms possibly with a licence. A more convenient way is through the API: the users only need to enter the data name or type without having to consider the details of the data acquisition process. With these considerations in mind, the API-based data acquisition method is the optimal method of data acquisition for this research and will be used when present.

3.1.1 File-based Method

The typical way for spatial data modelling is to import the data layer by layer, the input files can be retrieved through an online data portal. For instance, the Department for Environment Food & Rural Affairs in the UK has provided a platform for the users to download datasets such as LIDAR composite DTM, Digital Surface Model (DSM), point cloud, etc. For the retrieval of building models, if there is no available official resource, the user can still use open source like OpenStreetMap to download the building data in their area of interest.

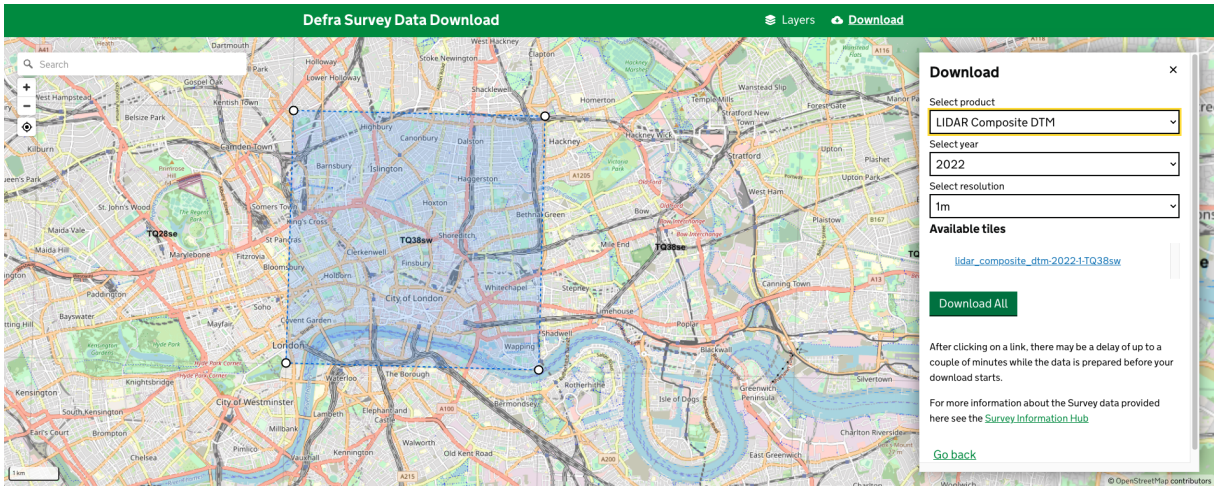


Figure 3.4: Tile data retrieved from the UK portal (https://environment.data.gov.uk/survey)

The file-based method may allow the users to either upload or draw an extent as the boundary for data retrieval from the portal's remote database, the downloaded files could be DTM in TIFF format or vectorised feature data like building footprints in shapefiles. These file formats often require further processing to be used in Rhino/Grasshopper.

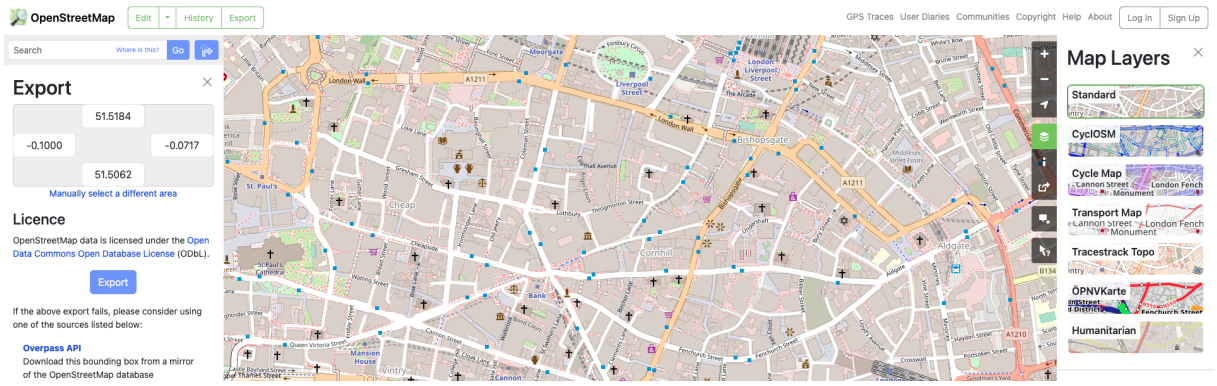


Figure 3.5: Tile data retrieved from OpenStreetMap portal (https://www.openstreetmap.org)

3.1.2 API-based Method

With the aid of the Internet, the API is usually implemented by the W3C protocols, specifically, the GET, POST requests etc. They are HTTP methods used to interact with web servers, where GET retrieves data from a server, and POST could also submit data to a server for processing (Purewal [2014]). In the GIS world, more specifically APIs are designed by the Open Geospatial Consortium (OGC).

The most frequently used **ocg!** APIs are Web Feature Service (WFS) and Web Map Service (WMS) APIs. A WFS request is used for retrieving vectorised feature data (In GML, JSON etc. formats) from the data source. The WMS request is used to return map images (In PNG, TIFF etc. formats) instead of features (Davis [2007]). Many of the formats could not be directly used in Rhino/Grasshopper, several plugins and custom parsing scripts are needed to do so.

3.2 Geometry Generation

The quality of the data from the Internet cannot always be guaranteed (especially for open-source data). There may always be issues considering its completeness, consistency and accuracy. Some of them could be fatal, such as missing data that cannot be repaired. Others can be repairable such as the overlapping, disjointedness or holes in the geometry.

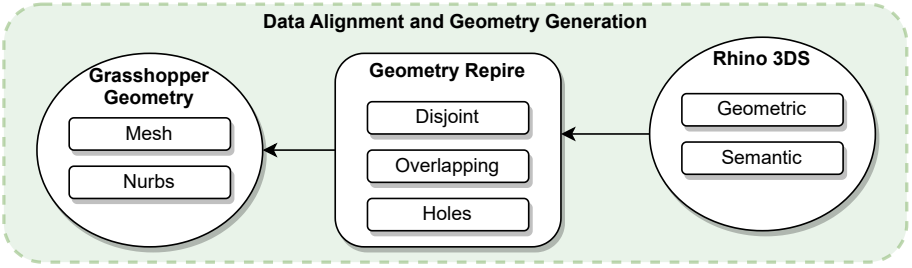


Figure 3.6: Data Alignment & Geometry Generation

Based on the data input, there may be different methods to check and repair the geometry for the environmental simulation analysis (Pađen et al. [2022]; Donkers et al. [2016], Ledoux et al. [2022]; Arroyo Ohori et al. [2022]). Among them, the TIN-based and Voxel-based methods may be the most frequently used. The TIN-based method may have higher accuracy but is more complex. The voxel-based method, although has a higher computational cost and lower accuracy, is more stable and simple to implement.

3.2.1 TIN-based Method

The Triangular Irregular Network (TIN)-based method is based on representing the geometry as a network of interconnected triangles. The basic idea is to use neighbouring features to check and repair non-consistent vertices (isolated vertices etc.), edges (dangling edge etc.) or faces (intersected faces etc.) to form a valid triangulated mesh (usually a 2-manifold) that closely approximates the original geometry. The most commonly used TIN is the Delaunay Triangulation (DT), which is a fundamental data structure for terrains, both for their representation and for their processing (Ledoux et al. [2022]).

3.2.2 Voxel-based Method

The Voxel-based method, on the other hand, involves dividing the 3D space into small, equally sized cubic cells called voxels. Each voxel is analysed and used to represent the geometry within that region. The exterior boundary (envelope) of the volumetric representation would also be a valid geometry that approximates the original one.

3.3 Simulation Experiment

The general steps of an environmental simulation can be summarised as follows:

1. Configuration: Begin by configuring the simulation engine and inputting the relevant geometry.

2. Parameter Specification: Specify the necessary environmental parameters, which may include variables like temperature and wind speed, as well as user-defined parameters like resolution and iteration times.
3. Simulation Execution: Run the simulation using the provided input and parameters.
4. Post-Processing: After the simulation is complete, perform post-processing tasks, including visualisation and analysis

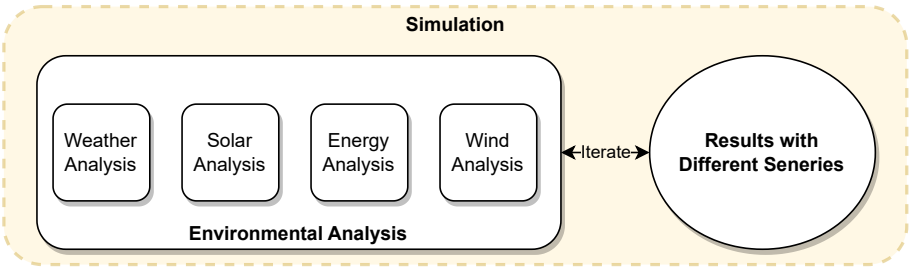


Figure 3.7: Simulation Experiment

In the context of the AEC industry, the primary focus of practitioners often revolves around the shape and transformations of the geometry. AEC professionals typically prioritise the structural and design aspects of their projects. As a result, it is crucial to encapsulate other complex environmental and simulation-related issues in as simplified a manner as possible. Some environmental simulation engines exist to achieve this goal.

3.3.1 Solar Analysis

Radiance[1] is an environmental simulation primarily focused on the accurate simulation of light, daylight, and energy flows within built spaces. Radiance is known for its precision in predicting the distribution of light and visualising the impact of architectural designs on illumination and thermal comfort. It is embedded in the Ladybug tool in Rhino/Grasshopper.

3.3.2 Wind Analysis

OpenFOAM[2] is an open-source Computational Fluid Dynamics (CFD) engine which plays a significant role in simulating airflow, thermal comfort, and pollutant dispersion within architectural and urban spaces. OpenFOAM offers flexibility and customization, allowing AEC professionals to address complex airflow and environmental issues in building design and urban planning. It is embedded in the Butterfly tool in Rhino/Grasshopper.

In the simulation sections that will follow (4.3, 5.3), an Minimal Reproducible Example (MRE) is proposed using mostly the default or automatically generated parameters for experimental purposes. It is important to emphasise that this MRE is not meant as an accurate representation of real-world scenarios due to the complexity of the involved parameters. It is solely intended to test the efficacy of the obtained models.

Among the numerous considerations to be made are the turbulence model, solving algorithms and solver. These are research topics in specific areas, and thus they are out of the scope of this project which is development-based. Even when applying a simple test case, using the laminar turbulence model, the steady incompressible recipe and the default solver for "simplefoam", there are still parameters to directly specify. Among them are the wind tunnel, boundary conditions, background mesh and snappyHexMesh (Blocken [2015], García-Sánchez et al. [2021]).

The testing of these parameters is out of the scope of this project, and as such set values were used. It is important to note that these parameters would not only influence the efficiency and accuracy of the simulation, but they may also influence the convergence of the simulations. In cases where convergence does not occur, obtaining accurate results becomes impossible. In this instance, the simulation successfully converged, demonstrating the effectiveness of the generated model.

[1] https://www.radiance-online.org/
[2] https://www.openfoam.com/

Following the above explanation of the limitations and goals of the MRE, the steps for the overall process to perform wind simulation from the generated city geometry are listed below.

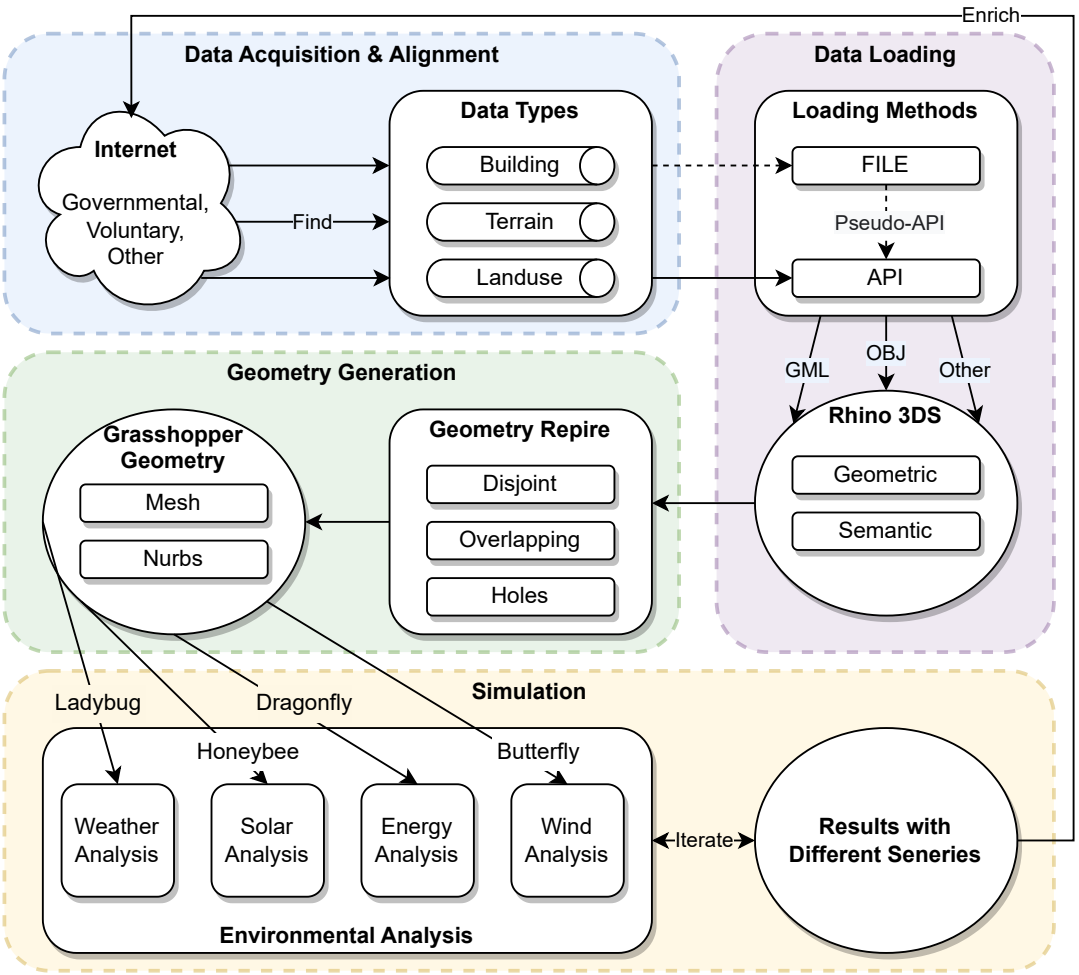


Figure 3.8: Overall Workflow

1. **Specifying needed parameters**: the necessary parameters are used as input. These are "case name" for project creation, "wind direction" for wind tunnel creation, "cell size of blockmesh" for background mesh creation, "refinement level of buildings" for refined mesh creation, and number "number of CPUs" for the parallel running.
2. **Simulation execution**: all the required steps are clustered in a transformer, after inputting the geometry and specifying the parameters, the simulation can be started.
3. **Visualisation**: due to the limited capability of Rhino and Grasshopper in handling the visualisation of complex models, the postprocessing is done in the ParaView application, which is packaged with OpenFOAM during the Butterfly installation.

