Automation in Construction 156 (2023) 105092

Contents lists available at ScienceDirect

Automation in Construction

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

Automation of escape route analysis for BIM-based building code checking

Simon Fischer *, Christian Schranz, Harald Urban, Daniel Pfeiffer

Research Unit Digital Building Process, TU Wien, Karlsplatz 13/E235-03, Vienna, 1040, Austria

ARTICLE INFO

Keywords:
BIM
IFC
Code checking
Escape route analysis
Navigation models
Route planning

ABSTRACT

The use of BIM is becoming increasingly important in the AEC industry because the included dataset enables the automation of many processes. One emerging application of BIM is the automation of building code checking by building authorities. Building code checking is a time-consuming process, particularly for escape routes. Therefore, its automation promises considerable optimisation potential. This study presents a semi-automated escape route analysis for building code checking. Our developed concept consists of two automated steps, with a legally required manual step in between: (1) escape route generation, which provides possible routes from which designers can select a valid combination, and (2) escape route validation, which checks the selected combination of escape routes for validity. The basis of both automated steps is a novel indoor navigation model. We implemented the concept in Solibri Office and validated it with real-world models. The developed escape route analysis method shows potential to improve designers' escape route planning and building authorities' escape route checking.

1. Introduction

Modern buildings, building processes, and dependencies between disciplines are becoming increasingly complex. Building Information Modeling (BIM) is a promising way to deal with this growing complexity. BIM models provide numerous project information datasets. This digital information can facilitate various automatic evaluations of building processes, leading to the potential for optimisation of the architecture, engineering, and construction (AEC) industry. The use cases of such automatic evaluations are spread over the entire life cycle of a building: clash detection of domain models in the planning phase [1,2], generation of material bills in work preparation [2], construction safety checking in execution [3,4], and energy simulations for operation [5–7]. The building permission process, an essential stage in the early phase of the life cycle of a building, has remained largely untouched by the new possibilities of automatic evaluation using BIM [8]. In this context, building code checking is a complicated, time-consuming, and repetitive activity with the potential for significant optimisation through automation [8,9]. Eastman et al. [9] reported that the growing complexity of buildings and building processes corresponds with new criteria for building codes and safety, making the analogue assessment of buildings with two-dimensional (2D) plans even more time-consuming.

The potential of automating building code checking based on digital building models has already been recognised, but is rarely implemented by building authorities. There are different approaches to developing such systems. Several researchers have addressed the question of how provisions and requirements written in natural language can be represented in a computer-interpretable format [10–14]. Other studies have focused on building model preparation [9,10] or enhancing the accessibility of data in the building model [15]. While these approaches address specific parts of the entire process, there have also been projects to implement an automated code-checking system on a more general level [9,16,17]. However, according to Beach et al. [8], practical implementation of these studies is scarce. The analogue assessment of building submissions is still state-of-the-art for most building authorities [18,19]. An exception is Singapore's CORENET project [20]. A recent Austrian research project is BRISE-Vienna, which was funded by the European Union Initiative Urban Innovative Actions (UIA) [21]. It aims at developing a complete building permission process for the building authority in Vienna based on digital building models in open formats, that is, an openBIM-based permission process. BRISE-Vienna includes all parts of the process, that is, from submitting the digital model via a web service over a semi-automated code-checking system [22] to the dissemination of information using modern technologies such as augmented reality (AR) based on digital building models [23,24]. The core of the project with the greatest potential for improvement in accuracy and time consumption is a semi-automated code-checking system. In this context, semi-automated means require final decisions from the building authority agents because of legal requirements.

A major issue in code compliance checking is the assessment of escape routes. Many regulations and requirements for escape routes

* Corresponding author.
E-mail address: simon.fischer@tuwien.ac.at (S. Fischer).

https://doi.org/10.1016/j.autcon.2023.105092
Received 29 October 2022; Received in revised form 19 July 2023; Accepted 9 September 2023
Available online 21 September 2023

are interwoven to ensure that all possibilities are covered. Therefore, automation can optimise the assessment of the benefits in terms of time consumption and consistency.

