



VGI3D: an Interactive and Low-Cost Solution for 3D Building Modelling from Street-Level VGI Images

Chaoquan Zhang¹ · Hongchao Fan¹ · Gefei Kong²

Accepted: 10 August 2021 / Published online: 23 September 2021
© The Author(s) 2021

Abstract

Applications in smart cities are inseparable from the usage of three-dimensional (3D) building models. However, the cost of generating and constructing 3D building models with semantic information is high both in time and in labour. To solve this problem, we developed a web-based interactive system, VGI3D, with the ambition of becoming a VGI platform to collect 3D building models with semantic information by using the power of crowdsourcing. VGI3D is a platform-independent software program that is composed of a spatially relational database (PostgreSQL/PostGIS) for the storage and management of spatially geometrical data and other software modules, allowing users to import, analyse, reconstruct, visualise, modify and export 3D building models according to the OBJ/CityGML standard. In this paper, we present the VGI3D in detail, focusing on relevant technical implementations, and report the results of limited usability testing aimed at optimising the system and user experience. After limited expert and non-expert participants' testing, we proved the usefulness of VGI3D and its promising value for the 3D modelling community.

Keywords 3D building modelling · Spatial relational database · Python Flask · User interaction · CityGML · VGI images · Low cost

Introduction

As the technology centre in Norway and the first big city to implement and test 5G communication in 2020, the Trondheim municipality is currently gearing up the construction of a smart city. In the coming years, a number of smart city-related applications will be planned. Meanwhile, increasing applications in smart cities necessitate a large number of 3D building models (3DBMs)—just think of solar simulation (Li et al. 2019), virtual tourism (Templin et al. 2020), urban planning (Park et al. 2021), augmented reality (Blut and Blankenbach 2021), path navigation (Liu et al. 2020), disaster management (Haynes et al. 2018), architectural design (Li et al. 2017), etc. Therefore, it is urgent to generate 3D building models with semantic

information. For this purpose, a joint laboratory called Nordig Lab was established as a collaboration between the Norwegian University of Science and Technology (NTNU) and Trondheim municipality. The main focus of Nordig Lab is digitalisation, which includes generating large scale 3D building models with rich semantic information as well.

3D building models are generally defined by five different levels of details (LoDs) according to the CityGML2.0 standard (Gröger and Plümer 2012). LoD0 is a representation of the ground boundary (or footprint) of a building. LoD1 is a cuboid obtained by extruding the LoD0 model. LoD2 consists of a simplified roof shape and multiple semantic classes of a building (e.g. wall, roof). In comparison with LoD2, LoD3 is often considered a more architecturally detailed model in that it contains windows, doors and other rich semantic information. LoD4 is more complicated and completes an LoD3 model by including indoor components. In general, most of the 3D building model-related applications are not just satisfied with the simple LoD1 models and instead want models to have at least a roof shape (i.e., LoD2 or better), particularly in smart cities (Monteiro et al. 2018). They prefer photorealistic 3D building models with windows, doors or balconies in LoD3 or higher, such as the

✉ Chaoquan Zhang
chaoquan.zhang@ntnu.no

¹ Department of Civil and Environmental Engineering,
Norwegian University of Science and Technology,
Trondheim, Norway

² School of Remote Sensing and Information Engineering,
Wuhan University, Wuhan, China

estimation of solar irradiation of buildings (Machete et al. 2018) and urban CO₂ emission simulation and measurement (Eicker et al. 2018). Even some attempts (Tang et al. 2020; Biljecki et al. 2016) have been tried to propose the less generic and more application-driven specification for 3D building models, since standard LoDs specification lacks flexibility and degree of freedom to some extent.

To meet the requirements of the abovementioned applications that require 3D building models with different LoDs, a few cities like New York and Berlin have created and freely released 3D city models based on the CityGML standard over the last decade. These 3D city models not only contain buildings but also include streets, trees, bridges and even terrain. They are defined and presented as the respective real-world objects with respect to their geometrical, topological, semantic and appearance properties (Stadler et al. 2009). Nevertheless, most of these 3D city building models are constructed in LoD1 or LoD2, and large-scale LoD3 models with semantic information are hardly available. Hence, that is the main motivation of this paper to generate 3D building models in LoD3 with semantic information.

3D building models are mainly generated from 3D point clouds and images. 3D point cloud data can be classified into three categories: airborne, terrestrial and mobile laser scanning data. While 3D modelling methods based on point clouds have been widely studied (Sun and Salvaggio 2013; Xiong et al. 2014; Wang et al. 2018), the quality of reconstructed 3D building models varies, owing to differences in point densities, scanning patterns and geometric characteristics (Yang and Dong 2013). In other words, the kind of data used depends on the specific needs of the application. For instance, we would like to create 3D building models in LoD3, and, in this case, airborne point cloud data are obviously inappropriate and cannot provide rich façade information due to the sparse point density of facades (Wu et al. 2017). However, terrestrial point clouds can. In the methods based on terrestrial point clouds, classic approaches usually learn attribute topology and grammar rules from precise descriptions and noisy observations to create high-resolution 3D building models (Becker 2009; Dehbi and Plümer 2011). Even though their reconstructed models are stable and complete, their algorithms are limited by the computational performance and complexity of the scene as well as types of architectural styles.

With the deepening of research and the updating and iteration of algorithms, great success has been achieved in recent years. Dehbi et al. (Dehbi et al. 2017) further optimised their previous work and proposed a statistical method, which is capable of successfully coping with complexity (a varying number of objects), uncertainty and unobservability in real-world problems. This idea also inspired other researchers and was applied to the 3D modelling of indoor environments (Tran et al. 2019). In addition, the great

method presented by Yu et al. (Yu et al. 2017) not only ensured a very high modelling precision but also stamped out the weaknesses of restricted architectural styles. However, to obtain impressive results, the above approaches have to rely on quite expensive point cloud acquisition devices not suitable for mass markets, and modelling procedures are time-consuming and laborious as well.

Another research branch for 3D building modelling is to use images. The classic multiview stereo (Furukawa and Ponce 2009) algorithm often needs to take a set of calibrated images that are surrounding the object. While it emphasises that it is an automatic method and does not require any initialisation, it often suffers from painful secondary editing to fix the dense 3D models they produce, contrary to the original intention of automation. Moreover, images cannot provide rich geometric and topological information about buildings. Hence, completely automatic modelling is known to omit user interaction, and it is generally accepted that such does not produce satisfying results in case of erroneous or partially missing data (Musialski et al. 2013). To address these issues, Wolberg and Zokai (Wolberg and Zokai 2018) designed a photo-centric 3D modelling system—PhotoSketch—that not only benefited automatic camera pose recovery and point cloud generation by using the structure from motion algorithm (SfM) (Ullman 1979) but also introduced user interaction to overcome geometry incompleteness. However, SfM requires many images with overlap and must know the internal parameters of a camera for camera calibration. In their experiment, it took them around 20 min to reconstruct only one building model, which reflected the inefficient computation and great reliance upon the number of images. Getting benefit from advances in deep learning, Liu et al. (Liu et al. 2021) proposed a translational symmetry-aware façade parsing approach for 3D building modelling and achieved extremely photorealistic 3D models. They fused semantic segmentation and instance detection to parse the façade elements into semantic grammars from just one image, and then reconstructed final 3D models by using procedural modelling. Despite gaining highly detailed 3D building models, the system asked users to repeatedly interact and gradually add floors, ledges, roof, windows, doors and balconies, respectively. However, they did not report the whole modelling time and hence we cannot know the efficiency of this approach and cannot compare it to other methods. In addition, to be honest, it is not friendly or easy for non-expert users to use because they may find the interaction process cumbersome and may lose patience.

Furthermore, most of the existing 3D building modelling systems appear predominantly desktop or lab-based, such as Kim and Han's work (Kim and Han 2018), and might need users to do some extra environmental configuration. That will undoubtedly further increase the difficulty of promotion to ordinary users and the mass market

appeal, even though these 3D building modelling systems are outstanding products. On the other hand, many smart city applications now urgently demand 3D building models, but the existing 3D modelling costs are not low in terms of modelling time, labour or data collection. No matter whether from the economic or efficiency point of view, using the existing solutions to simultaneously meet the above points is difficult. Therefore, we are motivated to design a web-based interactive 3D building modelling system with lower costs, faster speed and more intelligent processes.

The idea of volunteer geographic information (VGI) has received widespread attention since its inception. VGI employs tools to create, assemble and disseminate geographic data provided voluntarily by individuals (Sangiambut and Sieber 2016). Nowadays, smart devices with cameras are becoming increasingly powerful and cheaper, particularly for mobile phones. People have been more willing to share their daily life through images than ever. Thus, everyone can become a sensor to contribute image data by using mobile phones, digital cameras and even a GoPro. Although it is hard to evaluate and ensure the quality of VGI images, the convenience of VGI image data acquisition and richness of image data quantity can make up for the shortage of image quality to some extent, which resolves the problem of low-cost data acquisition.