4 Implementation - TIN Case

In this chapter, the TIN-based implementation is introduced. First, the overview of implementation is described. Then, in Section 4.1, Section 4.2 and Section 4.3, the data acquisition and alignment, geometry generation and simulation experiments are discussed in detail. Finally, a conclusion of this chapter is given as well as the limitations of the current implementation.

As mentioned in the Methodology chapter, there are three sub-goals of this project. The ideal input and output of the three parts are discussed below and the workflow diagram is shown in Figure 4.1.

1. The data acquisition and alignment are achieved via the file-based method, the primitive data is processed with the aid of QGIS, specific Python scripts executed in an Integrated Development Environment (IDE) such as PyCharm, and Python script components embedded in Grasshopper
2. The geometry generation is achieved through the TIN-based method, and the implementation is carried out with components and Python scripts embedded in Grasshopper.
3. The simulation experiments perform Solar and Wind analysis, and they were implemented with the use of Radiance and OpenFOAM engines, made available in Grasshopper through the Ladybug and Butterfly extensions. Paraview was used for the wind simulation visualisation.

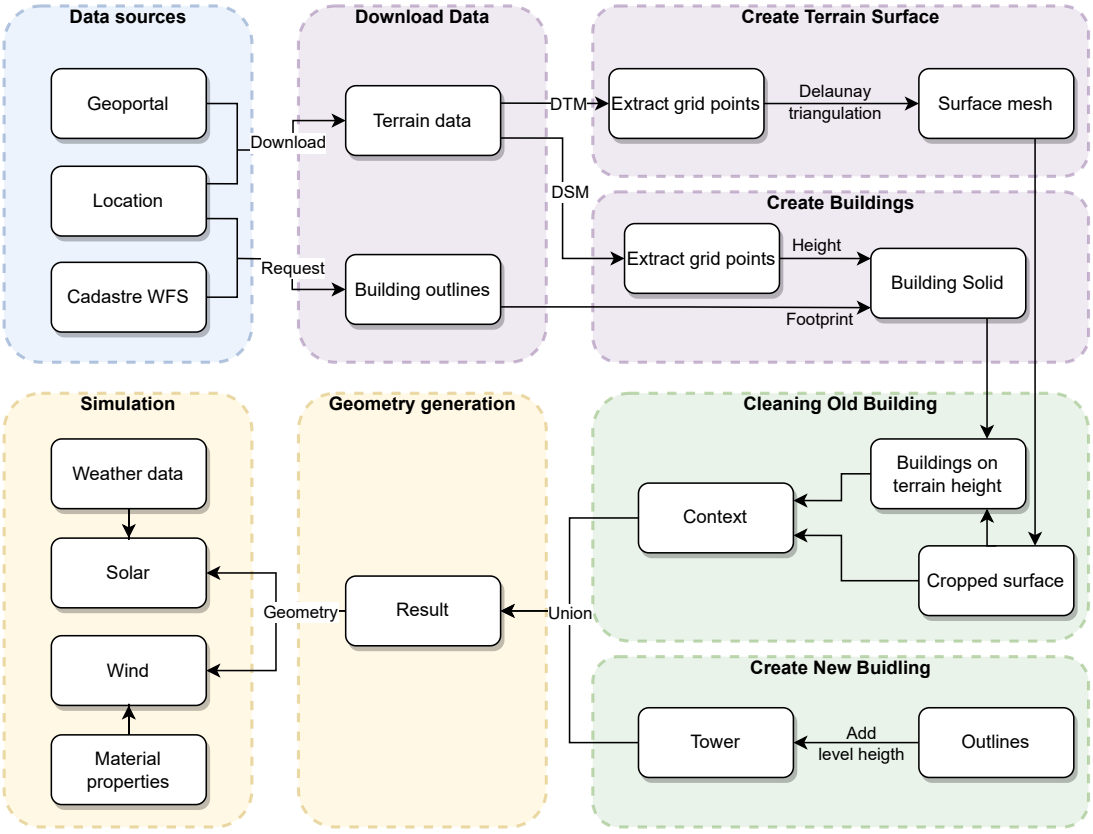


Figure 4.1: Implementation of TIN case

4.1 Data Acquisition and Alignment

Before the datasets can be processed in Grasshopper, it is necessary to correctly import them into the program. As different data formats and distribution methods are used by the different countries/regions, a phase of data acquisition and alignment needs to take place where this process is correctly carried out. Ideally, the data would be available through OGC APIs, such as WMS and WFS, and be in data formats natively readable in Grasshopper. This would reduce the number of actions required from the user, as no direct search and download of the data would be necessary, as well as guarantee the use of consistently up-to-date information.

However, in most of the experiences carried out this was not the case. Datasets were for the most part retrievable with the file-based approach (see Section 3.3.1) and some formats required additional plugins or external processing to be performed before they could be inserted into Grasshopper. In the following section, the processes needed to acquire and align the different dataset formats are described.

4.1.1 DTM

The terrain and surface height data is often available as a gridded raster, such as in a GeoTIFF format. Since this format cannot be directly read by Grasshopper, alternative methods need to be applied for its insertion.

The approach followed was that of transforming the raster file into a point cloud dataset by associating the x and y coordinates of each pixel's centre point with its corresponding z value. It was performed through the Python package named "rasterio" (Gillies [2019]) and stored as an XYZ file. This format can be parsed as a text file through the Grasshopper Python interpreter.

Another method was attempted but ultimately abandoned for its ineffectiveness. This involved installing the "Bison"[1] Grasshopper plugin, which with its "Import Mesh DEM" automatically takes a TIFF file as input and creates a Mesh in Rhino. However, it automatically translated the Mesh's centre to the Rhino origin, without explicitly giving the performed transformation. This caused aligning the mesh with the other geometries to be unsuccessful, as it was not possible to reproject it to a common coordinate system. Furthermore, it gave less control ability than the previously described method, as the individual insert points used to create the mesh could not be changed.

4.1.2 DSM

Building height data was in certain cases available from the DSM of the area of interest. To determine the building height corresponding to a building footprint, the height value above it is collected from the building DSM, and the 95 percentile is taken to determine the actual height. These heights and footprints are then used to make the LoD1 building solids

4.1.3 Building SHP

The building footprints as a SHP file was the most common way in which building geometry was obtained. This posed a problem as the built-in Grasshopper "Import SHP" has the limitation of not outputting the field values associated with each geometry, which in some applications were needed as the building height was stored inside them. For this reason, an external plugin was required to extract all the necessary data inside of Grasshopper. The one ultimately chosen was "Local Software"[2], which imports the geometries, attribute names and attribute values as three different Grasshopper trees.

The geometry entries and attribute value indices in each tree correctly correspond to each other, making them easily connected. Another plugin was considered, "Heron"[3], but it was ultimately abandoned as errors were encountered regarding incorrect geometry reprojection and geometry/attribute indices incongruence.

4.1.4 CityGML

City geometry in a CityGML format was available in certain cases, from which building geometry could be extracted. It was able to be retrieved directly through a WFS API, and so directly loaded into Grasshopper. By following the CityGML data model (OGC [2023]), the GML file was parsed with various Grasshopper Python components to extract the required geometries.

[1] https://www.bison.la/
[2] https://www.food4rhino.com/en/app/local-software
[3] https://www.food4rhino.com/en/app/heron

4.1.5 Boundary Selection

Often the acquired datasets span a large physical area, and as such can be computationally expensive to perform operations on. For this reason, a way to select a boundary to delimit the area to be loaded and considered for future operations is imperative to efficiently continue with the geometry reconstruction. Since the geometries are projected to a specific coordinate system, this is done by providing a link[4] to an online platform where the desired project coordinates can be chosen. Afterwards, a radius can be specified to create a boundary around it. Using this limit, the various geometries can be cropped, by either selecting the objects that are contained inside it in the case of buildings, or by cropping the mesh with that constraint in the case of the terrain.

4.1.6 Results

With the steps above followed the result will be the geometries being correctly imported into Grasshopper. Depending on the data available, this could mean having building footprints, full building models and terrain meshes.

After the main steps mentioned above, there will be two meshed models in each case. One is the DTM mesh generated from the point coordinates and the other is the buildings represented in Boundary Representation (Brep).

4.2 Geometry Generation

After the acquisition step, the 3D digital surface model(s) of the context could be obtained. The buildings are represented by Boundary Representations (Breps) in LoD1, and the terrain is represented in DT mesh. Since the building model is disjoint with the terrain model, it fails to reflect reality and may induce errors in further simulation. The following step after the acquisition is to correctly seal these two models as a watertight DSM. Our team has attempted to implement the Constrained Delaunay Triangulation (CDT) method proposed by Pađen et al. [2022].

However, the result reproduced by using the Rhino/Grasshopper turns out to be unusable due to the complexity of the resulting model and the limit of computation power. Therefore, a simplified process will be introduced as well to achieve the sealing goal. In this section, the process will focus on aligning the 3D geometry represented in TIN and Brep. In the meantime, one new building is inserted (operations developed for users) into the context to create different scenarios.

4.2.1 Terrain Generation

As introduced in the previous section, the terrain data was inserted in Grasshopper as a point cloud in XYZ format. A Grasshopper Python script allows for a mesh to be reconstructed. It also allows the user to input a thinning variable called Level of Simplification (LoS) filter a certain amount of points to simplify the mesh and optimise processing speeds. The code for the height extraction of TIFF files is shown in Appendix A.

4.2.2 Building Generation - CDT Method

According to the reconstruction process (Pađen et al. [2022]), the footprint points are first projected on the terrain mesh to perform height interpolation.

Then, the constraint is added at the overlapping part of two models, in which the edges are split at the intersection and new vertices are added. After adding the constraint, the original points and all the newly added constraint points are used as an input set to generate the CDT mesh.

Finally, each triangle will be addressed with semantic labels, and the height of all the CDT vertices in each footprint will be averaged to a certain level for the building's height extrusion. Our team adapts the workflow to fit Rhino/Grasshopper, the process is shown as the following:

1. For each building footprint, create a brep and convert it to a meshed surface.
2. Split the mesh surface with the terrain bounding box, which will give the intersecting points of cutting edges and surfaces.

[4]For example, in the case of the WGS 84 UTM zone 32N, this `link` was provided

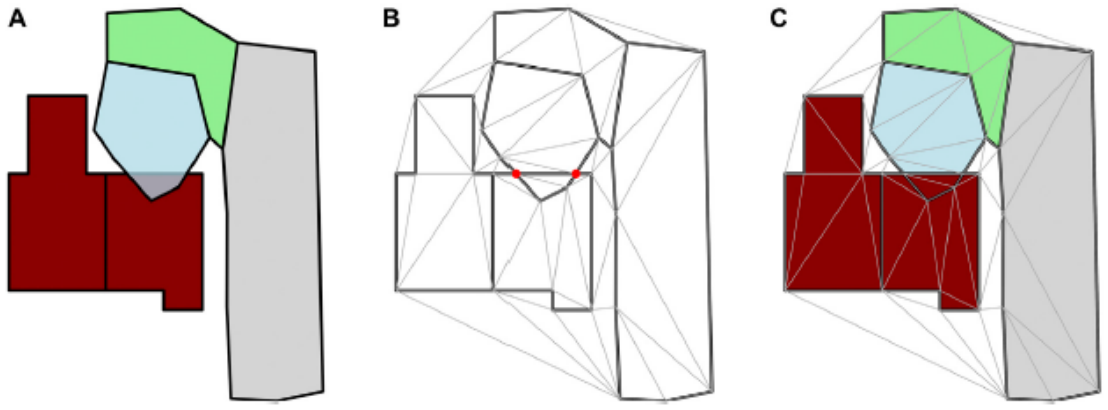


FIGURE 4
(A) 5 polygons: 2 buildings, 1 pond, 1 grass field, and 1 road. **(B)** Resulting CDT. Observe that 2 polygons are overlapping (pond and one building), their edges were split at the intersections and new vertices added (in red). **(C)** Triangulation with labels from the input polygons (with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

Figure 4.2: CDT method illustration (Pađen et al. [2022])

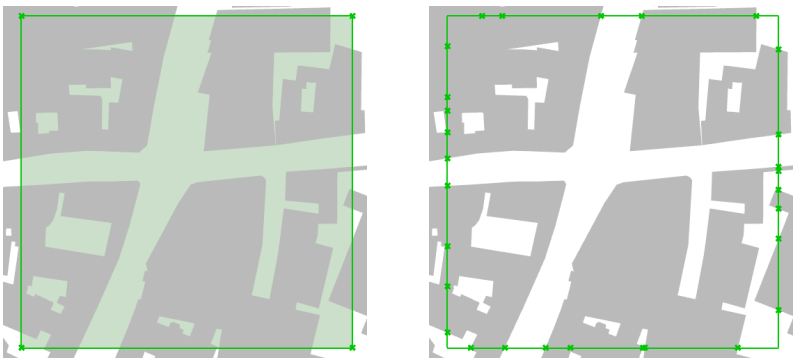


Figure 4.3: Terrain bounding box(left), Split edge points (right)

3. Deconstruct the mesh with a custom setting to control the maximum allowed edge length between meshed points. Note that in Grasshopper, the custom setting for max edge control will affect the density of meshed points lie inside the polygon, the shorter the edge, the denser the meshed points will be, which compromises the performance.