Escape route analysis is a special case of indoor pathfinding with additional requirements. Previous studies on this topic have shown promising results [14,25–28]. Moreover, the widely used BIM coordination and validation application Solibri Office [29] provides a checking rule to generate and check escape routes. However, an analysis of the legal situation in Vienna showed that these attempts were insufficient for a complete BIM-based permission process, as in BRISE-Vienna. A checking rule for escape route analysis must meet the essential requirements for the course of the escape routes. Examples include the definition of a starting location inside a room and the consideration of safe zones in a building. In addition, the exact geometry of the escape routes, which includes the position of fixed locations and the connections between them, must follow certain specifications. Furthermore, a characteristic of escape route analysis is the geometrical requirements of the elements along the routes, which depend on the number of people passing by. Thus, escape route analysis must include element checks based on the cumulative number of people from different starting locations. Finally, the Vienna Building Authority requires designers to specify escape routes that satisfy the regulations because designers are liable for them. A fully automated generation of escape routes represents a liability reversal, where the building authority is liable for the defined escape routes. This is not feasible. Therefore, the developed system must include functionalities for designers to define the course of routes.

The objective of this study is to develop an automated escape route analysis method that meets the criteria of a BIM-based building permission process and complex building regulations like in Vienna, enabling its implementation within the Vienna Building Authority. Thus, the approach must comply with the legal situation in Vienna (legal requirement), analyse escape routes at an appropriate time (technical requirement), and be accepted by the involved stakeholders, that is, the building authority and designers (social requirement). We developed a concept consisting of escape route generation to determine several possible routes and escape route validation to assess a network of selected routes. This approach aims to include the sophisticated geometrical calculations required in the analysis. Simple additional non-geometrical property checks of the elements along the routes (e.g. fire resistance) can be covered by existing checking rules. Moreover, maintainability is a significant concern because the Vienna Building Authority wants to maintain the checking rules. Therefore, the research team (including the building authority and designers) decided to exclude some special cases from the scope of the checking rules to facilitate maintainability. The building authority keeps these special cases for manual verification with its agents.

This study focuses on the development of a novel approach and provides a brief overview of the status of the Vienna Building Authority, previous endeavours in automatic escape route analysis, and the BRISE-Vienna project. Subsequently, the stages for creating a checking rule from the building code text are discussed. The main section explains the requirements and concepts of our escape route analysis method in detail. For the validation of the new approach and its inclusion in the permission process developed in the BRISE-Vienna project, we implemented the escape route analysis method in Solibri Office. Finally, we present and discuss the implementation results.

2. Background

2.1. The current status and requirements of escape route checking within the Vienna Building Authority

Digitisation and automation are not yet part of escape route checks within the Vienna Building Authority. It is still a manual visual process based on 2D information on building plans (floor plans and sections) and supplementary documents. The building authority considers an escape route to be a polyline consisting of straight lines. These escape routes must meet specific requirements, defined in three different regulations (OIB guidelines [30] in Austria): general fire protection (OIB 2 [31]), fire protection in parking garages (OIB 2.2 [32]), and user safety and accessibility (OIB 4 [33]). These regulations define the starting locations, safe zones, and destinations. A destination is always a direct exit to a safe area outside a building. A safe zone inside a building can be an emergency staircase that leads to a direct exit. Safe zones are interim destinations for escape route analysis because the escape route length is measured to these rooms. However, the route must also be calculated from the safe zone to the actual exit because the clear width, height, and elements along the path must also be checked in this section. Any spot in each common room in the building must reach a safe zone or destination within the required distance, that is, the maximum route length. A special starting point is the parking space inside a parking garage. In an emergency, it must be assumed that parking spaces are occupied by vehicles and, therefore, they are obstacles to the escape route. However, it must be possible to start the route from an occupied parking space.

An exception for the starting point is defined for flats with a maximum number of two levels and without a direct exit to the outside. For these flats, the escape route begins at the flat entrance door. A very important requirement for the course of the escape route is that inside rooms, the route requires the shortest polyline to connect any two points. This means that the escape routes run directly from corner to corner. Finally, there are many requirements for elements along escape routes. Some specify limits on element dimensions, such as the minimum width and height of doors, rooms, and staircases. Others refer to the fire resistance of these and other adjoining elements such as walls and ceilings. The characteristic of the minimum values of the object dimensions is their dependency on the number of people passing through the object while escaping.