In terms of modelling speed and intelligence, a deep learning technique would be an excellent assistant. In our work, users only need to simply and directly outline the façade of a building upon the image; then, our system will automatically detect façade elements (e.g. windows, doors, balconies) and adjust their bounding boxes. Finally, 2D locations of building elements are going to be transformed into a 3D coordinate system and, meanwhile, show a 3D model to users in real time. Due to the simplicity of interaction, it is easier to promote the system to ordinary users and it will certainly be beneficial to the spread and development of urban 3D modelling.

The novelty of our work is in combining interactive and low-cost 3D building modelling, interactive model updating, real-time 3D model visualisation and integration with a spatially relational database (PostgreSQL/PostGIS) and Python Flask (Flask 2021) web framework. In summary, we intend to verify among both expert and non-expert users that our system is useful and has potential value for the 3D building modelling community.

The rest of this paper is organised as follows: the ‘Methodology’ section presents our methodology, which contains the system architecture and design, algorithm, testing and evaluation and usability testing. Then, we show the implementation with the results of testing in the ‘Software Implementation and Testing’ section. A discussion is presented in the ‘Discussion’ section, and the ‘Conclusion and Future

Work’ section provides the conclusions as well as future work.

Methodology

In this paper, our system has the following functionality: (1) simple user interaction, including roof type and façade orientation selection and façade drawing; (2) geographically matching façade with an edge of the footprint on a 2D map as a reference to get the real ratio/scale; (3) façade elements detection, correction and inference of locations of façade elements; (4) 3D building modelling from 2D locations; (5) interactively editing/updating incorrect parts of reconstructed models; (6) cloud server (AWS EC2) capability for system deployment and PostgreSQL for data storage/retrieval; (7) integration with a Python Flask (Flask 2021) web framework, Three.js (Dirksen 2015) as 3D real-time visualisation and WebGL.

The presented work is based on our two earlier works (Kong and Fan 2020; Fan et al. 2021). The first one (Kong and Fan 2020) proposed a deep learning method to detect façade elements from low-quality street-level images. The second one (Fan et al. 2021) introduced the first version of VGI3D from the perspective of geographic information system (GIS). Now, this paper will concentrate on changes and new features and, meanwhile, further explain our system from the perspective of software engineering.

Designed User Experience

A summary of the designed user experience is presented here to help better understand our system, VGI3D. A typical use case would be as follows: (1) users upload no more than two images that belong to the same building but different façade directions to the building; (2) they select the façade direction for each image and its corresponding footprint edge from the map; (3) they pick a roof type, draw the façade boundary for each image and then click the ‘Save’ button; (4) they repeat this process until no images need to be handled, and then the reconstructed 3D building model with different elements colour would be shown in real time; and (5) additionally, if certain façade element is wrongly reconstructed, they click the ‘Update’ button and start to update the model in an interactive fashion as well by deleting the incorrect element from the 3D model, drawing the element boundary on the façade image, selecting a right element type for it and then click the ‘Save’ button to finish the modification; (6) lastly, the reconstructed 3D building

model can be exported in CityGML/OBJ format by clicking the ‘Download’ button.

System Architecture, Design and Data Flow

An abstract overview of system architecture is shown in Fig. 1, which adopts a classic software design pattern, MVC (model-view-controllers). ‘M’ means model and corresponds to the data layer. All interactions with the database are done here. ‘V’ refers to the display or interaction layer in which we can operate the system through the graphic user interface. ‘C’, the controller, refers to the business layer. Most key modules are placed there and handle the user’s actions. Generally, the controller interprets the mouse and keyboard inputs from the user, then launches a request to the database to get the data, process the data and then update the view.

The VGI3D is currently deployed on a cloud server (AWS EC2) with an Ubuntu version 16.04 operating system. In the data layer, we set up a relational database, PostgreSQL, with PostGIS plugin, for the purpose of spatial geometry search. The data layer mainly stores all the building footprints with geographic locations in Norway (would be used in edge selection in interaction layer), all the images that users uploaded and reconstructed 3D building models. The business layer is the core part of our

system. The concise pipeline is, first, detect the façade elements by using a deep learning model, YOLOv3 (Redmon and Farhadi 2018), such as windows, doors and balconies. Second, we correct the locations of façade, façade elements and roof to unify them under the same coordinate system. Third, we convert 2D elements into 3D and then obtain the final 3D model. A detailed sequence diagram of each step and data flow between steps can be found in our previous work (Fan et al. 2021). User interaction takes place in the interaction layer, which involves roof type and façade orientation selection, façade drawing and geographically matching the façade with an edge of a footprint on the 2D map as a reference to get the real ratio/scale. The last important item would make sure that the reconstructed 3D building model has a real scale, real orientation and real geographic coordinates.

Display layers include HTML5, cascading style sheets (CSS) and Bootstrap for rendering the user graphics, and Three.js is used to provide 3D-rendering support. jQuery is also employed to handle user interaction, mainly listening to response events. An HTTP connection, Ajax and Jinja2 template engine are intermittently needed to communicate with the frontend and backend. Data flow between the frontend and backend is represented in the format of JSON.

Fig. 1 Abstract overview of system architecture

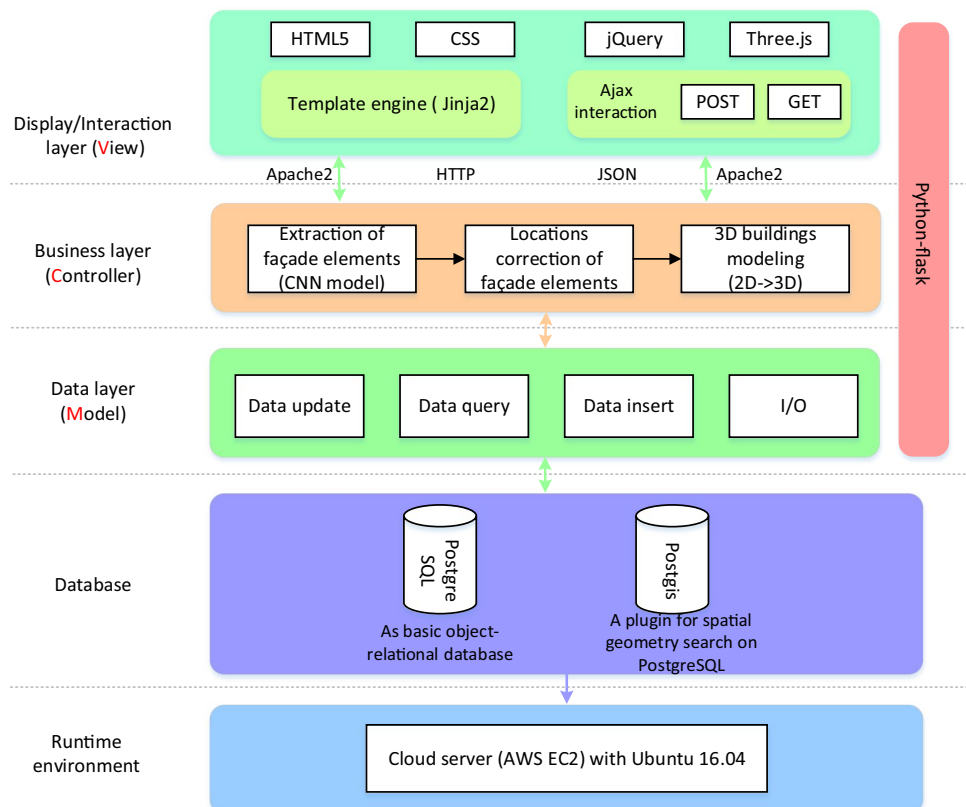


Fig. 2 System activities flow and communication between the database and Flask web service

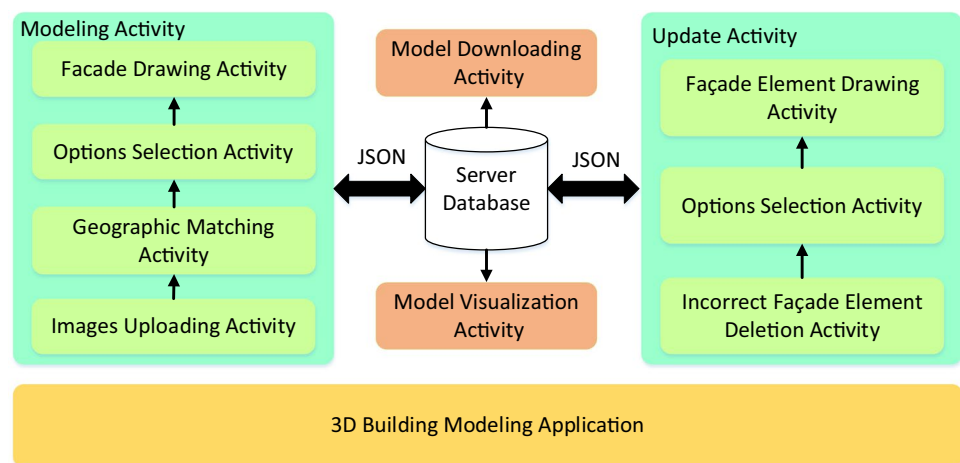


Figure 2 shows our core system design with the 3D building modelling application at the bottom. Activity flow follows the direction indicated, but the order within ‘modelling activity’ can be interchanged (i.e. no fixed order). The JSON data flow helps with communication between the database and web service and is capable of accessing via ‘modelling activity’ and ‘update activity’ as well.