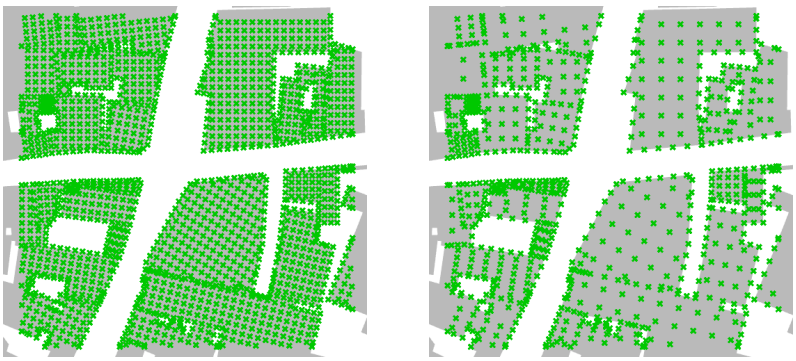


Figure 4.4: Meshed building points
max edge: left (2 units) - right(10 units)

4. Deconstruct the terrain bounding box to get its vertices and select all the meshed building points in the terrain bounding box.

5. Merge all the points obtained in step 2 to 4, project those points on the DT terrain mesh. In this step, the projected points will be automatically assigned with the linearly interpolated height values from the terrain mesh.
6. Average the height values and assign the averaged height to all the points from step 5 and the terrain points that fall in each building footprint.

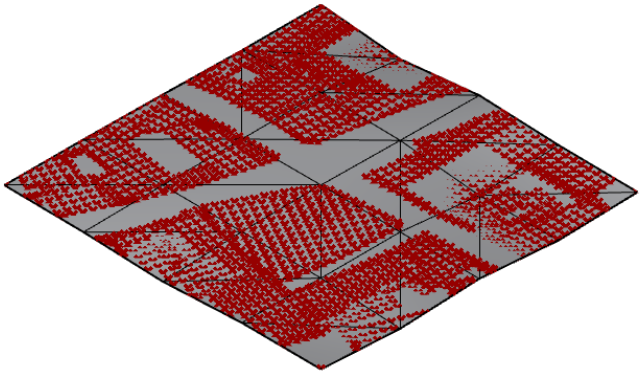


Figure 4.5: Merged points projection on the terrain mesh

6. Merge the projected points in step 6 with the rest of the terrain points, generate the CDT mesh from the merged point set.
7. Use the face normals to get the centres of each CDT, select the Constrained Delaunay Triangles (CDTs) that have its centre in the footprints, which helps to label the building CDTs.

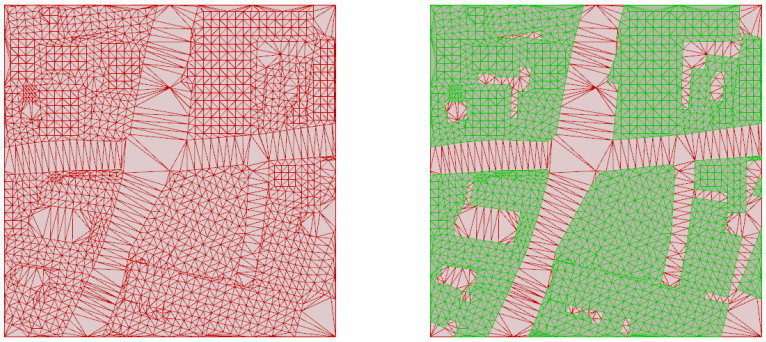


Figure 4.6: Resulting CDT mesh(left), Labelled building CDTs in green (right)

8. Find the lowest z-value of the CDTs in each footprint and project the CDTs on the plane at the lowest height, which gives a planar mesh for extrusion that covers all the footprint area and also guarantee the roof of the resulting LoD1 building model is flat.
9. Average the z-values of the CDTs in each footprint, compute the difference between the averaged and lowest z-values and add the result to the building's height that is derived from the shapefile, which gives the extrusion height.
10. Extrude the CDTs created in step 8 with the height in step 9.

The above process demonstrates the sealing of building and terrain models with the CDT method for generating a watertight model. Note that, since the buildings are extruded from the footprint at its lowest z-value, the extruded solid will intersect with the CDT terrain mesh, an extra step to remove the underground part is needed. However, the geometry generation time with the CDT method will last unexpectedly long and would reach the memory limitation of Rhino (2GB) when the bounding box size is set above 500 metres (see table 4.1) to merely integrate the original buildings and terrain. Furthermore, it may fail to allow users to perform further operations like insertion of a new building since each operation requires heavy recalculation, and this method is only tested on the UK case. All the

limitations compromise the flexibility and the usability of this method, making it an only experimental approach.

Table 4.1: Geometry Generation Time by CDT-method

Bounding Box size (m)	Time cost (s)
100	12.3
200	66
300	312
400	666
500	Out of memory

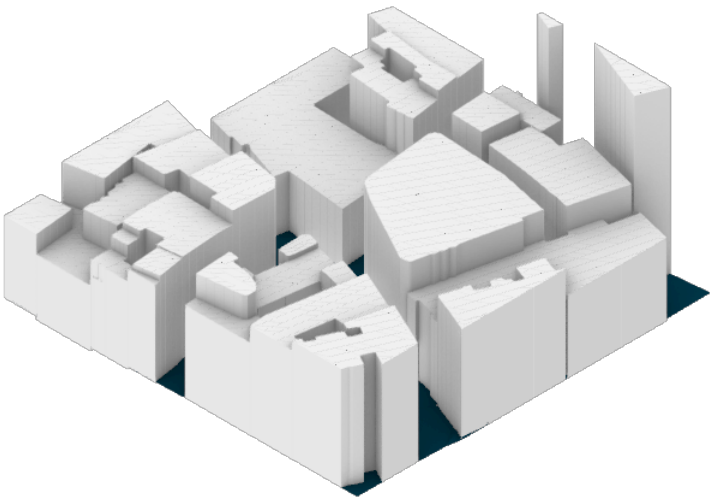


Figure 4.7: Resulting watertight model (the UK case)

4.2.3 Building Generation - Simplified Method

The simplified method for building generation treats the terrain and the buildings as separate geometries that intersect to have a watertight model with both geometries, without any buildings "floating" above the terrain. It assumes that following the data acquisition and alignment steps, the building footprints and terrain geometries are aligned on the x,y planes but have conflicting heights. The general steps performed to be able to achieve this are listed below.

1. Calculate the centre point of each building footprint.
2. Project the centre point to the terrain mesh model.
3. Translate the footprint along the z direction to the height of the projected centre point
4. Select a height for the building to "sink" under the ground, ideally, this should be a value that goes beyond the terrain's lowest point.
5. Extrude the footprint along the z direction by the sum of its corresponding height and the selected "sinking height"

It can be noted that the steps above can be simplified if the height given is a vertical coordinate datum, in which case the geometries can be extruded to that set height.

4.2.4 New Building Insertion

The insertion of the new building can be done by translating its geometries to the x and y coordinates provided by the user. To raise it to the accurate height, a similar approach was taken as in the previous section. However, this process was performed manually by sinking the building by a small amount, to ensure that the model would be watertight. This approach is only valid when the terrain has a gentle, not bumpy landscape. For a rugged and steep landscape, this approach will need to be modified to properly bridge the gaps between building footprints and the terrain or other alternatives need to be taken.

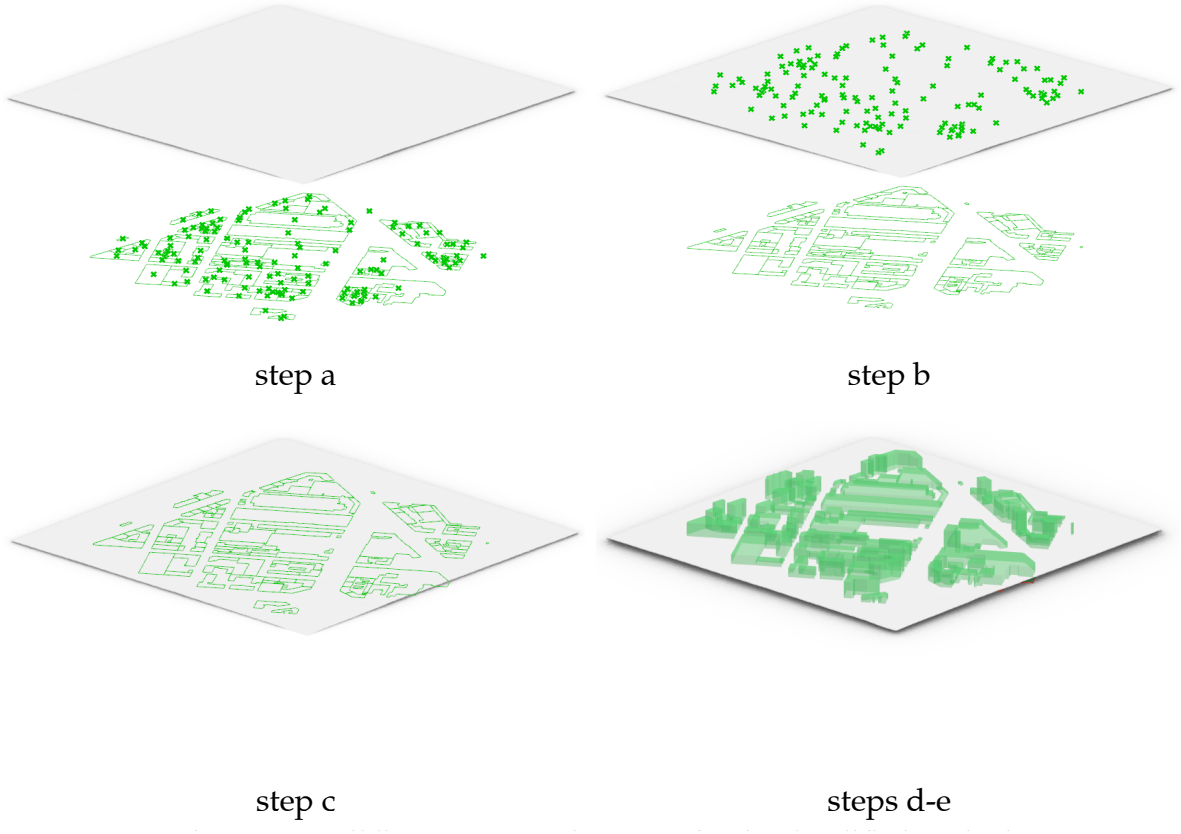


Figure 4.8: Building reconstruction steps for the simplified method

4.2.5 Results

The model shown in Figure 4.9 shows the resulting watertight model for the TIN case. Note that since which buildings fall in the area of interest is determined by the centre of each building footprint, it could let the buildings near the edges not be selected even if their footprint covers certain areas within the bounding box. Therefore, the users should set the bounding box extent larger than the area of interest to create a buffer, which guarantees that at least all the buildings near the newly inserted building are included and are watertight to the terrain within the desired area for further simulation. With the integrated model, the solar and wind experiments in Rhino, such as solar radiation, shadow calculations and wind analyses can then be performed.

4.3 Simulation Experiment

4.3.1 Solar Analysis

Various components are required to perform a solar analysis. The initial requirement is a specific date to perform solar analysis. To achieve the primary goal of preliminary environmental analyses this was limited to one day, however it is also possible to perform the analyses for larger stretches of time. Once the date is determined, it is necessary to search for an EnergyPlus Weather (EPW) file near the location, which contains the necessary location-specific weather data for the simulation. The reconstructed geometry through the TIN-based method of existing buildings, terrain and proposed future construction will be used in conjunction with the weather data to perform solar analysis. These must be in a Breps or mesh format to be used as input with Honeybee.

All these components were loaded into the Honeybee plugin within Grasshopper. The experiments will be conducted using this plugin, akin to the example depicted in Figure 4.10a.