Designers must define and highlight escape routes manually in building plans. These escape routes are checked by the authority in accordance with the specifications. The current check begins with a visual check by the building authority agents. They must detect legally required starting positions, permitted safe zones, and permitted fire exits using this visual check to determine whether the designers have placed the relevant routes. This step also includes a property check of elements (e.g. fire resistance) in and around special areas and along the given routes. The subsequent procedure of verifying the dimensions and geometry of the elements and rooms, as well as the length of the escape route, is particularly time-consuming because building authority agents must perform manual measurements for all escape routes. This task does not require human interpretation and is imprecise when performed manually. Therefore, it is suitable for automation and can be optimised.

2.2. Related work on escape route analyses

Existing work on automated escape route analyses based on digital building models provides promising approaches, but none has developed a complete checking system. There are three components for such an escape route analysis: an indoor navigation model, a pathfinding algorithm, and an element-checking procedure along the determined routes. The basis of this is an indoor navigation model. The representation of indoor circulation is complex because of the free movement in a continuous space [34–36]. This differs from circulation on streets, where particular paths with defined directions are given. The movement of people indoors is not restricted to these paths. Therefore, a continuous space must be converted into an indoor navigation model. The development of indoor navigation models has been the focus of research for several years [25,26,28,34,35,37–45]. A widely used type of navigation models are navigation graph networks based on graph theory [35]. A graph is defined by nodes representing fixed locations

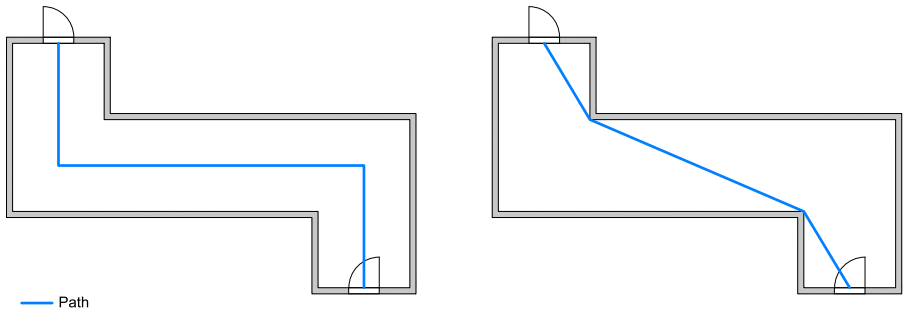


Fig. 1. Indoor navigation graph concepts: centreline-based (left) and visibility-based (right) graphs.

and the edges connecting these nodes. Navigation graph networks have advantages over other navigation models in terms of storage space and hold both semantics and geometrical information [35,37,46]. Currently, two approaches for generating indoor navigation graph networks have been discussed in the literature: centreline-based (medial axis transformation) [34,35,47,48] and visibility-based methods [25, 41,43–45,49] (Fig. 1). There is a lack of consensus among researchers regarding the most suitable method for escape route analysis. Some claim that centreline-based graph networks are consistent with human cognition [25,35,50] and building code requirements [25,26,35], and are, therefore, suitable for an escape route analysis. In contrast, Kneidl et al. [43,45] stated that visibility graphs are the basis for a realistic representation of human behaviour. The visibility graphs also comply with the way in which the Vienna Building Authority defines escape routes (straight connections from corner to corner).