Algorithms

Since the core concepts of our 3D modelling method are low cost and speed, one of the key algorithms is to automatically and accurately detect façade elements in a short time. As we know, building facades in street-level images generally have complex scenes, such as distortion, blur or bad illumination. The existing approaches demand users to manually draw façade elements, which is labour-intensive and time-consuming. Hence, we adopt a convolutional neural network (CNN) model, YOLOv3, to help us automatically extract façade elements. We follow the steps suggested by Kong and Fan (Kong and Fan 2020) and train YOLOv3 on the FacadeWHU dataset (Kong and Fan 2020), which includes 900 street-level images (50 images in Trondheim, Norway, and 850 images in Paris, France). Then, we can obtain the location of every façade element on the image coordinate system. Since street-level images usually have more or less perspective distortion, we need to correct the extracted façade elements and then project them to a 3D space. Besides, for visualisation and modelling completeness purposes, we also need to construct the building’s roof according to roof type that was selected by the user before. The top edge of the façade is regarded as the bottom of the roof. The height of the roof can be estimated according to the ratio of façade length to width. Algorithm details can be found in the ‘Methodology’ section of our earlier work (Fan et al. 2021).

In some cases, however, this CNN model may sacrifice accuracy or completeness while ensuring speed. Some façade elements might be incorrectly detected because of their complicated scenes or low-quality images. Therefore, interactively updating 3D models is needed and necessary. For this purpose, VGI3D allows users to firstly select and delete the incorrectly reconstructed façade element based on the original model. Next, they can simply outline the wrongly reconstructed façade element upon an image and select a corresponding element type for it. Then, an updating algorithm will automatically merge the new façade element into the original 3D model and adjust the locations of elements to make the overall model look coordinated and harmonious. After that, the new modified model will be shown to users in real time. This algorithm worked well and met the requirements of fast speed, ease of use and low cost.

Last but not least, to apply 3D building models into various map-based applications, it is essential to give 3D building models real geographical coordinates instead of relative coordinates. We asked users to select a footprint of the building that they want to construct on the map (here, it was OpenStreetMap) before outlining the façade, then further select the edge from the footprint, which corresponds with the façade direction being drawn. In other words, users should pick up an edge for each façade within the image. Our modelling algorithm will be able to calculate the ratio (façade height/façade width) of real coordinates and assign real geographical coordinates for each vertex of the geometric 3D model.

Testing and Evaluation

In addition to usual unit tests during development, functional testing was also carried out to make sure the system worked as planned—for example, by raising technical issues. The

functional testing is mainly conducted around the following aspects:

- (1) Image-uploading activity, including uploading more than two images and continuously uploading images.
- (2) Geographic matching activity, including map layout/location, highlighting selected footprint and edge with different colours.
- (3) Options activity, including operating under different conditions, e.g., different combinations of façade direction and roof type.
- (4) Façade/façade element drawing activity, including drawing operation and deleting wrong drawing operations.
- (5) Incorrect façade element deletion activity, including selecting and highlighting incorrect elements and deleting incorrectly reconstructed elements.
- (6) Visualising and exporting the 3D building model.

The system was evaluated by conducting a user study with 30 non-expert/expert participants. The Participants were (a) shown a tutorial video and a slide presentation about how to operate the system, (b) were given some time to try the system by themselves, and finally (c) were asked to fill in a user feedback form. A copy of the user feedback form is available in Appendix A. Most questions of this form are for participants to rank a specific aspect of the system from one (least) to five (most) points. The raw data of their answers can be found in Appendix B. Please note that Nos. 1–15 are expert participants and Nos. 16–30 are non-expert participants. The grouping criteria is roughly based on: the research fields are strongly or relatively related to the 3D model as the expert group; the research fields are weakly or slightly related to the 3D model as the non-expert group.

Software Implementation and Testing

The VGI3D has been implemented and can be accessed at: <https://18.210.26.42:5002/facade/>. We also invited expert and non-expert users to test the software. In this section, software implementation will be demonstrated.

Implementation

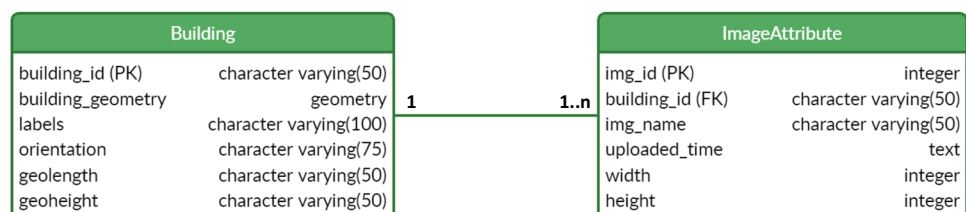
A driving principle behind our design is to enable the system's interaction as simple as possible, which would be friendly to non-expert users and would be more beneficially promoted to the mass market. For this purpose, we borrowed the drawing idea from the Mapbox (Mapbox 2021) SDK (software development kit) to achieve the façade polygons on images. A building footprint database with PostGIS was created in advance to supply geographic information of footprints and buildings' height. Then, Leaflet (Leaflet 2021) was utilized to select edges of the footprints from an embedded map and wrapped and transmitted them to the backend for calculation via GeoJSON. Relying on these geographical information, we could calculate the real ratio of building facades, which would further ensure reconstructed building models have real geographic coordinates.

Another driving principle behind our design is modelling buildings anywhere. This tries to ensure the modelling system is viable in a general environment and, hence, users will not be constrained to a limited number of facade styles. To realise this goal, we adopted a CNN model, YOLOv3, for helping to automatically detect façade elements (i.e. window, door, balcony) within one second. The utilization of CNN not only satisfied the modelling anywhere but also saved considerable time for the entire workflow because the step of façade elements extraction is the most time-consuming according to previous work (Nishida et al. 2016).

After the interaction with all images, the created data are transferred to the backend in JSON format to reconstruct 3D building models. The content of data includes edges' geographic coordinates, locations of façade boundaries, locations of all the façade elements, building's height, uploaded images, roof type and façade orientations. In addition, all these data will be stored in two separate tables of database (as shown in Fig. 3). One is used for storing building's geometry and the other is for image attributes. They can communicate with each other via a foreign key, *building_id*. The reconstruction part of models has been explained in detail in our previous work (Fan et al. 2021), so we are not planning to describe the reconstruction part here once again.

Although our objective is capable of automatically and accurately reconstructing building models with arbitrary

Fig. 3 Table structure of the database



facade styles, the actual situation always deviates from the ideal situation. For example, there are tons of styles of doors and windows in the world and our CNN model cannot correctly detect all of them due to an inadequate training dataset. Sometimes, incorrect or missing detection can also be caused by low-quality images, such as distortion, blur, illumination or reflection. Therefore, update activity was implemented to interactively correct the wrong façade elements. In our approach, users can (a) select and highlight the element that they want to modify; (b) remove it by clicking the ‘Delete’ button, which is located at the upper right corner of the 3D viewer; (c) draw the corresponding element’s boundary that we deleted before; (d) select the

element type and façade orientation as extra but essential attributes; (e) click the ‘Update’ button to wrap the data into a JSON file and transmit them to the backend; (f) the system would simultaneously execute the update function and update the database; (g) lastly, the modified 3D model would be returned to the frontend. Figures 4 and 5 illustrate the whole modelling process. The benefit of this approach was to make the modelling process more flexible and convenient and make the results more accurate to some degree.

Once modelling has been completed, exporting 3D building models is possible and available by clicking the ‘Download’ button. We currently support two types of 3D formats, OBJ and CityGML.