The conducted experiments firstly assess the direct sun hours. This involves analysing the influence that new buildings have on the total hours of direct sunlight received by the surrounding areas. Sec-

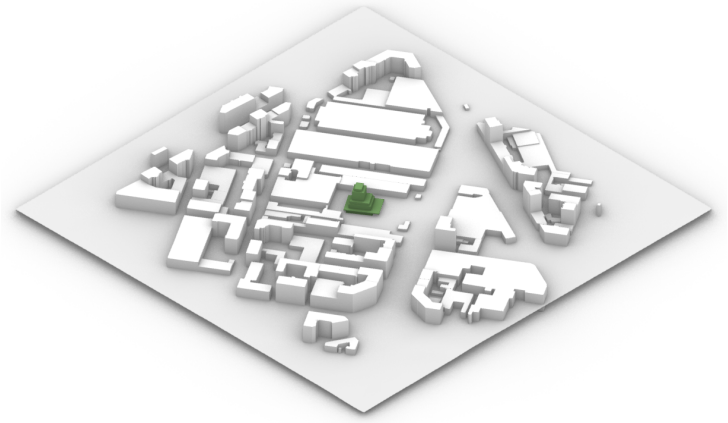
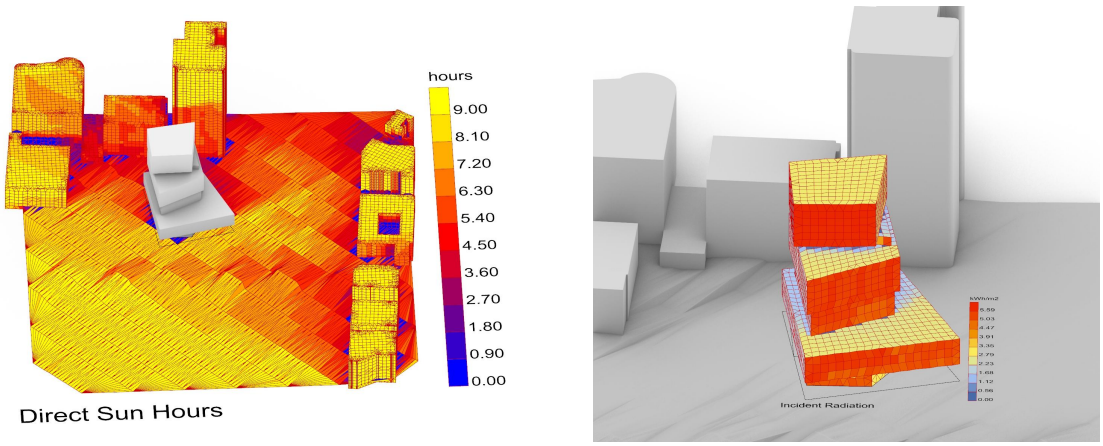


Figure 4.9: Resulting watertight model (the Italy case)

ondly, the analysis also focuses on the solar radiation received by the new building itself. It calculates the number of kilowatt-hours per square metre that fall on the facade, as shown in Figure 4.10b.



(a) Direct sun hour analysis

(b) Solar irradiation analysis

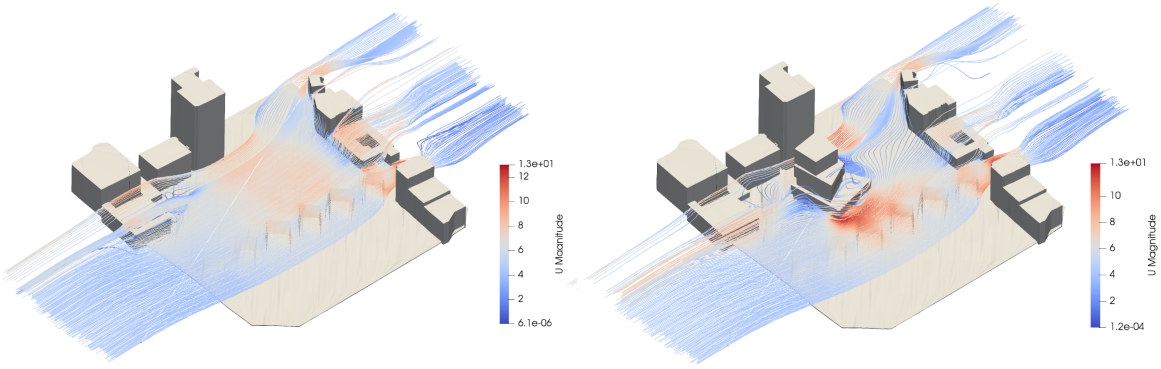
Figure 4.10: Results of Solar Simulation Experiment

4.3.2 Wind Analysis

The methodology and limitations in Section 3.2.2 regarding the use of Butterfly and OpenFOAM are followed when performing wind simulation with the generated geometry through the TIN-based method. The results of the simulation, as seen in Figure 4.11, show the wind force at a pedestrian level after the insertion of a new construction in a previously empty plot. Tow achieve this result from the reconstructed geometries in Grasshopper, the geometries generated via the TIN-based method require specific preparatory transformations. To comply with the requirements of Butterfly components, these input geometries must adhere to two conditions: they should be solids defined through Breps and be closed Breps.

The building geometries typically fulfil both conditions, as their reconstruction process ensures the creation of a closed solid representation without openings. However, the terrain model might necessitate certain transformations, particularly when in a mesh format.

The first transformation that needs to take place is converting the mesh into a Brep structure. This can be achieved by individually converting the mesh's triangles into Breps and subsequently uniting into a single Brep. Although this process generates a Brep, it does not yet ensure that it will be closed. An additional step is needed which involves providing the terrain a certain thickness and extruding it in the negative z direction by that amount. Upon the completion of these steps, the geometries



(a) Before insertion

(b) After insertion

Figure 4.11: Results of Wind Simulation Experiment

derived from the TIN case will conform to the Butterfly requirements and the wind simulation will be executed.

4.4 Region-Specific Implementations

4.4.1 TIN Case in Germany

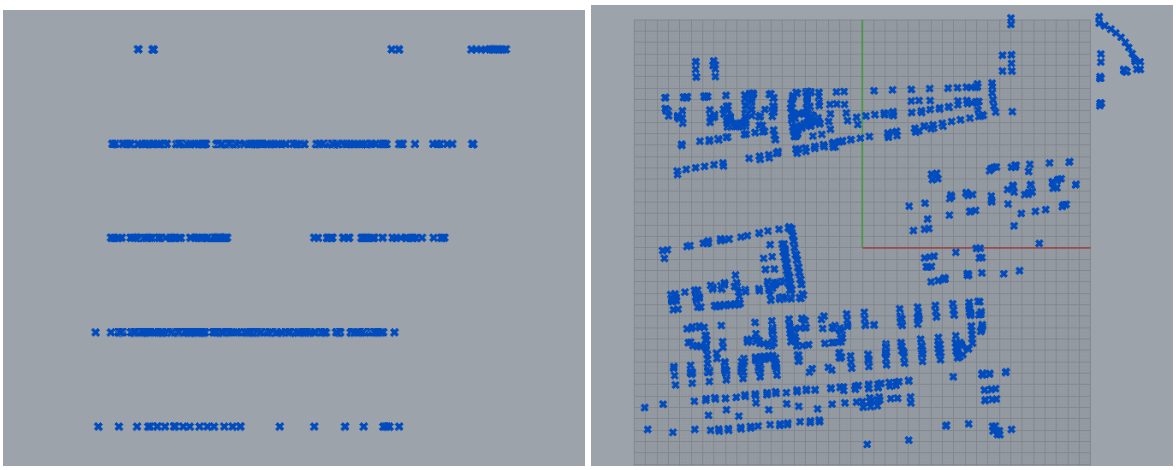
The data that is required for the contextual analysis (e.g. buildings in LoD2 and terrain) in Hamburg is delivered for the entire region, subdivided into different tiles. To preprocess this data to find the necessary tiles for the area of study, a Python script was created that would read through all of the different CityGML files, read out the bounds and return the file names of the files that would be relevant. Although this approach should have worked, the speed and memory efficiency of Python within Grasshopper was a limiting factor; the script would require more than 10 minutes to run at least with no guarantee that it would work at all. Therefore, the decision was made to learn some C# and to transform the Python code into something that runs more natively within Grasshopper. This led to the code working, resulting in the correct CityGML file names being filtered out. Since the point cloud (XYZ files) for the terrain had the same name as the CityGML tiles, this task was rather trivial.

Upon further inspection of the buildings dataset, it seems that the ground points supplied with the CityGML do not match the ground polygons of the buildings when using a modified version of the original implementation of the GML reader. However, an unlikely solution where the ground floor points are instead combined with the wall points instead of being separately processed lead to correct results (Figure 4.12). The reason for this is unknown and was not found due to time constraints, but despite this it was possible to generate watertight models that can be employed for both solar and wind analysis. Even so, it should be noted that since the terrain from the XYZ files matched up with the ground floors of the buildings which effectively would also create a watertight model, the original solution would have been usable for the simulations.

4.4.2 TIN Case in Hong Kong

The data retrieved via the API in Hong Kong is already watertight with all the DSM features represented in OBJ format, the wind simulation should be ideally available based on it. However, it will be challenging to achieve due to the following reasons:

1. The original OBJ tiles overlap with each other to a certain extent; how to seamlessly combine adjacent tiles is the main challenge for building a valid TIN - based model in the case of Hong Kong.
2. The insertion of a new building would be another challenging point with the original TIN case in Hong Kong, it will require a process similar to the CDT building regeneration method mentioned above to incorporate the new structure into the original context. According to our experiment, it could be too computationally expensive to achieve, which is not a viable solution for our target analyses.

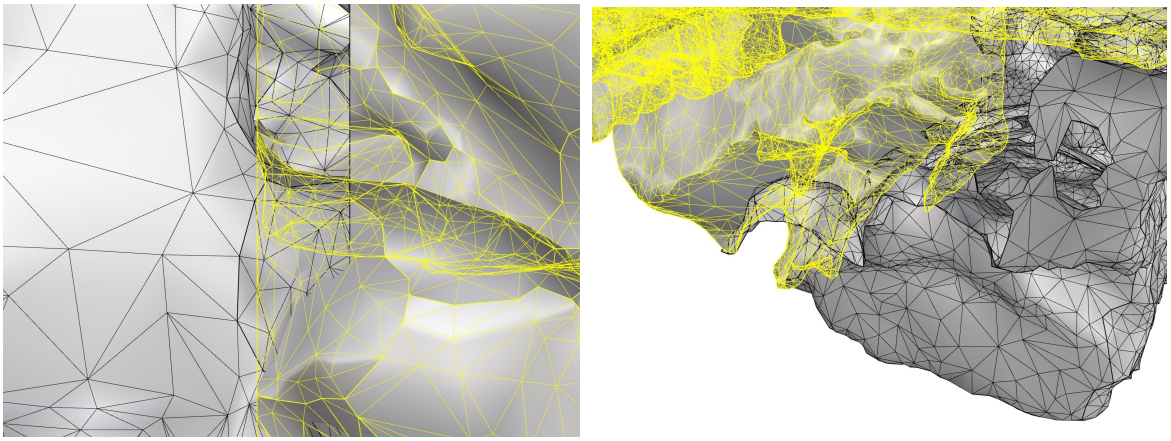


(a) Top View of points when separately processed (b) Top view of points when added to wall surfaces

Figure 4.12: Ground points of the CityGML when processed in the original and combined method

3. The resolution of the primitive model could be overly detailed for running wind simulation with the Butterfly plugin in Rhino. Certain simplification methods are needed to reduce the resolution of the model to run the wind simulation on one single tile. If the user's area of interest is larger than a single tile, the issues stated above will need to be taken into consideration, which makes the TIN case in Hong Kong difficult to apply.

Based on these reasons, our team turned to the voxelization method which is described in Chapter 5.



(a) Top view (b) Perspective view

Figure 4.13: Overlapping and Gaps between the Tiles

4.4.3 TIN Case in Italy

The Piedmont Geoportal served as the source for data used in implementing the TIN-based approach. The data was only available through direct download, requiring the user to establish a direct link to the Grasshopper file.

Building footprints were made available as a shapefile that divided the region into tiles. Among the fields corresponding to each footprint, a height value was given. This height did not correspond to a specific vertical coordinate datum but rather represented the height of the building measured from a ground point. For this reason, an assumption was necessary when reconstructing the building geometry to determine the starting point from which to extrude the solid. It opted to use the footprint centre's projection on the terrain as the reference height for the extruding. In the area considered for the simulation, this did not pose a great difference, as the terrain was mostly flat. However, in conditions where the terrain height differs greatly in small differences, this choice could lead to discrepancies

between the reconstructed building geometry and the real-life structures. Since no information was given regarding the methodology behind the original height value, a certain margin of error was unavoidable.

The terrain data was provided in the form of a TIFF file, which was also organised in tiles that spanned the region. To generate a TIN mesh, the file needed to be converted into an XYZ file format, so that a series of points could be used for the TIN mesh construction. Due to limitations with the Python component within Grasshopper, this conversion process was performed using an external Python script.

4.4.4 TIN Case in Spain

For Spain, the data is available from two different places. The first is the Geoportal for the terrain data and the second place is the Cadastre WFS, from which we download the building outlines. We use the terrain model to create the surface mesh. From the Cadastre data together with the surface model, we create the building solids as explained earlier in this chapter. The building solids and terrain data together form the context of the location. Then we load a new building as described in the previous sections.

4.4.5 TIN Case in the UK

The terrain mesh generation in the UK case shares the same method as the Italy case to convert TIFF to XYZ file with the aid of the external Python package. As for the building, the official free data portal only provides a coarse building footprint shapefile without height, which is not sufficient for creating LoD1 building and several footprints are dissolved into a whole, and the detail shapes are lost. In this case, the building footprints of the UK are retrieved from OpenStreetMap, and the heights are extracted via the "zonal statistic" plugin in QGIS (see Appendix A) to speed up the points in cures (here refers to building footprints) selection, the averaged heights are then addressed to each building footprint.

After the preparation, our team tried a more rigorous CDT approach mentioned in the Section 4.2.1 to correctly embed the building footprints to the terrain. The constraints within DT mesh help to solve the overlapping issue of two layers and enable the users to semantically label each of the CDTs, which is beneficial when there are more than two surface roughness values apart from only building and terrain in this case, each surface type could be identified and set with different parameters while running wind simulation. Moreover, the CDT approach also allows users to select and crop the 3D model as they desire, the buildings near the edge could be retained, which is another advantage compared to the simplified method in Section 4.2.2.