The next step in escape route analysis is to apply a pathfinding algorithm to the indoor navigation model. This step has been included in previous studies [25,26,28,41–45]. Several researcher [41–45] have focused on simulating pedestrian dynamics, such as evacuation scenarios. These approaches are based on visibility graphs and are enhanced with dynamic edge weights based on the actual travel time, which is dependent on the density in a specific area (number of pedestrians in the area). Thus, it is simulated that large crowds slow down the routes. Kneidl et al. [43] additionally considered different levels of familiarity with an environment. Although this approach is a promising way to simulate actual human behaviour, it does not reflect the requirements of Vienna's building code. The building code focuses on travel distance along a defined escape route rather than on pedestrian dynamics. Lin et al. [28] developed an indoor navigation model rich in semantic information relevant to escape routes, such as the status of doors, the function of spaces, and the risk of obstacles. Furthermore, they applied different pathfinding algorithms and compared their performances. Lee et al. [25] developed a universal circulation network (UCN) that focused on general circulation inside buildings. They applied a pathfinding algorithm to calculate the walking distances between different locations. The UCN can be used for escape route analyses but does not address building code requirements (e.g. starting from the most remote point inside a room). In contrast, Kannala [26] analysed different fire codes from Northern America and Finland and determined five essential requirements an escape route analysis must address: number of occupants, minimum number of escape routes from one starting position, maximum travel distance, minimum width, and minimum height of escape routes. The focus on these universal demands makes the approach applicable to various fire codes in different countries but does not cover all regulations concerning escape routes. The approaches in Lee et al. [25] and Kannala [26] were both implemented in Solibri as an extra module. At present, Solibri provides its own escape route analysis for digital building models.

The approach of Kannala [26] and that provided by Solibri also include parts of the third component of an escape route analysis, that is, the element checking. Elements along escape routes must meet several requirements concerning their dimensions (width and height) and properties (e.g. fire resistance). This distinguishes escape route analysis from basic indoor circulation. Although the two approaches include all three necessary components, they do not cover the specific requirements of Vienna's building code, such as the handling of parking spaces inside parking garages.

Similar topics of interest in automated indoor path finding include dynamic emergency evacuation and rescue planning [47,48,51,52]. Research on this topic focuses on pathfinding in the event of an emergency using real-time information in an indoor navigation model. Although these concepts consist of the same three components as escape route analyses, the requirements for each component are different; for example, dynamic emergency evacuation considers topological connections between rooms in the navigation model for fire-spread simulations.

This section presents interesting approaches to escape route analysis. However, none of the approaches fully satisfied the requirements of the procedure in Vienna; therefore, they were insufficient for the BRISE-Vienna permission process. A novel indoor navigation model that represents the checking of the building authority using direct connections with an additional width check of the connections is required. This can be achieved by combining a visibility-based approach and a width check for the edges. The approaches of Höcker et al. [41], Borrmann et al. [42], and Kneidl et al. [43,44,45] already consider a width check, but are based on density instead of defined minimum dimensions. Furthermore, all reviewed approaches involve the automatic generation of escape routes or simulations of pedestrian flow. The legal situation in Vienna requires a system in which the designer decides the course of the escape routes. This strengthens the need for further enhancement of the aforementioned concepts to develop a comprehensive escape route analysis method that is suitable for code checking.

2.3. BRISE-Vienna

The BRISE-Vienna project aims at developing a semi-automated digital permission process based on openBIM standards. The project included the entire process, from online submission over a semi-automated code-checking system to the visualisation of the impact of the planned project with AR. Detailed information regarding this process is provided elsewhere [22,23,53,54]. This section focuses on the relevant information for developing a semi-automated escape route analysis.

The objective of the code-checking system is the semi-automated checking of a submitted BIM model for compliance with the building code. The submitted BIM model is called the building application model (BAM). The BAM is not an additional individual domain model. Instead, the architecture model is used with additional geometric (level of geometry (LOG)) and alphanumeric information (level of information (LOI)) requirements for building permission. The basis for the LOG and LOI of the BAM is the requirement for submittal planning, LOG300 and LOI300, respectively. The BIM model, as an information basis, has a