Fig. 4 (a) Main modelling activity; (b) geographic matching activity; and (c) reconstructed 3D building model visualisation

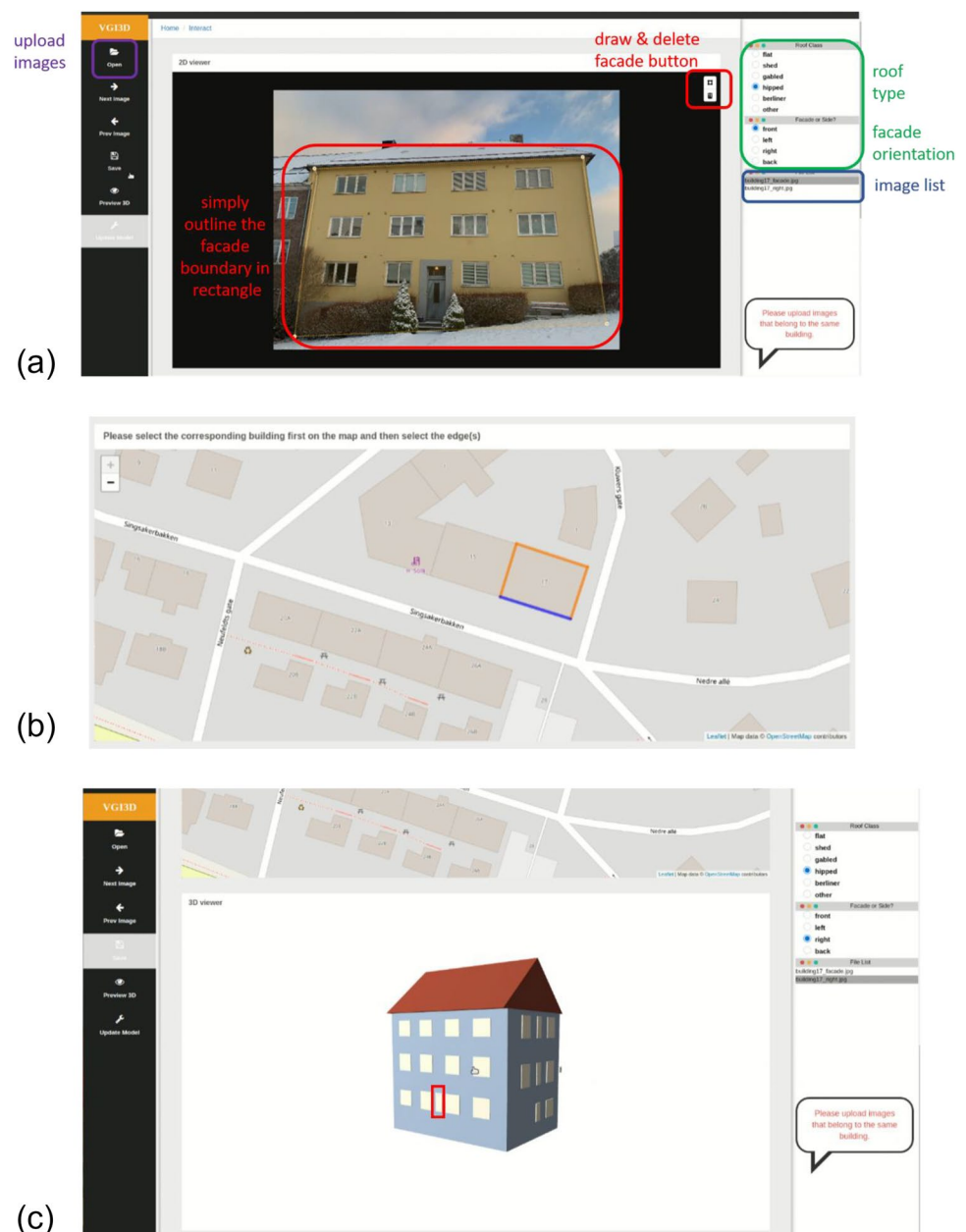
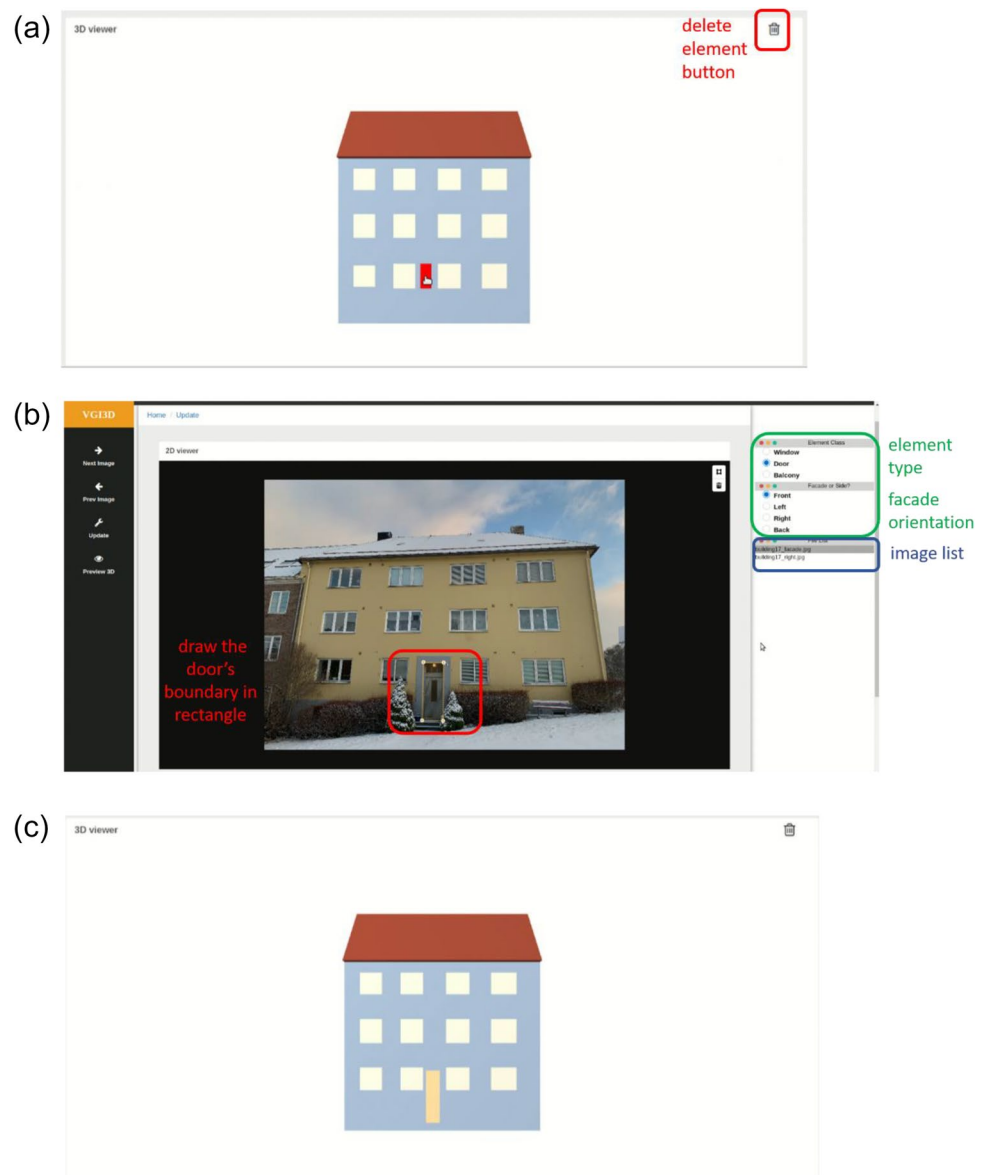


Fig. 5 (a) Incorrect façade element deletion activity; (b) model updating activity; and (c) new 3D building model after updating



Software Functional Testing

The system was opened on a laptop in browser mode and the main modelling page appeared. First, image uploading activity was activated, in which preprepared images that belong to the same building were uploaded by clicking the fold icon of the left sidebar, as shown in a purple rectangle of Fig. 4a. Then, one image in the image sequence was displayed in the 2D viewer of the workspace. Second, we searched and chose the footprint of this building (i.e. highlighted in orange) from the map, then further selected the corresponding edge (i.e. highlighted in blue) of the façade that the participant was handling, from the map to complete the geographic matching activity (as shown in Fig. 4b). Third, we respectively selected the roof type and façade orientation for the current processing image from the right sidebar to finish the options

selection activity (see green rectangle in Fig. 4a). Fourth, we began to draw the façade boundary on the image, which generally is a rectangle, to complete the façade drawing activity (see red rectangle in Fig. 4a), then clicked the 'Save' button to transfer the data to the backend and save them into the database. Fifth, we switched to the next image by clicking the 'Next Image' button, then repeated the second to fourth steps until all images were done. Once the functional testing of modelling activity had been finished, the reconstructed 3D building model was visualised in a 3D viewer workspace as expected (see Fig. 4c). Additionally, if participants did not select any item of a certain kind, such as roof type, edges or façade orientation, a dialogue box would pop up to remind participants not to omit certain items. Or, another case was that, if participants would like to reupload images and discard all the operations that they had done previously, the

system would also pop up a dialogue box to ask whether they wanted to discard them or not. These eventualities were all considered during the development stage and were worked as planned.