However, the finer the mesh, the more challenges we could encounter. The main challenges of CDT approach lie in the list order of CDTs layer and building footprint layer, as each footprint is divided into multiple pieces of CDTs, if the order is not consistent, each CDT will be extruded with the wrong height. To solve this, we first have to flatten the CDTs list to keep the building footprints list in order, we then select the CDTs in each building footprint and assign it with the correct height.

Apart from the height issue, the CDT approach requires heavy computation, which might not be viable in terms of timely interactive operation, and the output result may contain unknown errors such as invalid Breps after extrusion due to the environmental setting of Rhinoceros.

We have tested the CDT-generated UK model in wind simulation, and the result was not reasonable since the wind simulation unexpectedly stopped at the snappyHexMesh (Blocken [2015], García-Sánchez et al. [2021]) and collapsed. Based on the result, we have two assumptions considering the inducing reasons for the error:

1. **The shape of mesh**: CDTs can result in "sliver (super thin) triangles" near the cutting edge. It might introduce invalid geometry that causes the problem during the snappyHexmesh process.
2. **The inherent characteristics of Rhinoceros**: There might be some compatibility issues in conducting the simulation process with CDT-based models in Rhinoceros due to the inherent application capability. Moreover, since the CDT-based model is computationally expensive for reconstruction considering the detailed segmentation of the input model. All these possible factors may lead to the failure of further usage of the model and to implement simulations on a normal laptop.

Due to the time limitation, we could not figure out why the error occurred, it remains a task to solve for future work. Considering the reasons mentioned previously, we can say that the CDTs approach might not be suitable for generating the necessary model for wind simulation in Rhinoceros. Thus, using a simplified method for geometry generation or voxelizing the CDTs model could be a more applicable alternative in terms of wind simulation in Rhinoceros.

4.5 Evaluation of the TIN-based Construction Method

The TIN-based method allows for a direct recreation of geometries from available datasets. Among its advantages is the small loss in data through the reconstruction process, as the geometries remain mostly unaltered or simplified unless specified deliberately. This ensures that the building shapes, position and height faithfully represent the original dataset. The terrain undergoes the most substantial processing, as it is usually provided in a grid format, requiring interpolation for its 3D representation. Moreover, the process yields shorter processing times, which are mainly influenced only by the size of the original datasets, as the operations involved are generally simple. However, this same speed does not extend to the simulation processes. The potentially detailed nature of the TIN-based model may challenge the simulation's processing capabilities, negatively impacting processing time.

For these same reasons, this approach heavily relies on the quality and coherence, consistency among diverse datasets, including existing buildings, terrain data, new constructions, and other urban objects. One of the primary challenges when combining data through the TIN-based approach is ensuring proper alignment. Typically, alignment can be achieved when the data shares the same coordinate system, as is often the case, although there are instances where discrepancies arise, as observed in the cases of Italy's and Spain's height data.

An additional challenge involves the varying dataset resolutions. In many of the explored experiences, limited choices in available datasets lead to the integration of geometries with varying resolutions. Existing buildings often have a different LoD from the newly inserted ones, while terrain data usually employs a grid structure that determines its resolution. These differences might not pose a problem for certain users, such as architects seeking a general visualisation of the context in which to insert detailed project models. However, in cases such as solar and wind environmental analyses accuracy might be essential, and in these cases it could vary based on which area of the model it was performed in.

5 Implementation - Voxel Case

In this chapter, the Voxel-based implementation is introduced. First, the overview of the implementation is described. Then, in Section 5.1, Section 5.2 and Section 5.3, the data acquisition and alignment, geometry generation and simulation experiments are discussed in detail. Finally, a conclusion of this chapter is given as well as the limitations of the current implementation.

As mentioned in the Methodology Chapter, there are three sub-goals of this project. In this Section, the ideal input and output of the three parts are discussed below. The in-detailed steps (usually the most tricky parts) will be left in the later 3 Sections (The workflow diagram is shown in Figure 5.1).

1. The data acquisition and alignment are achieved by the API-based method, the implementation is achieved by Python script embedded in Grasshopper.
2. The geometry generation is achieved by the Voxel-based method, and the implementation is in C# (for efficiency reasons) in Grasshopper.
3. The simulation experiments are for Solar and Wind analysis, and the implementation is by the Radiance and OpenFOAM engines, Ladybug and Butterfly tools in Grasshopper, and Paraview is used for visualisation.

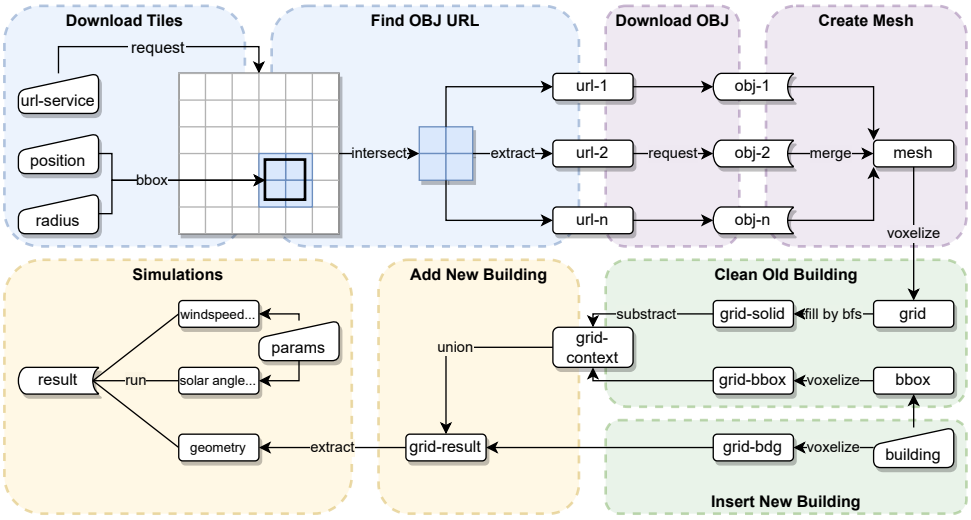


Figure 5.1: Implementation of Voxel Case

5.1 Data Acquisition and Alignment

After the data finding and researching processes, there may be an initial idea to manually download and load the data into the Rhino/Grasshopper. There may also be possibilities to automate the above process by using APIs. Datasets may be in different formats and organisations which may be requested by different types of APIs (OGC get/post or W3C get/post).

In this Section, a typical acquisition case for Hong Kong is introduced: W3C GET request is used to get the GeoJSON file and to further get the OBJ files. The main steps are described below.

5.1.1 GeoJSON Parsing

1. Because of the particularity (The function of specifying the bounding box of the tiles is not supported) of the Service Protocol, the whole GeoJSON file must be requested.
2. The requested GeoJSON file (in text form) could be transformed into a Python dictionary. There are geometries (4 coordinates of the rectangles) and corresponding downloading links for OBJ files for each tile.
3. Given the position (longitude and latitude of the place of interest) and the radius (size of the bounding box), the querying rectangle could be constructed.

4. Due to the particularity of the querying shape and queried shapes (They are axis-aligned rectangles), four point-in-rectangle tests could be used to detect if two rectangles are intersected.
5. One traversal is used to check the intersection between the querying rectangle, and all the queried (extracted) rectangles and the downloading links (in string form) for OBJ files are put into a list.

5.1.2 OBJ Downloading

1. Due to the big size of the OBJ files and the fact that they are zipped, it is safer and more convenient to first download them as temporary files (A temporal folder would be created).
2. By using the links for OBJ files obtained in the previous main step, Python requests are used to download the zip files into the temporal folder.
3. By using the suffix names (.obj) of the files in the zip files, the OBJ files are extracted from the zip files. The process is repeated until all the downloading links are used.

5.1.3 Mesh Loading

1. Because there is no local library in Grasshopper to load the OBJ files, another Python module is needed. A mesh list is created first as a container for all the results.
2. The OBJ files are read line by line and all the vertices and faces are extracted and organised in order which could be used to construct one mesh. Once one OBJ file is loaded, the mesh is appended to the mesh list.
3. All the meshes in the mesh list are joined to a whole. The mesh is moved from the real coordinate to the origin for the convenience of the AEC practitioners.

5.1.4 Results

After the main steps mentioned above, one mesh could be obtained, an example (one tile) is shown in Figure 5.2. It can be seen that the resulting mesh is too detailed and complicated which may lead to efficiency issues for later environmental simulations. If zooming in, it can also be found that there are overlappings, disjoints and gaps in the mesh (especially in the boundaries of tiles) which may lead to correctness issues for later environmental simulations.

An efficiency experiment[1] is also carried out to test the time consumption for the data acquisition which is shown in Table 5.1. It can be seen that there's a linear relationship between the number of tiles and the time cost for downloading. Note that the sides of a bounding box are equal to two times the size of the search radius.

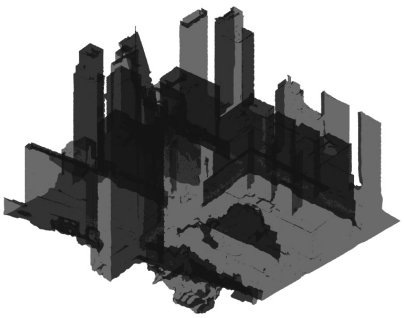


Figure 5.2: Result of Acquisition

1. When the search radius is small, the bottleneck of the data acquisition process is the network speed due to the big size of the OBJ files.
2. When the search radius is big, the bottleneck of the data acquisition process is the disk. For example, the 25 obj files in Table 5.1 account for 972 MB (nearly 1 GB).
3. When downloading the obj files, the main memory is not the bottleneck as the obj files are downloaded one by one. However, it is untested if the main memory could be the bottleneck when loading the obj files as all the files should be loaded together.

[1] Device: Lenovo ThinkBook 14 G5+ IRH; CPU: 13th Gen Intel(R) Core(TM) i7-13700H 2.40 GHz, RAM: 32.0 GB; OS: Windows 11 Family.

Table 5.1: Acquisition Time

Search radius (m)	Tile number	Time cost (s)
10	1	24.0
20	2	34.2
40	2	27.8
80	4	45.5
160	9	128.9
320	25	254.8

5.2 Geometry Generation

After the acquisition step, the 3D digital surface model(s) (represented by mesh) of the context could be obtained. They are detailed and fit well with the real terrain, buildings, trees etc. However, there are overlappings, disjoints and gaps, especially on the boundaries of tiles. In this Section, the voxelisation method is used to fix the mesh. In the meantime, one new building is inserted (operations developed for users) into the context to create different scenarios.

5.2.1 Building Preparation

1. The new building should be placed in the proposed place of the context, it could be an internal or external model in Rhino or a parametric model in Grasshopper.
2. It is better for users to first place the model near the context (In Rhinoceros), then use the movement parameters (In Grasshopper) to place the model in the context and set the original model invisible (shown in Figure 5.3).
3. It is better to model the base of the building for later convenience (used for creating the bounding box to remove original voxels). More operations for the buildings are not provided in this workflow such as rotations and scaling which could be carried out before the placement step by users if they like.

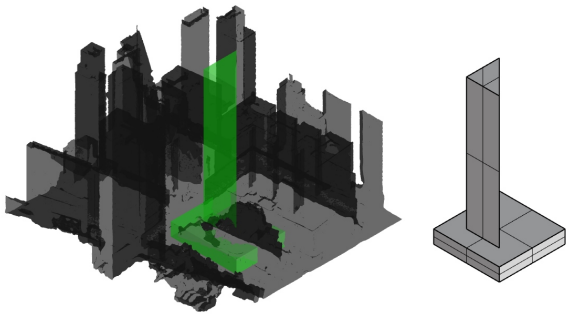


Figure 5.3: Building Preparation

5.2.2 Context Repairment

1. Create the bounding box of the context and the building (Use the max and min of the context and the building), given the resolution, the Grid (voxels) could be created by the VoxelTools Plugin in Grasshopper.
2. The places and numbers of the voxels will not be changed. Later operations are only based on the boolean values (True or False representing the filled or empty) of the voxels.
3. The Breadth First Search (BFS) Algorithm (the 2D example is shown in Figure 5.4) is used to fill the voxels underground making the grid a solid one. This is used to ensure the consistency of the later insertion of the building.

5.2.3 Models Integration

1. Convert the surface to the voxelized 3D grid.
2. Get the bounding box of the building.
3. Use the bounding box to clip the context.

4. Insert the building into the context and note that there may be possible non-watertight issues (shown in red circle).
5. Therefore, instead of directly executing step C, it is better to first fill the underground with the BFS algorithm.
6. Use the bounding box to clip the context.
7. Insert the building into the context and note that there's no non-watertight issue anymore.