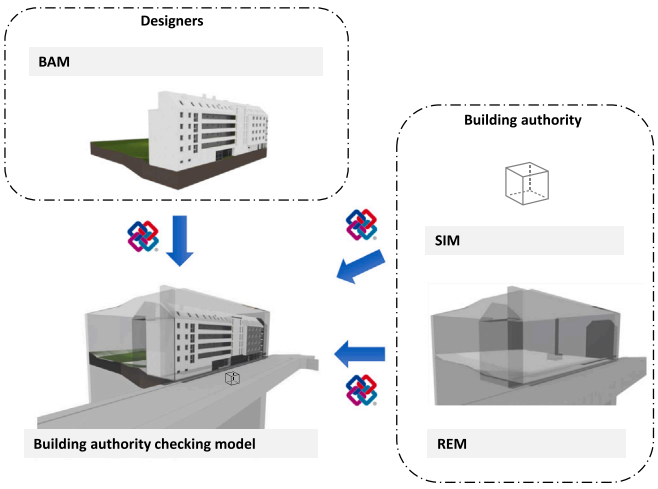


Fig. 2. Combination of BAM, REM, and SIM within the building authority checking model.

significant advantage over 2D plans; it holds geometric and semantic information, which allows legal compliance checks in an automatic manner [9,16,17]. For the permission process, the BIM model must follow open standards (openBIM) because the building authority (in Austria) does not require designers to use specific software. Therefore, the ISO-certified Industry Foundation Classes (IFC) format [55] is the basis for BIM models used in the permission process, which is the globally accepted standard for the exchange of models in openBIM projects.

In addition to the BAM, the code-checking system requires a second data source, that is, a computer-readable representation of the building code [16]. These regulations were formulated with the objective of being understandable to humans, not computers. Hence, human interpretation is required. Building codes must be adapted to enable fully automated systems in the future. In addition, according to Beach et al. [8], there is a lack of trust in automated systems. Therefore, the developed code-checking system must be semi-automated with automated calculations in combination with human oversight and modification. In general, the building code includes two types of regulations: technical specifications for the building itself and the building potential of a construction site (e.g. building alignment, permitted building height, and required shape of the roof). All the information required to check the technical specifications is mandated to be in the BAM. To check the building potential of a construction site, geometrical and alphanumeric restrictions are required in a computer-readable form. This is included in the three-dimensional (3D) reference model (REM) and the service information model (SIM). The REM is the 3D envelope of the permitted building space. Overlaying the REM and BAM enables checking all geometric specifications of the building site. In contrast, the SIM includes alphanumeric specifications for the construction site (e.g. parking space regulations). It is also described by an IFC file excluding the geometry information. The building authority generates both REM and SIM after the legal submission of the BAM. Together, these three models are the inputs for the code-checking system (Fig. 2).

In the BRISE-Vienna project, the code-checking system was developed in the software Solibri Office, which provides implemented checking rules for common use cases. Most of the regulations in the analysed building codes can be checked using a combination of these rules. This significantly reduced the number of necessary newly developed rules in the BRISE-Vienna project. However, some regulations, such as escape route analysis, require entirely new functionalities, for which Solibri Office provides an application programming interface (API).

3. Rule-developing stages

In the BRISE-Vienna project, the research team used a four-stage process to develop new complex checks that cannot be realised with existing rule templates of Solibri Office. Examples include rules for building code compliance checking of escape routes, fire compartments, light exposure, clear view of windows, the height of fall protection, width of stair landings, and detection of flat partition walls. The first stage concerned the building code analysis. Subsequently, concepts for different components of the checking rule were developed. Software implementation was performed in stage three, and finally, the results were validated.

3.1. Building code analysis

The first stage of the rule development process was the building code analysis. At this stage, together with checking software experts and building authority agents, the feasibility of automated checking was analysed for each regulation in the building code. The objective is to identify all regulations that can be represented by a checking rule and the related required information in the building models. The relevant building codes in BRISE-Vienna are the Vienna Building Code [56], its subsidiary Vienna Garage Law [57], the Zoning and Development Plan, and the aforementioned OIB guidelines (building regulations) [30]. Not every regulation within a building code can be checked fully automated. One limitation is that some expressions require interpretation [11,14]. The critical aspect in determining whether a regulation requires interpretation is the target value. Qualitative values, such as “to a sufficient extent”, found in Vienna’s building code, cannot be determined without interpretation [11,12], either from humans or artificial intelligence (AI) approaches. Because the legal situation in Vienna forbids decision-making through AI, only regulations with quantitative target values are suitable for full automation. However, the code-checking system can still assist the building authority agents with regulations that contain qualitative target values. It can highlight relevant objects with qualitative target values and thus reduce a building authority agent’s search effort.