Sometimes, the incorrect façade elements (e.g. the door outlined in red in Fig. 4c) are inevitable after reconstruction due to the reasons we mentioned above. Such will lead us to test update activity by clicking the ‘Update Model’ button in the left sidebar. Meanwhile, we will advance to a new page, ‘update page’. In the 3D viewer workspace, a deletion icon was available, which was used to delete the selected wrong façade element with highlighted colour (see Fig. 5a). That was the first step under update activity, i.e., incorrect façade element deletion activity in Fig. 2. Second, we found the right image that we would like to draw upon via clicking the ‘Next Image’ or ‘Prev Image’ button, then manually drew the boundary of the façade element on the image whose location corresponded to the element we deleted just now. In this way, we completed the testing of the façade element drawing activity (see red rectangle in Fig. 5b). Third, we selected the appropriate element class and façade orientation from the right sidebar (see green rectangle in Fig. 5b). Fourth, we clicked the ‘Update’ button to transmit the data to the backend and executed the model updating functionality. Meanwhile, we updated the database (i.e., column *building_geometry* and *labels* of table *Building*) accordingly and the updated result would be displayed in the 3D viewer in real time (see Fig. 5c). We repeated this process until no other incorrect façade elements needed to be updated. Please note that only one façade element can be updated at a time. Finally, exporting 3D models was supported according to the OBJ/CityGML standard.

Until now, we have managed to test all functionalities and everything worked well, with the only possible challenge being the quality of images. Images of a low quality, particularly for façade parts, can affect the completeness or stability of façade elements extraction, but it is typically

not severe enough to kill performance. The system most likely performed well thanks to the fact that modelling has a powerful CNN detection model as a reliable cornerstone, simple but considerate interaction logic and novel interactive updates. The system was also tested extensively in practice, which tremendously reduced the potential problems during actual use. Furthermore, since VGI3D was developed based on HTML5, it is possible to open it in the browser on any device. Hence, we tested the system on a mobile phone, but, unfortunately, the user experience was not good at all because the screen of a mobile phone was too small and the layout of widgets had changed a lot and made the system look very ugly.

Software Evaluation

To evaluate the usability of VGI3D, we conducted a small scale of usability testing at the Norwegian University of Science and Technology, Norway, and Wuhan University, China. 30 expert/non-expert participants (aged 25–43 years old, including 7 women and 23 men) were invited to experience and test the VGI3D. Six of them are expert participants and the others are non-expert participants. Their research fields include 3D city modelling, 3D visualization, photogrammetry, urban planning, BIM for asset assessment/management, spatial analysis, computer science, GPS, railway design and facility management. Before the formal testing, they would be shown a tutorial video and a slide presentation about how to operate the system and be given some time to get familiar with the system. After that, our testing kicked off. Each participant was randomly assigned a preprepared folder where storing building images, and the participant could use them for testing. Once the testing was complete, they were asked to fill in a user feedback form. This form included questions about, for instance, their demographic information, previous experience (‘How experienced are you with 3D modelling, e.g., Sketchup?’), satisfaction (‘The

Table 1 Summarized suggestions/comments from participants during the usability testing

‘The user interface is clear and concise, and easy to understand overall.’
‘This button is slightly small; making it bigger would be better.’
‘Overall, the system is good and promising and is beneficial to my research.’
‘From the BIM point of view, the reconstructed 3D model is not as accurate and detailed as BIM model.’
‘The tips are not obvious. Had better move them to a more obvious place.’
‘Downloading model spent a little longer time in my case.’
‘The modelling time is beyond my expectation.’
‘Deleting incorrect façade elements sometimes would face bugs.’
‘The reconstructed model is cool and 3D viewing is smooth.’
‘Include building height to get proper element ratio.’
‘Some reconstructed windows are uneven in size.’
‘Hope to improve the system so that it can reconstruct buildings with arbitrary footprints.’
‘Cannot find a specific location name by searching on the map. Hope to fix this problem.’

interaction operation is easy to understand’), understanding of the system, any subjective comments/suggestions, etc. Their comments/suggestions have been summarized and listed in Table 1, as shown below.

After the usability testing, we fortunately received some positive feedbacks from participants: 93.3% (28/30) of the participants reported that the user interface was ‘clear’ and ‘easy’ to understand. In terms of 3D modelling, two thirds of participants (20/30; 66.7%) could manage to reconstruct 3D models. Most of the participants who could not successfully reconstruct the 3D models either were non-experts or were inexperienced in 3D modelling. Participants largely thought our system could reconstruct buildings at a rapid speed, and the plausibility of modelling results as well as 3D viewing capability exceeded their expectations. Most importantly, some participants said VGI3D was very beneficial to their own researches. Meanwhile, we also received negative feedbacks from participants. For instance, one of them reported

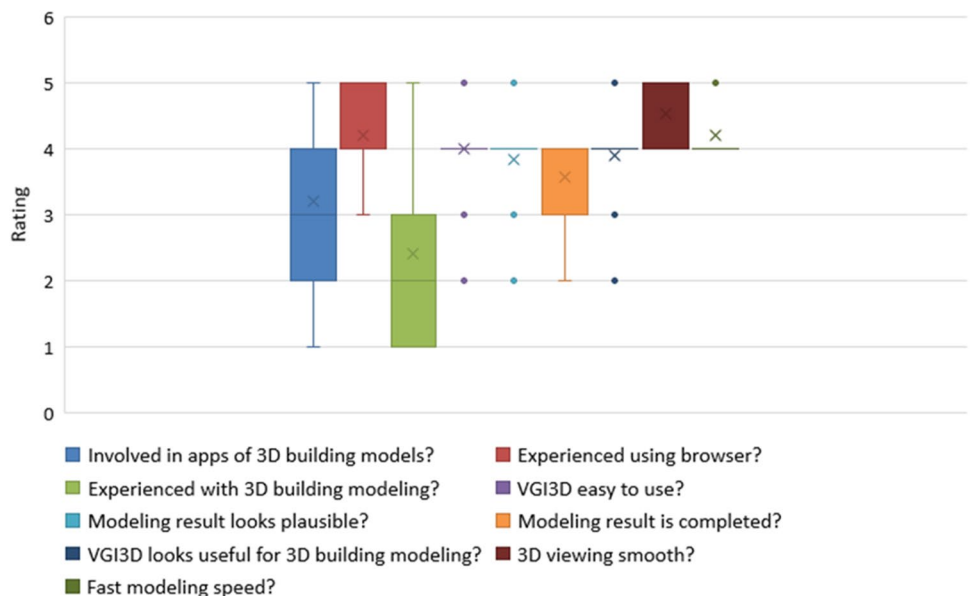
that the tips were not obvious and were easy to be ignored. Three of them found the same bug when trying to delete the wrong façade elements (i.e. failed to delete). Since BIM participants have been used to utilizing high-detailed 3D models for research, they thought our models were not very photorealistic.

Apart from the qualitative comments, we also performed the quantitative evaluation and analysis based on user-feedback-form data from the statistical point of view. The raw data in Appendix B has been statistically summarised in Table 2. Outliers are identified as single points in box plots as shown in Fig. 6. Besides observing the spread and centrality of data, we also proposed related questions by judging how some responses of a participant were associated with others. Since our sample size of participants was not too big, its use would lead to a sparse scatter plot. Sometimes it could be nonlinear in appearance but usually included bound data internally (see Fig. 7). Therefore, to identify associations between participants’

Table 2 Statistics about raw data collected from participants. Mean(μ), standard deviation(σ), lower quartile(Q1), middle quartile(Q2), upper quartile(Q3), interquartile range (IQR). 3DBMs: 3D Building Models; 3DBMing: 3D Building Modelling; 3DMing: 3D Modelling

	Involved in 3DBMs apps?	Experienced using browser?	Experienced with 3DMing?	VGI3D easy to use?	Modeling result looks plausible?	Modeling result is completed?	VGI3D looks useful for 3BMing?	3D viewing smooth?	Fast modeling speed?
μ	3.20	4.20	2.40	4.00	3.83	3.57	3.90	4.53	4.20
σ	1.30	0.60	1.33	0.52	0.58	0.56	0.60	0.50	0.40
Q1	2.00	4.00	1.00	4.00	4.00	3.00	4.00	4.00	4.00
Q2	3.00	4.00	2.00	4.00	4.00	4.00	4.00	5.00	4.00
Q3	4.00	5.00	3.00	4.00	4.00	4.00	4.00	5.00	4.00
IQR	2.0	1.0	2.0	0	0	1.0	0	1.0	0
Skew-ness	-0.19	-0.11	0.51	-1.45	-0.98	-0.84	-0.90	-0.13	1.50