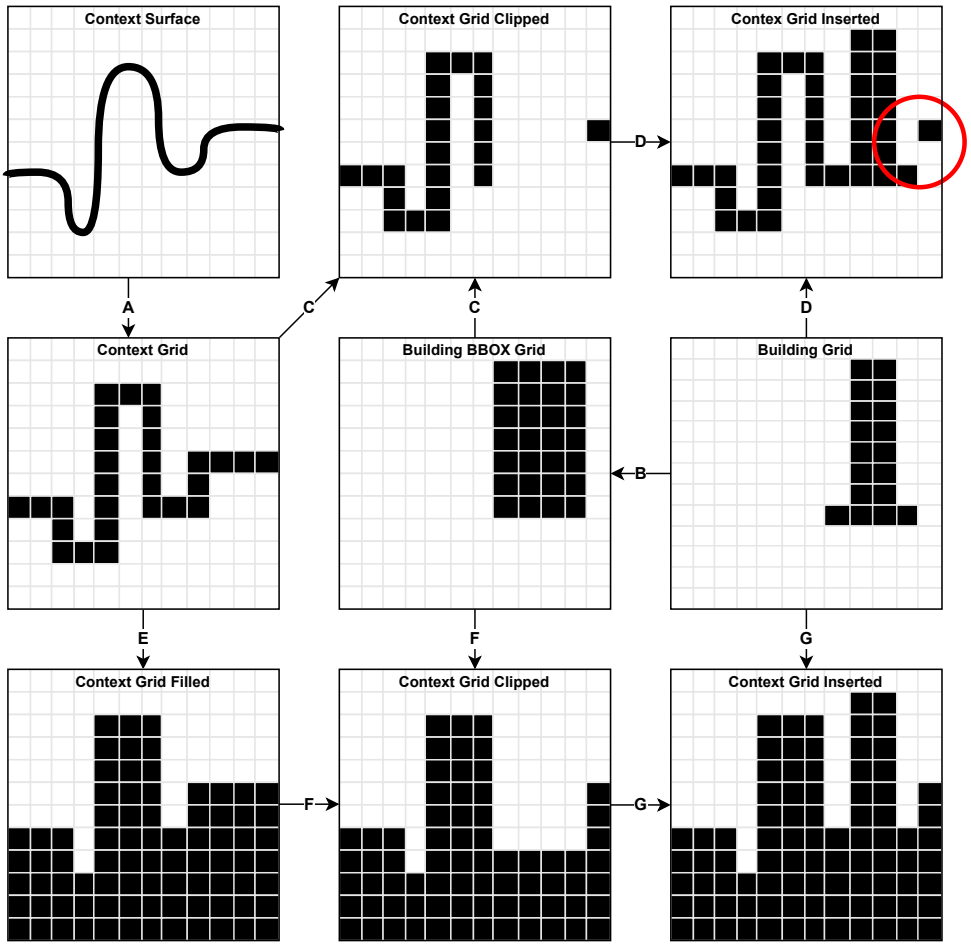


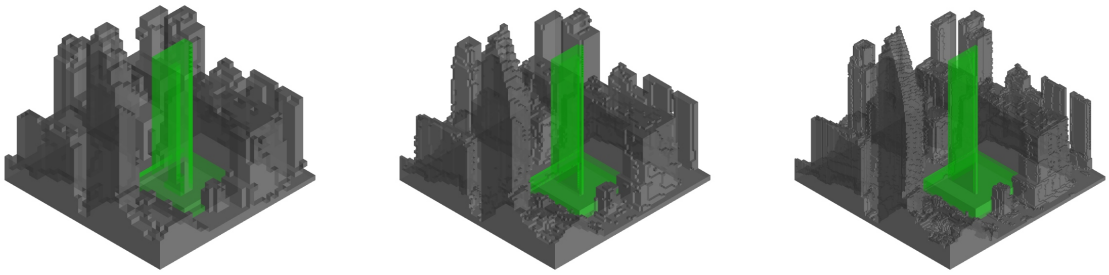
Figure 5.4: The Reasons for BFS

5.2.4 Results

The model shown in Figure 5.5 shows the resulting models of a tile with different resolutions. With the increasing resolution, the models are smoother and fit better than the original model. Also, it can be seen that the artefacts mentioned above can all be solved and the details of the model are still kept. It can be seen that the result with 2m×2m×2m resolution is already acceptable for wind simulation by visual interpretation (Blocken [2015], García-Sánchez et al. [2021]).

An efficiency experiment (shown in Table 5.2) is carried out to test the time consumption change with the increasing tile number and resolution as controlled parameters. It can be seen that there's a linear relationship between the tile number and time consumption and an exponential relationship between the resolution and time consumption.

The bottleneck here is the main memory. When the resolution is too high, the memory used when doing the breadth-first search would reach the memory limitation of Rhinoceros (2GB) which would cause the Rhino to crash: frequent exchange of information between memory and disk severely lowers the running speed (thus, it is better that users could first try the lower resolution).



(a) 4m×4m×4m Resolution (b) 2m×2m×2m Resolution (c) 1m×1m×1m Resolution

Figure 5.5: Results of Alignment

Table 5.2: Voxelisation Time

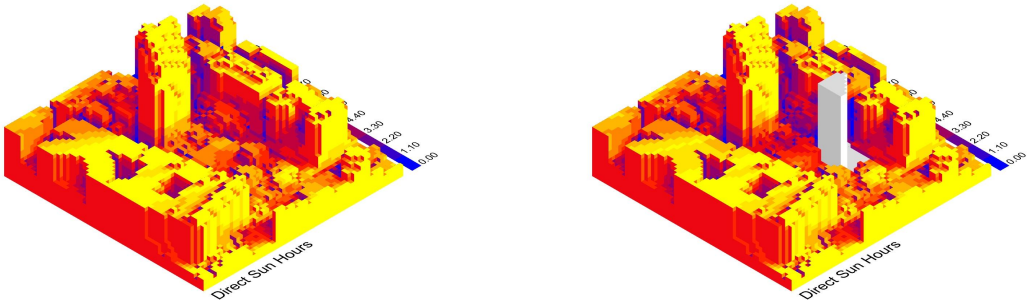
Tile number	Resolution (m×m×m)	Time cost (s)
1	4×4×4	≈0
1	2×2×2	14.0
1	1×1×1	109.9
4	4×4×4	6
4	2×2×2	49.3
4	1×1×1	Out of memory
9	4×4×4	15.2
9	2×2×2	116.0
9	1×1×1	Out of memory

5.3 Simulation Experiment

After the geometry generation step, a 2-manifold could be obtained which is the geometry input of the environmental simulations. Similar to the TIN-based method, an MRE is proposed using mostly the default or automatically generated parameters as an experiment.

5.3.1 Solar Simulation

The comparison of the solar simulation results before and after the insertion of a new building could provide insight into the effectiveness of the mesh models. The results are visualised in Rhino which are shown in Figure 5.6. The faces are colourised by the direct sun hours.



(a) Before Insertion (b) After Insertion

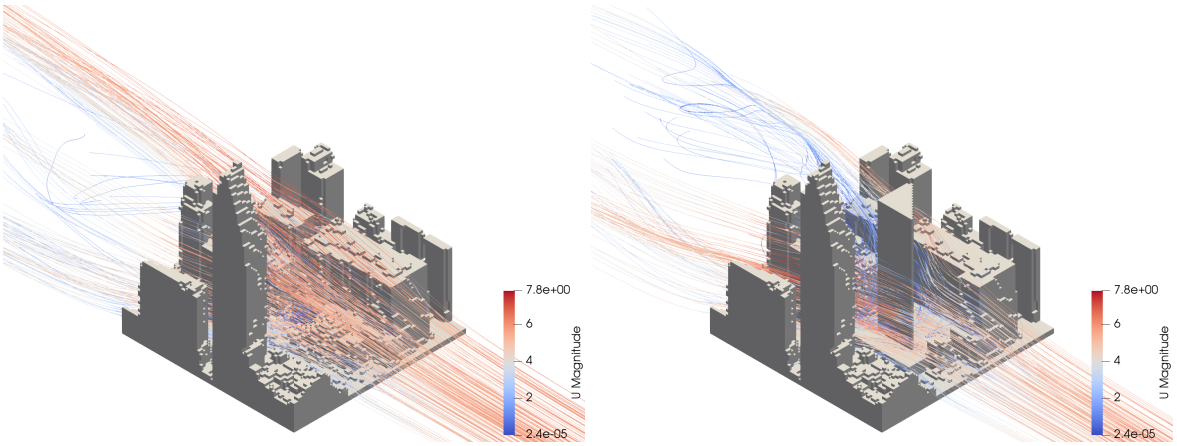
Figure 5.6: Results of Solar Simulation Experiment

It can be seen that adding a new tall building will shadow some areas, creating places lacking sunshine. In some cases, especially when building a tall building in the south (In the case of the Northern

Hemisphere) of a residential building, the sunshine needs of the residents would not be met.

5.3.2 Wind Simulation

The comparison of the wind simulation results before and after the insertion of a new building could also provide insight into the effectiveness of the mesh model. The results are visualised in Paraview which are shown in Figure 5.7. The Stream Tracer is used to show the wind flow, the tracers are colourised by the magnitude of the wind speed.



(a) Before Insertion (b) After Insertion

Figure 5.7: Results of Wind Simulation Experiment

It can be seen that adding a new tall building will move the areas with higher wind speeds from top to bottom, creating a winding canyon which negatively influences the wind comfort for pedestrians. At the same time, an eddy will form on the leeward side of the inserted building which may negatively influence the dispersion of the pollutant particles.

An efficiency experiment is also carried out to check the running times (Only running time, the time for blockmesh and snappyHexMesh are not considered) with different numbers of tiles. The results are shown in Table 5.3.

Table 5.3: Wind Simulation Time

Search radius (m)	Tile number	Time cost (s)
10	1	100.3
80	4	152.5
160	9	214.4

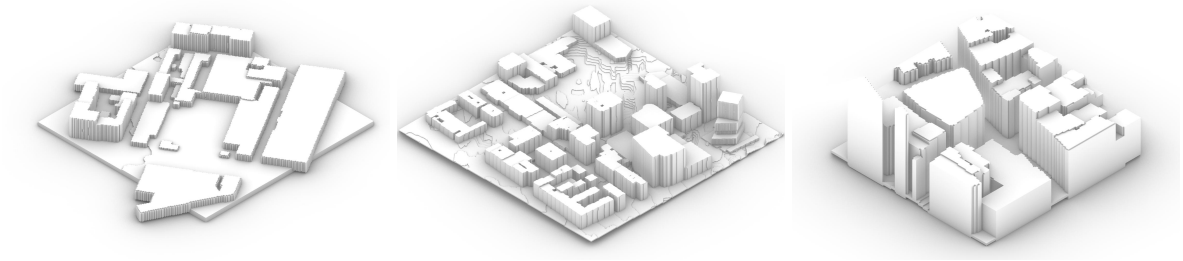
It can be seen that the increasing number of tiles would not significantly increase the running time. On the one hand, parallel computing makes it difficult to analyse where the bottlenecks are, on the other hand, the complexity (mainly accounted by the number of faces) of the final mesh may not form a linear relationship with the original size of the model (many other parameters would influence the mesh size).

5.4 Region-Specific Implementation

Tests are carried out for the inputs from all the places (Spain, the UK and Italy) and their voxelised results are shown in Figure 5.8 which demonstrates the broad applicability of the voxel approach.

As for the Hong Kong case with the integrated detailed terrain and building models in one file, the voxelisation method works well to align and generate proper geometries for later environmental simulations.

As for other cases (Germany, Italy, Span and the UK) with separate terrain and building models, the voxelisation method could act as a "simplifier" after the TIN-based steps, although reducing some accuracy, making the models simplified and regular which may be safer for later simulations.



(a) Italy (b) Spain (c) UK

Figure 5.8: Region-Specific voxelisation Results

5.5 Evaluation of the Voxel-Based Method

In general, the voxel-based method can easily convert various vector model inputs into models that meet the needs of real-time environmental analysis and simulation (especially when the TIN-based method is not applicable or suitable). This method is suitable for data format input in various types and regions. At the same time, users can edit the context and insert new structures very flexibly.

The challenge comes from the efficiency issues caused by too fine voxelization resolution, especially when Rhinoceros itself limits memory usage. However, even under this limitation, the accuracy of the obtained voxel model can already meet the customer's requirements.

6 Conclusion

6.1 Summary

This report has outlined the process of developing an open-data-based tool that builds upon the original interface created by Royal HaskoningDHV, designed to bridge the gap between geographical data and AEC industry applications. The focus of the project was to transform spatial data as it is known to geomatics experts, into workable data for architects for contextual analysis in Rhinoceros and Grasshopper. In the end, the purpose of the tool is to facilitate architects and engineers in improving their designs based on the environmental impact that these may have. The original tool provided worked only on data within the Netherlands; however, the end goal was for the tool to be usable for other countries/regions as well. The research consisted of multiple phases, such as the evaluation of the data available for different countries/regions, the acquisition and alignment of relevant data into usable input for Grasshopper, and finally the implementation of these data workflows into wind and solar analyses.

During the data evaluation stage, it became evident that geographic data availability and accessibility tend to vary greatly across different countries and regions, leading to significant challenges in developing a universally applicable tool. Data in Germany for example is fragmented by administrative regions which requires end-users to navigate different data portals and delivery methods. On the other hand, data from Hong Kong is much more centralised and accessible via API, which is more conducive for integrating into a tool. In any case, the lack of standardisation between countries/regions often requires manual data retrieval strategies that would hinder the automation of data integration, making it less viable for end-users such as architects who may have a limited understanding of geodata.

Data alignment methods would also vary greatly depending on the data itself with different results. For instance, data in Italy required extruding 2D shapefiles of the building footprints into 3D using height values given with footprints. However, the lack of explanation of how this height value was originally gathered, as well as inherent variability in the terrain combined with the method for building reconstruction meant that an unavoidable margin of error would be present. In a different place like Spain, the extrapolation of the heights of buildings were extrapolated using a digital surface model instead. The differences in the applied techniques highlight the difficulty of a one-size-fits-all solution, due to the inherent lack of data standardisation and interoperability.

In the end, two techniques were envisioned for the open-data tool; the TIN-based method and the Voxel-based method. Both of these methods have different qualities. For example, the TIN-method is the truest to the input data and as such may field much higher quality analyses. However, this variant also relies heavily on the quality and interoperability between datasets, often requiring heavy data alignment before different datasets can be used. On the other hand, the Voxel-based method is more conducive to different data types and allows end-users to insert new structures very flexibly. However, possible issues could arise from using too fine voxelisation resolution, and the results in terms of analysis may also be affected depending on the resolution chosen.

6.2 Limitations of exploratory analysis

Although exploration of data was performed for five of the countries/regions of study, our analysis of what may constitute roadblocks for an open-data-based tool is by no means exhaustive for other countries/regions that may deliver and organise their data by different standards.

Another limitation of our research lies within the inherent characteristics of Rhinoceros. Since the current version of Rhinoceros uses an older subset of Python (namely IronPython 2.7) in a closed-off environment where the usage of external libraries is limited, many of the tools that GIS experts would use to transform data were unavailable. This combined with Rhinoceros's memory limitations meant that sometimes alternative and more obtuse rounds were required to do the same thing, such as translating Python code into C#, which none of the authors are proficient with.

Finally, the data found during the exploratory phase is not exhaustive; there is a possibility that datasets were overlooked due to language barriers, the permeability of data platforms and similar factors.

6.3 Future Work

Since these are the first iterations of workflows in data transformation, the code used may benefit from extra passes in optimisation, both to improve readability and to improve performance.

Although the current iteration of the data workflows only includes what is strictly necessary for the contextual analysis (e.g. LoD2 buildings and terrains), AEC professionals may profit from the implementation of additional types of data such as on vegetation, land use, transport and so on.

This report has primarily focused on exploring what is necessary to combine different data-flows. Further improvements in the implementation of an open-data tool could profit from input from actual-AEC professionals by the use of methods such as questionnaires or moderated/unmoderated testing.

6.4 GitHub repository

The grasshopper files of all the cases mentioned in this report can be found at the following links: https://github.com/biscuittsai1022/Synthesis-project_1-repository_2023.

A Data acquisition with Python package and QGIS

A.1 Codes for conversion of GEOTIFF to XYZ file

The code below shows the conversion from the GeoTIFF file to the XYZ file with the Python package "rasterio" (Gillies [2019]), which can be executed by IDE like Pycharm.

```
import rasterio

def tif_to_xyz(input_path, output_path):
    # Open the raster dataset
    with rasterio.open(input_path) as src:
        # Create a file for writing XYZ coordinates
        with open(output_path, 'w') as xyz_file:
            # Loop through all rows and columns of the raster
            for row in range(src.height):
                for col in range(src.width):
                    # Read the elevation value (z) at the current pixel
                    z_value = src.read(1)[row, col]
                    # Get the (x, y) coordinates of the current pixel
                    x, y = src.xy(row, col)
                    # Write the (x, y, z) coordinates to the XYZ file
                    xyz_file.write(
                    "{:.2f} {:.2f} {: .2f}\r\n".format(x, y, z_value))
    print("XYZ coordinates have been written to 'roi.xyz'.")

def main():
    tif = 'tiff file directory path'
    roi_xyz = 'Output directory path'
    tif_to_xyz(tif, roi_it_xyz)

if __name__ == '__main__':
    main()
```

A.2 QGIS operation for building heights extraction

The following screenshots show the process to perform the "zonal statistics" operation in QGIS. In this project, the external process is used only for the case in the UK since the building data retrieved from the local portal does not contain height information. The input data used here include the DSM GEOTIFF file and building shapefile extracted from OpenStreetMap (see Section 3.1.1).

1. Import the target building shapefile and DSM GEOTIFF file in QGIS.
2. Search for "zonal statistics" in the toolbar.
3. Set the building footprint as the input layer, DSM as the raster layer.
4. Click on the button beside the statistic to calculate the row and select the desired statistical attributes. In this case, the mean is selected to compute the averaged z-values within each building footprint.
5. Click run and export the resulting layer as a new shapefile

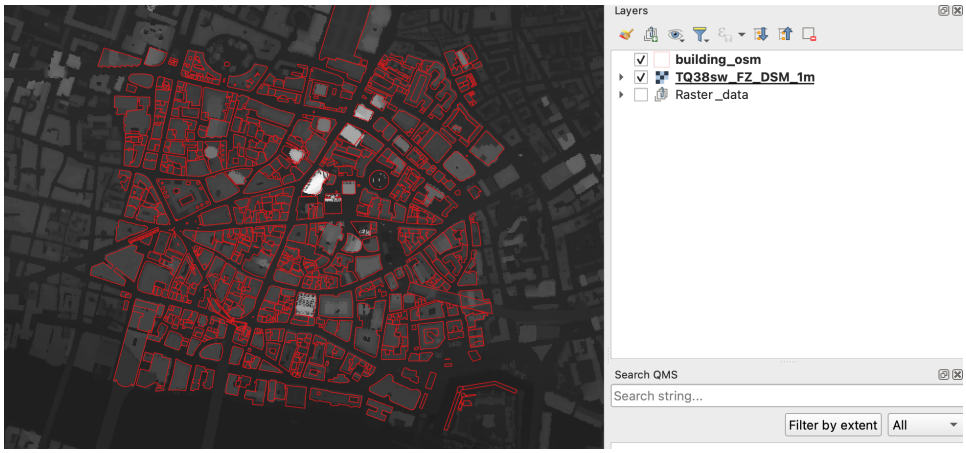


Figure A.1: Import the building footprint and DSM raster data

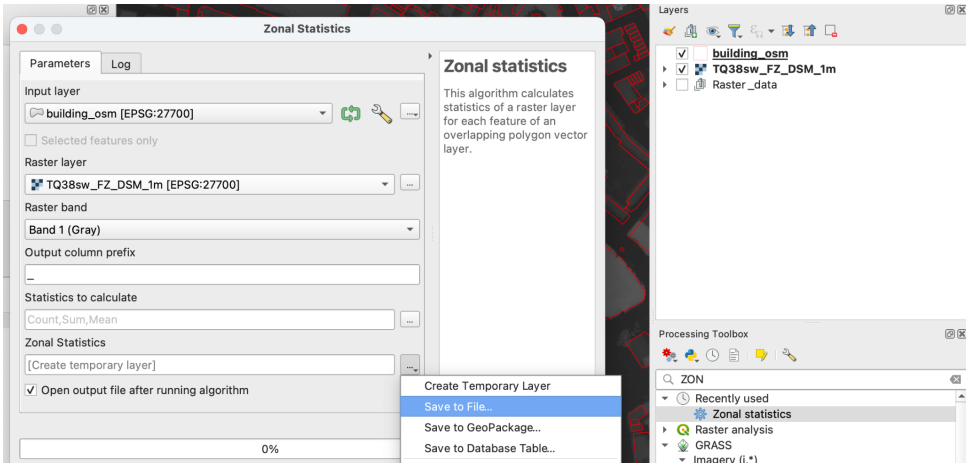


Figure A.2: Input and Raster layer setting

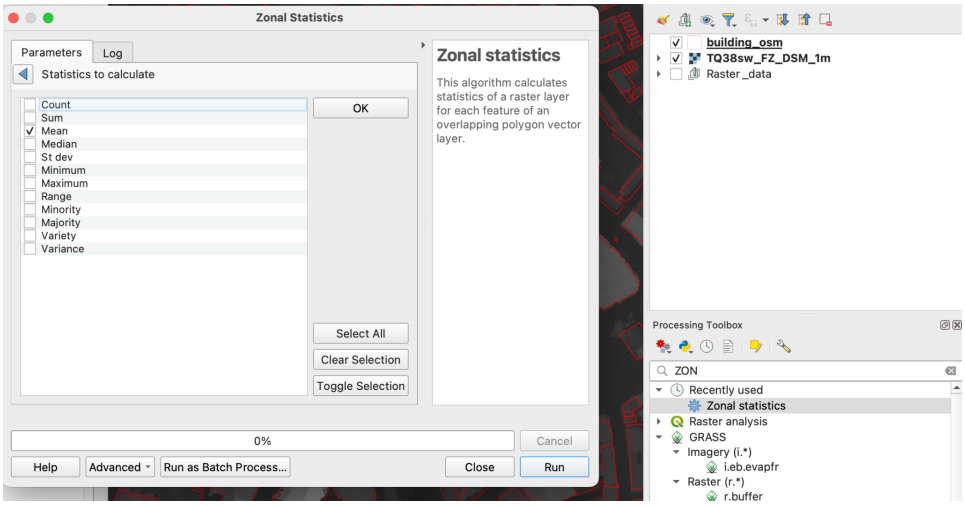


Figure A.3: Select the desired attribute

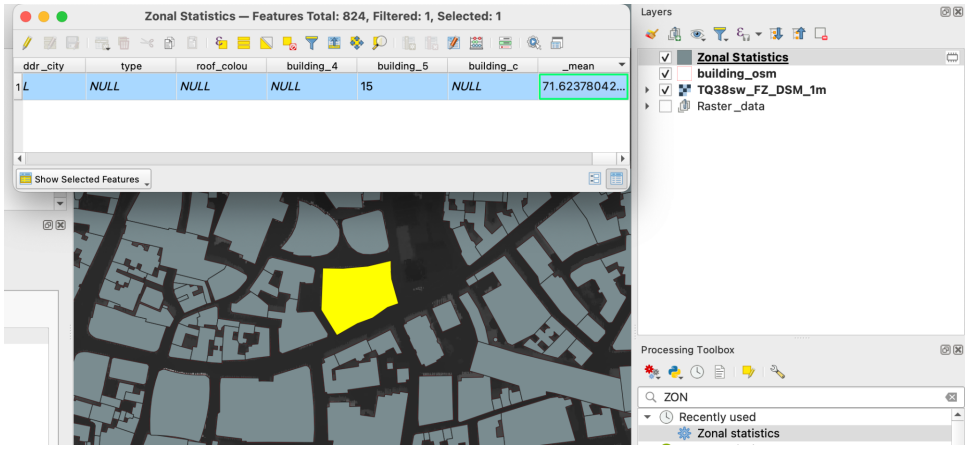


Figure A.4: Building's height stored in the last column

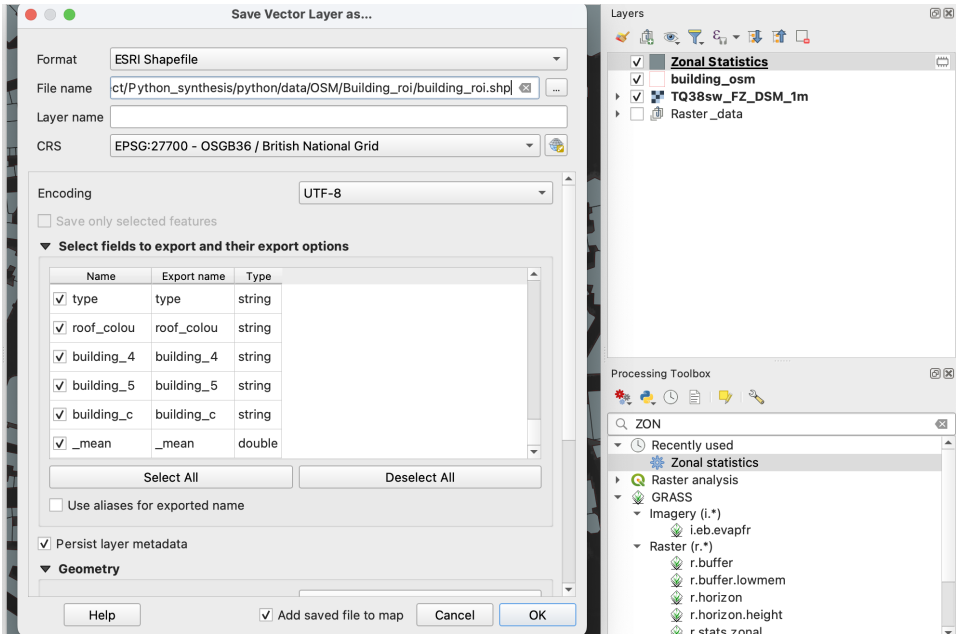


Figure A.5: Export the enriched building footprints to shapefile

B Data finding statistics tables

This appendix shows the detailed results in data finding mentioned in Chapter 2, including 10 countries/regions, starting from Hong Kong to Ireland. The corresponding links of different places for data acquisition mentioned in Section 3.1 are also included.

	Type	Name	Format	API	Detail	Coverage	Other
Hongkong	Building	3D Spatial Data 3D-BIT00	3DS/FEX/MAX/VRML	WFS/WMS	3D LoD2 Model	Fully	Tiling Indexes of the shapes; Files are separate for one tile
		3D Visualisation Map (2017)	OBJ	WFS/WMS	3D LoD2 Model	Partially	Tiling Indexes of the shapes; Phtogramatry
		Building	GML etc.	WFS/WMS	2D Vector	Fully	All the data at once; With floor attribute; Too big for a normal PC to load
	Landuse	2022 Raster Grids on Land Utilization	GEOTIFF	WMS	2D Raster	Fully	All the data at once
		Digital Topographic Map iB1000	GML etc.	WFS/WMS	2D Vector	Fully	Tiling Indexes of the files
	Terrian	Digital Terrain Model (DTM)	GEOTIFF	WMS	2D Raster	Fully	All the data at once
	Orthophoto	Digital Orthophoto DOP5000	GEOTIFF	WFS/WMS	2D Raster	Fully	Tiling Indexes of the files
	Road	Road Centreline	GML etc.	WFS/WMS	2D Vector	Fully	All the data at once
		Road Network	GML etc.	WFS/WMS	2D Vector	Fully	All the data at once
CSDI Portal - Common Spatial Data Infrastructure (CSDI)							

	Type	Name	Format	API	Detail	Coverage	Other
Finland	Building	Buildings 3D	CityGML	Must first buy (is it free) and then download	3D LoD2 Model	Partially	
	Landuse	Topographic map (raster)	PNG		2D Raster	Fully	
		Topographic map (vector)	Shapefile		2D Vector	Fully	
	Terrian	Elevation model	GEOTIFF		2D Raster	Fully	
		Laser scanning data	LAZ			Fully	
	Orthophoto	Orthophoto	JPG2000		2D Raster	Fully	
	Cadastral	Cadastral index map (vector)	Geopackage		2D Vector	Fully	
https://asiointi.maanmittauslaitos.fi/karttapaikka/							