Fig. 6 Box plot of data for illustrating skewness



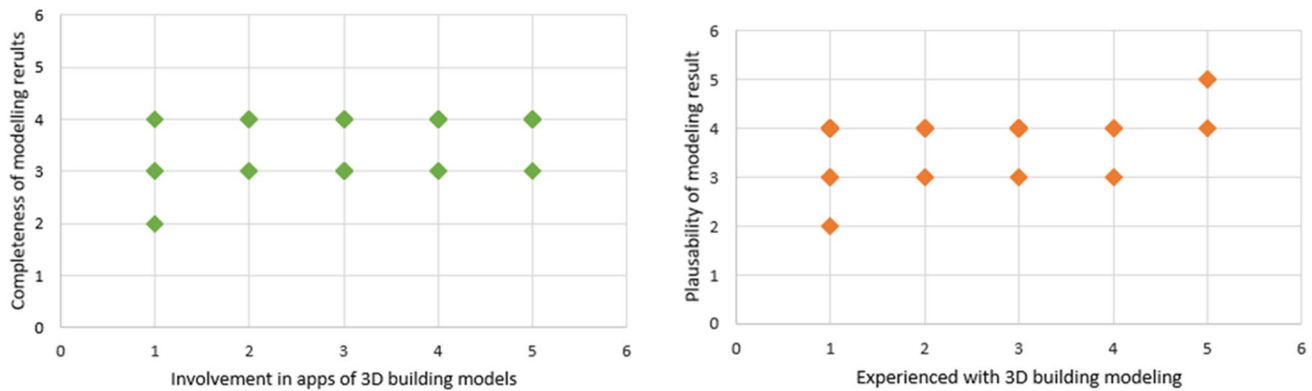


Fig. 7 Examples of sparse scatter plots with nonlinearity in appearance but including bound data points internally

responses data, we computed Spearman's rank correlation coefficient, which is capable of handling linear, nonlinear and skewed relationships. In our case, we had to calculate the Spearman's coefficient with full covariance rather than an approximate formula because of the presence of bound data and, thus, duplicate ranks. Table 4 compares the correlation coefficients between all possible question pairs.

Discussion

Data Analysis

The answers to questions in the user feedback form usually have skewness. We can find their skewed degrees (i.e. magnitude) and nature (i.e. positive or negative skewness) from Table 2. The first column of Fig. 6 shows a wide range of knowing about applications of 3D building models and with a negative skewness of the data. This is what we expected because our participants are from different research fields and the number of participants is limited. We cannot guarantee that every participant is quite involved in applications related to 3D building modelling.

Almost all participants were able to use the browser skillfully with a mean rating of 4.20, slightly negative skewness and no outliers. Most of the participants (24/30; 80%) were not very familiar with 3D building modelling, as seen by an apparent positive skewness and mean rating of 2.40, which is interesting when we compare it with the mean value of 3.20 recorded for the first column (involvement in apps of 3D building models), which has opposite skewness. This might indicate the participants currently do not often use 3D building modelling tools, not to mention the interactive 3D modelling system. To rule out the impact of non-expert participants (Nos. 16–30) on the results, we further calculated the mean value of expert group (Nos. 1–15) on question 'how experienced with 3D modelling?' and obtained the

mean rating of 3.47. This degree is between 'moderately' and 'very' and further verified our guess is correct. This could be interpreted as the novelty of our interactive VGI3D to the 3D building modelling community. almost all participants (28/30; 93.3%; rank over 4) thought our system was easy to use, with a mean rating of 4.00 and negative skewness. Furthermore, most participants (24/30; 80%) reported the modelling results looked plausible by a mean rating of 3.83 and a lower quartile (Q1) value of 4.0, reflecting the fact that most participants highly appraised to VGI3D. Both the modelling result completeness and system usefulness to the 3D building modelling field were considered positive, with mean ratings of 3.57 and 3.90, respectively.

In terms of system usefulness to 3D building modelling, we not only obtained the highest rating of 5 three times but also got the low rating of 2 once. As for the low rating, the participant had no 3D modelling experience and his research interest (railway design) was quite different from the 3D building modelling. As a result, he did not understand the system well; hence, he ended up failing to reconstruct the building model. The ability of understanding varies from person to person and we cannot expect everyone to be able to use our system smoothly; still, this special case encourages us to better optimise the system in the future.

In addition, to further analyse the assessment of different types of participants towards the system, we grouped them by their research fields and then compared the ratings between different groups on each question. Due to space limitations, only 'mean (μ)' is displayed here as shown in Table 3. The remaining statistical indicators (standard deviation and skewness) can be found in Appendix C. From Table 3, we can discover that all the groups gave the highly positive assessment (rank over 4) in terms of smooth 3D viewing capability and fast modelling speed. Together with the plausibility of modelling results and VGI3D usefulness, there is an interesting finding over them. The research fields (BIM and facility management) that need high-detailed 3D

Table 3 Mean value (μ) comparisons between different types of practitioners regarding each question in user feedback form

	Involved in 3BMs apps?	Experienced using browser?	Experienced with 3DMing?	VGI3D easy to use?	Modeling result looks plausible?	Modeling result is completed?	VGI3D looks useful for 3BMing?	3D viewing smooth?	Fast modeling speed?
3D city modeling	4.75	4.25	4.75	4.50	4.50	3.75	4.50	5.00	5.00
3D visualization	4.50	4.50	3.00	4.50	4.00	4.00	4.50	5.00	4.00
Photo-grammetry	4.30	3.67	3.33	4.00	4.00	4.00	4.00	4.67	4.33
Urban planning	4.00	3.50	3.00	4.00	4.00	3.50	4.00	4.50	4.00
BIM for asset assessment / management	3.75	4.50	2.75	4.00	3.50	3.25	3.75	4.50	4.25
Spatial analysis	2.67	4.33	1.00	4.00	4.00	3.67	4.00	4.33	4.00
Computer science	1.80	4.40	1.40	3.80	3.80	3.60	3.80	4.40	4.00
GPS	1.50	4.00	1.00	4.00	4.00	3.50	4.00	4.50	4.00
Railway design	1.50	4.50	1.00	3.00	2.50	2.50	2.50	4.00	4.00
Facility management	3.00	4.00	2.00	4.00	3.67	3.67	3.67	4.33	4.00

models tend to give the moderate evaluation, which might be because these two groups have got used to utilizing the realistic models and think our reconstructed models are not as realistic as the actual buildings, sometimes even worse. Therefore, they do not think our models can be applied to their researches. Additionally, strictly speaking, only two groups (3D city modelling and photogrammetry) have the 3D modelling experience with the mean rating of 4.75 and 3.33, respectively. However, other groups except railway design group still gave us the positive assessment on ease of use, plausibility, completeness, VGI3D usefulness, etc., which indeed reflect that overall our system is great.

Correlation Analysis

Table 4 presents the correlation between all questions based on Spearman's rank correlation coefficient. All correlations were positive except those that were associated with experience using browsers. In our testing, all participants have rich experience in using browsers. These negative correlations could be interpreted as whether or not having experience with using browsers has no direct relationship with other modelling-related responses. That makes sense if we connect with the practice. Nowadays, well-educated people generally have rich experience in using browsers, but they do not necessarily understand 3D building modelling. From Figs. 6 and

7, we can discern that Spearman's coefficient is appropriate in possible nonlinear data and skewed data.

First, we observed whether or not involved in 3D building model apps or experience with browsers or 3D building modelling associated with opinions regarding whether VGI3D was easy to use/understand, modelling results looked plausible and completed, and if VGI3D was indeed useful for the 3D building modelling community. Our findings in Table 4 demonstrate weak positive correlations between involvement in 3D building model apps and the completeness of modelling results, but with nearly 100% confidence and moderate positive correlations with ease of use, modelling results plausibility as well as VGI3D usefulness. However, this weak correlation does not mean that it lacks support from participants as illustrated in Fig. 7(left). Figure 7 (left) seems to show an upward trend; however, we still require more statistical data to verify our guess because the current sample size is not big enough. Another interesting finding was that experience with 3D building modelling showed weak positive correlations with the plausibility and completeness of modelling results as well as smooth 3D viewing capability. This finding was contrary to our intuition. After carefully analysing the raw data, the reason for this may be that non-expert participants have little experience in 3D modelling, which reduced the overall correlations among responses. However, the relationship between experience with 3D building modelling and the plausibility

Table 4 Correlation table showing Spearman's rho and significance values for parameters

	Involved in 3BMs apps?	Experienced using browser?	Experienced with 3DMing?	VGI3D easy to use?	Modeling result looks plausible?	Modeling result is completed?	VGI3D looks useful for 3BMing?	3D viewing smooth?	Fast modeling speed?
Involved in 3BMs apps?	1								
Experienced using browser?	-0.22 (p=0.25)	1							
Experienced with 3DMing?	0.76 (p=0.00)	-0.10 (p=0.60)	1						
VGI3D easy to use?	0.54 (p=0.00)	-0.17 (p=0.36)	0.50 (p=0.00)	1					
Modeling result looks plausible?	0.51 (p=0.00)	-0.28 (p=0.13)	0.33 (p=0.08)	0.68 (p=0.00)	1				
Modeling result is completed?	0.36 (p=0.05)	-0.01 (p=0.95)	0.14 (p=0.45)	0.45 (p=0.01)	0.64 (p=0.00)	1			
VGI3D looks useful for 3BMing?	0.47 (p=0.00)	-0.08 (p=0.66)	0.50 (p=0.00)	0.82 (p=0.00)	0.63 (p=0.00)	0.34 (p=0.06)	1		
3D viewing smooth?	0.50 (p=0.00)	0.05 (p=0.78)	0.31 (p=0.09)	0.40 (p=0.03)	0.56 (p=0.00)	0.60 (p=0.00)	0.27 (p=0.14)	1	
Fast modeling speed?	0.44 (p=0.01)	0.10 (p=0.60)	0.72 (p=0.00)	0.37 (p=0.04)	0.28 (p=0.13)	0.08 (p=0.68)	0.38 (p=0.04)	0.30 (p=0.11)	1

of modelling results in Fig. 7 (right) shows a completely different trend from the correlation. Higher ratings suggested that, regardless of whether participants have rich 3D modelling experience or not, they did feel that the modelling results looked plausible.

Next, we would like to know whether or not VGI3D usefulness for the 3D building modelling community was associated with any item of easy usage/understanding of VGI3D, the plausibility of modelling results or the completeness of modelling results. Table 4 clearly demonstrates strong correlations between usefulness for the 3D building modelling community and ease of use/understanding of VGI3D and the plausibility of modelling results with nearly 100% confidence. However, no strong correlation could be discovered between the completeness of modelling results and usefulness.

Our final observation concerned the ease of use/understanding of VGI3D and the plausibility of modelling results. We found a distinct positive correlation with value of 0.68 between them, which appeared to be consistent with our intuition. Until now, all observations and correlations have been based on limited sample data. We can only conjecture the meanings behind them, but they still give us much help and clues to identify which aspects of the system will affect users' perspectives and will be beneficial to us for optimising the system in the future.






Sensitivity Analysis

To better display the low sensitivity of our system, we selected five different buildings from simple to complex in terms of façade complexity (as illustrated in Table 5) for sensitivity analysis. Building A had the simplest façade structure with 7 windows and it was reconstructed by 2 input images in 48 s. Due to its simple façades, our system correctly detected all windows and hence no need to manually and interactively update. Building A took another 19 s for export as CityGML model where mainly did the rotation, translation and scaling according to the real building size and its real orientation in reality, and did the coordinate transformation (i.e. convert from local coordinate system to world coordinate system).

For building B, its facade became a little more complex with 16 windows and 1 door, and was bigger in building size compared to building A. The door of building B was not correctly detected and was wrongly recognized as a window, probably because of the similar features between them in this case. Thus, we had to manually update the door and it took us another around 15 s. Since building B was reconstructed through one image, it would consume less time when exporting.

Building C had more windows than building B and was reconstructed through 2 images. The testing results showed that semi-auto time was similar to building A's and a similar

Table 5 Summary of buildings with various sizes and façade complexities for sensitive analysis. The fifth column shows the interactive modelling time (semi-auto), time of interactively updating incorrect façade elements (update), exporting 3D model time (export) and updated façade elements in (-). Blue is for door, cyan is for window and green is for balcony

No.	Num of imgs	Building size (L×W×H) m	Façade complexity	Modelling time (s) (semi-auto + update + export)	Output 3D model size (KB)	3D viewing
A	2	11×4.7×8.7	7 windows	48 + 0 + 19	87.9	
B	1	21.4×8.3×12.4	16 windows 1 door	24 + 15 + 11 (1 door)	112.7	
C	2	16×12.1×14.6	21 windows 1 door	49 + 32 + 21 (1 door + 1 window)	150.9	
D	2	11.7×10.4×20.7	17 windows 1 door 5 balconies	47 + 59 + 20 (1 door + 1 balcony + 2 windows)	162.5	
E	1	30.3×15.1×20.6	28 windows 4 doors 6 balconies	25 + 13 + 11 (1 balcony)	238.5	

situation appeared in the building D case. In fact, more than half the time was consumed by user's intersection; however, façade elements detection usually was done within a similar time regardless of the façade complexity. In other words, as long as the proficiency of user interaction can be improved, the overall time can be further reduced. Since two elements were wrongly detected, user spent another 32 s on updating. Meanwhile, exporting time (21 s) similar to building A's was recorded.

Balconies showed up in both building D and E and made the façades become more complicated. However, the semi-auto time and exporting time seemed not to change a lot and still keep similar/stable compared to previous cases. Additionally, updating time is affected by input images' quality and proficiency of user interaction. If doors or windows are obscured by cars, that undoubtedly cannot detect them correctly and thus would increase the time and workload of manual updates, just like two missed windows in building D. The proficiency of user interaction depends on users and hence it is hard to measure. In addition, the only thing that changed was the size of the output 3D model. Their sizes increased with the increment of façade complexity as we expected, but even the most complex building E, its output size was still less than 240 KB.

Finally, in terms of 3D viewing capability, our system gave users a good 3D visualization experience without any laggings. We can also verify this point according to

participants' feedback on Q11 (Is the 3D viewing capability of the model smooth?). All the participants select the 'Tend to agree' or 'Strongly agree'.

In summary, our system has low sensitivity regarding various sizes and complexities of buildings on the functionalities of the system including the modelling time, 3D viewing capability and the output size of 3D models.

Limitations

First of all, the number of participants was relatively small, and their comments and responses may not reflect the views of all those working in the domain of 3D building modelling.

Second, given the limited experience of the design team, we could not carry out sufficient user requirements analysis at the beginning of the system design. For this reason, we could only quickly develop the system and let users give us feedback through usability testing.

Third, the current VGI3D system cannot provide an overview of building models after one user managed to generate a few models. Additionally, current 3D models are visualised in '3D viewer' under the relative coordinate space. If they can be directly visualised on the map with real geographic coordinates, that would be greater.

Fourth, since the building heights are not available except in Trondheim, Norway, the current 3D models after exporting as CityGML are given a fixed height, 12 m. In the future, we plan

to add an option and allow users to enter the number of floors and then the system can roughly estimate the height of buildings. Furthermore, except for Norway, Sweden and Italy, footprints from other countries are not available for the time being.

Fifth, now VGI3D can only reconstruct buildings whose footprints are in rectangle-like shape. We will improve the modelling algorithm and extend it to arbitrary footprints. However, it might lead to a new challenge, mainly on roofs. Since our current roofs are automatically reconstructed according to user's roof type selection, if building's roof is fancy and consists of several different parts, the original method for automatic roof reconstruction will not be appropriate anymore. To solve this challenge, we are considering whether users could sketch a roof as the customized roof type in the future version.

Last but not least, the current performance in façade elements detection still has room for improvement. For example, one participant reported '...almost 25% of windows and one big door were not be detected in my test case' and '...if [it] let me manually draw too many windows at one time, I would lose patience'. Although interactively updating models is one of our highlights, ideally, we want to make users reduce interactive manipulation as much as they can. One possible solution to this problem is to extend the training dataset with more different kinds of facade images. Another solution is to collect the labelled data when users manually draw façade elements, then add them into the training dataset and retrain the deep learning model again to attain a better detection accuracy.

Conclusion and Future Work

This paper presented the VGI3D system, which is a web-based and interactive solution for 3D building modelling, from the perspective of software engineering. VGI3D consists of a spatially relational database (PostgreSQL/PostGIS) for the storage and management of spatially geometrical data and other software modules, allowing us to upload images as input, analyse and reconstruct, visualise, modify and export virtual 3D building models according to the OBJ/CityGML standard. Functional and usability testing under limited participants were also conducted, which was intended to assess the potential usefulness of our work for the 3D building modelling community and to better optimise the system and user experience at the same time. We interpreted our findings as that, regardless of whether or not participants have rich 3D modelling experience, they did think VGI3D is useful and has promising value for the 3D building modelling community. However, our system still has much room for improvement and optimisation. On one hand, it is necessary to further strengthen our CNN model and enable it to detect façade elements as accurately as possible so as to reduce the user interaction costs for updating 3D models. On the other hand, from suggestions/comments,

it was apparent that reconstructing complex buildings would be helpful for applications that place high demands on models.