	Type	Name	Format	API	Detail	Coverage	Other	AP
United Kindom	Building	3D Model	3DS/DWG/4CS/SKP	- (download link)	3D LoD1-3 Model	Fully	Tiling Indexes of the shapes; Various licensed data providers and all data is chargeable	https://shop.centremaps1ive.co.uk/map.php#
		osfeatures:Zoomstack_(District/Local)Buildings	SHP(exportable to other formats)	WFS	2D Vector	Fully	All the data at once; With shape area attribute but no height ; Too large to process	https://api.os.uk/features/v1/wfs?key=ClUBAysXtDE7W09uC3izGRJ3rfJfAv8u
	Landuse	Land Cover Map 2021	PNG	WMS	2D Raster	Fully	All the data at once; Enviromental perspective	https://catalogue.ceh.ac.uk/maps/2ad19a50-b940-469e-a40d-17818b77020c?cache=false
	Terrian	LIDAR DTM-2022	TIF/laz	- (download link)	2D Raster/Point cloud	Partially	Tiling Indexes of the shapes;	https://environment.data.gov.uk/DefraDataDownload/?Mode=survey
		OS Terrain® 50	SHP/GML/Geopackage/Vector tile(MBtile)	- (download link)	2D Vector	Fully	All the data at once; Contour line only	
	Orthophoto	Vertical Aerial Photography Tiles RGB	ECW	- (download link)	2D Raster	Partially	Tiling Indexes of the shapes; quite old	https://environment.data.gov.uk/DefraDataDownload/?Mode=survey
	Road	osfeatures:Zoomstack_Roads	SHP(exportable to other formats)	WFS/WMS	2D Vector	Fully	All the data at once; 3 classes(National, Regional and local); Too large to process	https://api.os.uk/features/v1/wfs?key=ClUBAysXtDE7W09uC3izGRJ3rfJfAv8u
		oproad_essh_gb	SHP/GML/Geopackage/Vector tile(MBtile)	- (download link)	2D Vector	Fully	Tiling Indexes of the files	
	Cadastral	National Polygon dataset	SHP	- (download link)	2D Vector	Fully	Tiling Indexes of the files, chargeable	

	Type	Name	Format	API	Detail	Coverage	Other
Taiwan	Building	3D Model	KMZ,I3S,3D Tiles	- (download link)	3D LoD1-3 Model	Partially	Tiling Indexes of the shapes; Chargeable
		Taiwan e-map BUILD	SHP(exportable to other formats)	WFS	2D Vector	Fully	Needs to apply for WFS and only for research use
	Landuse	Land Use Investigation (LUI) Map	PNG / SHP(exportable to other formats)	WMS / WFS	2D Raster / 2D Vector	Partially	Needs to apply for WFS and only for research use
	Terrian	LIDAR DTM	-	-	Point cloud	Partially	Under construction, limited access
		1/1000 topographic map	DWG / PNG	- (download link) / WMTS	2D Vector	Fully / Partially	CAD data is chargeable
	Orthophoto	Orthophoto	PNG / GeoTiff	WMS / - (download link)	2D Raster	Fully / Partially	Tiling Indexes of the shapes; Chargeable
	Road	3D road	KML	- (download link)	3D LoD1 Model	Partially	Testing, needs to apply for data download
		Road network	SHP	- (download link)	2D Vector	Fully	
	Cadastral	National Polygon dataset	DWG / SHP /PNG	- (download link) / WFS / WMS	2D Vector	Partially	Tiling Indexes of the files, chargeable, needs to apply for WFS and WMS
https://maps.nlsc.gov.tw/							

https://maps.nlsc.gov.tw/

https://maps.nlsc.gov.tw/S09SOA/

3.4 Product chunking

Ordnance Survey divides Great Britain into squares of 100 km by 100 km. Each of these squares has a unique two-letter reference, for example, TG in Figure 2 below.

To ensure that file sizes are manageable, GML and shapefile files are supplied as 100 km-by-100 km tiles.

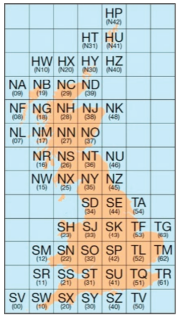


Figure 2. UK map showing 100 km-by-100 km tiles used to chunk up the OS Open Roads GML and shapefile data.

	Type	Name	Format	API	Detail	Coverage	Other	
Spain	Building	Building outline from cadastre	GML	WFS	2D	Fully		https://www.catastro.minhap.es/webinspire/index_eng.html
	Cadastre	Parcels from cadastre	GML	WFS	2D	Fully		https://www.catastro.minhap.es/webinspire/index_eng.html
	Landuse	Land Cover Map 2018	GDB or Geopackage	Download	2D	Fully	Complete country	http://centrodedescargas.cnig.es/CentroDescargas/index.jsp
	Road	Transport networks	SHP	Download	2D	Fully	Per region	http://centrodedescargas.cnig.es/CentroDescargas/locale?request_locale=en
	Terrain	DSM of buildings and vegeation	COG (Cloud Optimized GeoTIFF)	Download	2.5D raster	Fully	Download per tile	http://centrodedescargas.cnig.es/CentroDescargas/locale?request_locale=en
		DSM of buildings	ASCII (.asc) ESRI arrayfile	Download	2.5D raster	Fully	Download per tile	http://centrodedescargas.cnig.es/CentroDescargas/locale?request_locale=en
		DTM	ASCII (.asc) ESRI arrayfile or COG	Download	2.5D raster	Fully	Download per tile in 2, 5, 25 or 200 m grid spacing	http://centrodedescargas.cnig.es/CentroDescargas/locale?request_locale=en
		Lidar coverage file	LAZ(LAS)	Download	pointcloud	Fully	Per area of 2x2 km	http://centrodedescargas.cnig.es/CentroDescargas/locale?request_locale=en

	Type	Name	Format	API	Detail	Coverage	Other	
France	Building	Building outline from cadastre	GML	WFS	2D	Fully		https://geoservices.ign.fr/services-web-experts-parcellaire
	Cadastre	Parcels from cadastre	GML	WFS	2D	Fully		https://geoservices.ign.fr/services-web-experts-parcellaire
	Landuse	Land Cover Map multiple years	image	WMS	2D	Fully		https://geoservices.ign.fr/services-web-experts-ocsge
	Road	Transport networks	GML	WFS	2D	Fully		https://geoservices.ign.fr/services-web-experts-cartovecto
	Terrain	DSM	image	WMS	2.5D raster	Fully		https://geoservices.ign.fr/services-web-experts-altimetrie
		DTM	image	WMS	2.5D raster	Fully		https://geoservices.ign.fr/services-web-experts-altimetrie
		Lidar coverage file	COPC (Cloud Optimized Point Cloud)	Download	pointcloud	Fully	Per tile	https://geoservices.ign.fr/lidarhd

	Type	Name	Format	API	Detail	Coverage	Other	
Italy	Building	Building footprints (Piedmont)	SHP	Download	2D polygon	Fully (Piedmont)		https://www.geoportale.piemonte.it/geonetwork/srv/eng/catalog.search#/metadata/r_piemon:da9b12ba-866a-4f0f-8704-5b7b753e4f15
	Transport	Roads (Piedmont)	SHP	WFS	2D line	Fully (Piedmont)		https://www.geoportale.piemonte.it/geonetwork/srv/eng/catalog.search#/metadata/r_piemon:da9b12ba-866a-4f0f-8704-5b7b753e4f15
		Train lines (Piedmont)	SHP	WFS	2D line	Fully (Piedmont)		https://www.geoportale.piemonte.it/geonetwork/srv/eng/catalog.search#/metadata/r_piemon:da9b12ba-866a-4f0f-8704-5b7b753e4f15
	Terrain	Terrain	DTM 5m	image	Download	2.5D raster		http://wms.pcn.minambiente.it/ogc?map=/ms_ogc/WMS_v1.3/raster/DTM_20M.map
		DSM 5m	image	Download (after email request)		Fully (Piedmont)		
		DTM 20m	image	WMS	2.5D raster	Fully		http://wms.pcn.minambiente.it/ogc?map=/ms_ogc/WMS_v1.3/raster/DTM_20M.map
		DTM 40m	image	WMS	2.5D raster	Fully		http://wms.pcn.minambiente.it/ogc?map=/ms_ogc/WMS_v1.3/raster/DTM_20M.map

	Type	Name	Format	API	Detail	Coverage	Other	
Australia	Building	3D model Victoria	?	/	3D Lod 1-2 Model	Fully (Victoria)	Only accessible by email	
		3D model Brisbane	?	/	3D Lod 1 Model	Fully (Brisbane)	Only accessible by email	
	Cadastre	Parcels from cadastre						
	Landuse	Land Cover Map	image	WMS, WMTS		Fully		
		land use of Australia	image	Download		Fully		Catchment scale land use of Australia – Update December 2020 - DAFF (agriculture.gov.au)
		Land Zoning (NSW)	json	WFS, WMS		Fully (NSW)		Environmental Planning Instrument - Land Zoning | Dataset | NSW Planning Portal
	Road	National Roads	ATX (ArcGIS)	Download		Fully		Product catalogue - Geoscience Australia (ga.gov.au)
	Terrain	DTM	image	WMS	2.5D raster 30m accuracy	Fully		
		DTM	image	WMS	2.5D raster 5m accuracy	Partially	Being updated with new data, currently incomplete	

	Type	Name	Format	API	Detail	Coverage	Other	
Germany	Building	Gebäude (buildings)	GML	WFS	LOD0	Fully (Hamburg)		https://opendata-esri-de.opendata.arcgis.com/datasets/1ef87c53adff4c07b9b3010e9d8264fb
		3D-Gebäudemodell LoD2-DE Hamburg	CityGML	Download	LOD2	Fully (Hamburg)	Many separate files	https://metaver.de/trefferanzeige?docuuid=2C1F2EEC-CF9F-4D8B-ACAC-79D8C1334D5E
	Cadastre	Liegenschaftskataster	?			Fully (Hamburg)	Not free	https://serviceportal.hamburg.de/HamburgGateway/Service/Entry/GS
	Landuse	Bodennutzung	image	WFS		Fully (Hamburg)		https://metaver.de/trefferanzeige?docuuid=D059011F-EDBD-4810-9307-BA8D227B5008
	Road	Hauptverkehrsstraßen	ATX (ArcGIS)	WFS	Line features	Only most important roads Hamburg		https://data.europa.eu/data/datasets/af8eca8f-5d29-488f-9bb5-0653ba412759?locale=en
	Terrain	Digital height model Hamburg DGM 1	image	WMS	2.5D raster 30m accuracy	Fully		https://data.europa.eu/data/datasets/93a2b94f-78b1-47d0-a124-7cf1ef467186?locale=en
https://www.gebaeudedaten.ch/								

	Type	Name	Format	API	Detail	Coverage	Other	
Ireland	Building	No public dataset available	?	?	?	?	Only accessible as a product from OSI.ie likely. There is a mention of the dataset on inspire but not findable	https://inspire-geoportal.ec.europa.eu/download_details.html?view=downloadDetails&resourceId=%2FINSPIRE-67c9c760-1be3-11e3-851a-52540004b857_20230315-125002%2Fservices%2F1%2FPullResults%2F81-100%2Fdatasets%2F17&expandedSection=metadata
	Cadastre	Leasehold	gmz/gml	Atom	Digitized from paper maps 1/1000	Ireland		https://inspire-geoportal.ec.europa.eu/download_details.html?view=downloadDetails&resourceId=%2FINSPIRE-67c9c760-1be3-11e3-851a-52540004b857_20230315-125002%2Fservices%2F1%2FPullResults%2F61-80%2Fdatasets%2F14&expandedSection=metadata
		Freehold	gmz/gml	Atom	Digitized from paper maps 1/1000	Ireland		https://inspire-geoportal.ec.europa.eu/download_details.html?view=downloadDetails&resourceId=%2FINSPIRE-67c9c760-1be3-11e3-851a-52540004b857_20230315-125002%2Fservices%2F1%2FPullResults%2F61-80%2Fdatasets%2F15&expandedSection=metadata
	Landuse	No nationwide dataset available	?	?				
	Road	National Roads	shapefile	ESRI (WFS?)	Low, line features only, inaccurate to basemap	Fully		https://data-osi.opendata.arcgis.com/datasets/osi::roads-national-250k-map-of-ireland/explore
	Terrain	DTM	Raster	?	2m interval	?	Paid dataset...	https://osi.ie/products/professional-mapping/products-professional-mapping-height-data/

Bibliography

Arroyo Ohori, K., Ledoux, H., and Peters, R. (2022). *3D modelling of the built environment*. self-published.

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*, 91:219–245.

Davis, S. (2007). *GIS for web developers*. Pragmatic Bookshelf Dallas.

Donkers, S., Ledoux, H., Zhao, J., and Stoter, J. (2016). Automatic conversion of ifc datasets to geometrically and semantically correct citygml lod3 buildings. *Transactions in GIS*, 20(4):547–569.

García-Sánchez, C., Vitalis, S., Paden, I., and Stoter, J. (2021). The impact of level of detail in 3d city models for cfd-based wind flow simulations. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46:67–72.

Gillies, S. (2019). Rasterio documentation. *MapBox: San Francisco, CA, USA*, 23.

Ledoux, H., Arroyo Ohori, K., Peters, R., and Pronk, M. (2022). *Computational modelling of terrains*. Zenodo.

Ledoux, H., Ohori, K. A., Kumar, K., Dukai, B., Labetski, A., and Vitalis, S. (2019). CityJSON: a compact and easy-to-use encoding of the CityGML data model. *Open Geospatial Data, Software and Standards*, 4(1).

OGC (2023). Standard citygml. https://www.ogc.org/standard/citygml/.

Pađen, I., García-Sánchez, C., and Ledoux, H. (2022). Towards automatic reconstruction of 3d city models tailored for urban flow simulations. *Frontiers in Built Environment*, 8:899332.

Purewal, S. (2014). *Learning Web App Development: Build Quickly with Proven JavaScript Techniques*. " O'Reilly Media, Inc.".

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Palatino.