Overall, our VGI3D needs less input and user interaction and is easy to manipulate. It is suitable for quick modelling but still with relatively high details (i.e. LoD3). We hope our work could provide a new idea for the 3D building modelling domain and let more volunteers join this community to contribute to wider applications in smart cities.

Furthermore, VGI3D is currently running on an Amazon cloud server (EC2). At the end of this year or the beginning of 2022, it will be moved to the website of Trondheim municipality and released as a citizen participation project in Trondheim. The announcement regarding the host change will be made publicly available when the new website is ready to be accessed.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s41651-021-00086-7>.

Author Contribution Hongchao Fan formed the research idea and revised the manuscript. Chaoquan Zhang designed and conducted the experience, analysed the results and drafted the manuscript. Gefei Kong co-conducted the usability testing and analysed the results. All authors read and approved the final manuscript.

Funding Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital). This research was supported by research fund from the Norwegian Public Roads Administration under project name λRoad.

Data Availability The FacadeWHU dataset that support this study are available at: <https://drive.google.com/drive/folders/1BeKXMuNAaRcwP6lSeiqFOSfscbP9ZKm7?usp=sharing>

Code Availability The code is currently stored on a local area network (LAN) of university and have not submitted to like Github. If this paper is accepted, we will release the code there immediately.

Declarations

Ethics Approval Not applicable.

Consent to Participate Not applicable.

Consent for Publication All authors approve that the Journal of Computational Urban Science can publish our article.

Conflict of Interest The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Becker S (2009) Generation and application of rules for quality dependent façade reconstruction. *ISPRS J Photogramm Remote Sens* 64(6):640–653
- Biljecki F, Ledoux H, Stoter J (2016) An improved LOD specification for 3D building models. *Comput Environ Urban Syst* 59:25–37
- Blut C, Blankenbach J (2021) Three-dimensional CityGML building models in mobile augmented reality: a smartphone-based pose tracking system. *International Journal of Digital Earth* 14(1):32–51
- Dehbi Y, Plümer L (2011) Learning grammar rules of building parts from precise models and noisy observations. *ISPRS J Photogramm Remote Sens* 66(2):166–176
- Dehbi Y, Hadji F, Gröger G, Kersting K, Plümer L (2017) Statistical relational learning of grammar rules for 3D building reconstruction. *Trans GIS* 21(1):134–150
- Dirksen J (2015) Learning Three.js—the JavaScript 3D Library for WebGL. Packt Publishing Ltd
- Eicker U, Zirak M, Bartke N, Rodríguez LR, Coors V (2018) New 3D model based urban energy simulation for climate protection concepts. *Energy and Buildings* 163:79–91
- Fan H, Kong G, Zhang C (2021) An Interactive platform for low-cost 3D building modeling from VGI data using convolutional neural network. *Big Earth Data* 5(1):49–65
- Flask (2021). <https://flask.palletsprojects.com/en/1.1.x/>. (Accessed 29 March, 2021).
- Furukawa Y, Ponce J (2009) Accurate, dense, and robust multiview stereopsis. *IEEE Trans Pattern Anal Mach Intell* 32(8):1362–1376
- Gröger G, Plümer L (2012) CityGML—Interoperable semantic 3D city models. *ISPRS J Photogramm Remote Sens* 71:12–33
- Haynes P, Hehl-Lange S, Lange E (2018) Mobile augmented reality for flood visualisation. *Environ Model Softw* 109:380–389
- Kim H, Han S (2018) Interactive 3D building modeling method using panoramic image sequences and digital map. *Multimed Tools Appl* 77(20):27387–27404
- Kong G, Fan H (2020) Enhanced Facade Parsing for Street-Level Images Using Convolutional Neural Networks. *IEEE Trans Geosci Remote Sens*. <https://doi.org/10.1109/TGRS.2020.3035878>
- Leaflet (2021). <https://leafletjs.com/>. (Accessed 29 March, 2021)
- Li L, Tang L, Zhu H, Zhang H, Yang F, Qin W (2017) Semantic 3D modeling based on CityGML for ancient Chinese-style architectural roofs of digital heritage. *ISPRS Int J Geo Inf* 6(5):132
- Li L, Lei Y, Tang L, Yan F, Luo F, Zhu H (2019) A 3D spatial data model of the solar rights associated with individual residential properties. *Comput Environ Urban Syst* 74:88–99
- Liu H, Li W, Zhu J (2021) Translational Symmetry-Aware Facade Parsing for 3D Building Reconstruction. arXiv preprint arXiv:2106.00912
- Liu J, Luo J, Hou J, Wen D, Feng G, Zhang X (2020) A BIM Based Hybrid 3D Indoor Map Model for Indoor Positioning and Navigation. *ISPRS Int J Geo Inf* 9(12):747
- Machete R, Falcão AP, Gomes MG, Rodrigues AM (2018) The use of 3D GIS to analyse the influence of urban context on buildings' solar energy potential. *Energy and Buildings* 177:290–302
- Mapbox (2021). <https://www.mapbox.com/>. (Accessed 29 March, 2021)
- Monteiro CS, Costa C, Pina A, Santos MY, Ferrão P (2018) An urban building database (UBD) supporting a smart city information system. *Energy and Buildings* 158:244–260
- Musialski P, Wonka P, Aliaga DG, Wimmer M, Van Gool L, Purgathofer W (2013) A survey of urban reconstruction. In *Computer graphics forum* (Vol. 32, No. 6, pp. 146–177)
- Nishida G, Garcia-Dorado I, Aliaga DG, Benes B, Bousseau A (2016) Interactive sketching of urban procedural models. *ACM Transactions on Graphics (TOG)* 35(4):130
- Park Y, Guldmann JM, Liu D (2021) Impacts of tree and building shades on the urban heat island: Combining remote sensing, 3D digital city and spatial regression approaches. *Comput Environ Urban Syst* 88:101655
- Redmon J, Farhadi A (2018) Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767
- Sangiambut S, Sieber R (2016) The V in VGI: Citizens or civic data sources. *Urban Plan* 1(2):141–154
- Stadler A, Nagel C, König G, Kolbe TH (2009) Making interoperability persistent: A 3D geo database based on CityGML. In *3D Geo-information sciences* (pp. 175–192). Springer, Berlin, Heidelberg
- Sun S, Salvaggio C (2013) Aerial 3D building detection and modeling from airborne LiDAR point clouds. *IEEE J Sel Top Appl Earth Obs Remote Sens* 6(3):1440–1449
- Tang L, Ying S, Li L, Biljecki F, Zhu H, Zhu Y, Yang F, Su F (2020) An application-driven LOD modelling paradigm for 3D building models. *ISPRS J Photogramm Remote Sens* 161:194–207
- Templin T, Popielarczyk D (2020) The use of low-cost unmanned aerial vehicles in the process of building models for cultural tourism, 3D web and augmented/mixed reality applications. *Sensors* 20(19):5457
- Tran H, Khoshelham K, Kealy A, Díaz-Vilarinho L (2019) Shape grammar approach to 3D modeling of indoor environments using point clouds. *J Comput Civ Eng* 33(1):04018055
- Ullman S (1979) The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153), 405–426
- Wang R, Peethambaran J, Chen D (2018) LiDAR point clouds to 3-D Urban Models: A Review. *IEEE J Sel Top Appl Earth Obs Remote Sens* 11(2):606–627
- Wolberg G, Zokai S (2018) PhotoSketch: a photocentric urban 3D modeling system. *Vis Comput* 34(5):605–616
- Wu B, Yu B, Wu Q, Yao S, Zhao F, Mao W, Wu J (2017) A graph-based approach for 3D building model reconstruction from airborne LiDAR point clouds. *Remote Sens* 9(1):92
- Xiong B, Elberink SO, Vosselman G (2014) A graph edit dictionary for correcting errors in roof topology graphs reconstructed from point clouds. *ISPRS J Photogramm Remote Sens* 93:227–242
- Yang B, Dong Z (2013) A shape-based segmentation method for mobile laser scanning point clouds. *ISPRS J Photogramm Remote Sens* 81:19–30
- Yu Q, Helmholtz P, Belton D (2017) Semantically enhanced 3D building model reconstruction from terrestrial laser-scanning data. *J Surv Eng* 143(4):04017015

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.