According to Macit İlal and Günaydın [11], there are two approaches for structuring code representations. The first approach, which was reported by Han et al. [10], used a structure similar to that of the building model. The components of a particular class can be easily checked against all rules applicable to that class. This approach has advantages in rule execution, which accelerates the checking process [11]. On the other hand, the approach of Nisbet et al. [13] maintained a code representation similar to the structure of the building code. This provides a higher level of maintainability in case of regulatory changes [11]. In the BRISE-Vienna project, the approach of Nisbet et al. [13] was applied because maintenance is a significant concern, since the Vienna Building Authority wants to maintain all checking rules.

However, some regulations are too interwoven to be considered individually. This applies to escape route analyses. Specifications for escape routes are spread over three different OIB guidelines (building regulations); however, they still depend on each other. Because of the close relationship between different requirements, some regulations are merged into a single checking rule.

Another limitation of automated verification of regulations is the information available in building models [12]. Therefore, all information required for automatic checking must be identified and included in the LOG and LOI requirements of the models. For nonstandard information, the IFC scheme provides the functionality to define custom objects (*IfcBuildingElementProxy*) and properties to include arbitrary information in the model.

The result of the code analysis was the regulation information matrix (RIM). This includes the interpretation of each regulation (if and how it can be checked automatically), required checks with the corresponding LOG and LOI requirements, and dependency on other regulations. Thus, the RIM is the basis for the development of the checking rules.

3.2. Rule conception

In the rule conception stage, the concepts required to represent a regulation as a checking rule were defined. For escape route analysis, concepts are required for the general process of the checking rule and its three components: the navigation model, pathfinding algorithm, and element-checking procedure. These concepts are described in detail in Section 4.

The rule conception stage was executed in close collaboration with the building authority agents and designers. The interests of the designers were represented by the Austrian Chamber of Architects and Civil Engineers and one of the major Austrian consulting planning offices, both of which were partners in the BRISE-Vienna project. The integration of the building authority agents ensures that the assessment using the developed checking system meets the standards of the current manual assessment. The interest of designers is that the effort to include additional nonstandard information in the building models required for automated assessment is compensated by other benefits. The integration of both stakeholders is important for the acceptance of the BIM-based permission process.

3.3. Software implementation

After the structure and the concepts of a rule were defined, the rule was implemented in the BIM checking software Solibri Office. The API provided by Solibri enables the development of new checking rules based on the programming language Java. The use of Solibri Office for new checking rules follows the general use of Solibri Office in the BRISE-Vienna project. The implementation of our escape route analysis method is described in Section 5.

3.4. Validation

The final stage of developing a new automated or semi-automated checking rule involves validating the results. Three evaluation criteria were used for this purpose. The first concerns the legal requirements of the building authority and the correct representation of the building code (defined in the RIM). For escape route analysis, this includes compliance of the general process with the legal requirement that designers must predefine escape routes. Moreover, the navigation model must provide geometry and connectivity that comply with the way escape routes are defined by the building authorities (including width checking). The pathfinding algorithm must connect each starting position to the nearest exit using the shortest possible path. Finally, the element-checking procedure must include dimension checks depending on the number of people. These legal requirements are described in detail in Section 2.1.

The second factor is the duration of the checking process (technical requirements). The authors wanted to compare the time required for the semi-automated escape route analysis with the time required for manual checking. Owing to legal labour reasons, permission to measure the time for manual checking was not granted. Hence, only the time required for BIM-based checking is given.

The third criterion is acceptance by stakeholders of the permission process: building authority agents and designers. For the building authority agents, this includes the comprehensibility and reproducibility of the escape routes because they must make decisions based on the provided results. For designers, the benefits and additional effort of the new approach are the most important factors for acceptance.

To evaluate the first two criteria, a checking rule was applied to fictional and real-world test models. The fictional models included test cases that represented the individual specifications defined in the building code. The general functionality of the developed checking rule was verified using these models. In addition, during the pilot phase, real-world models tested the checking rule under realistic conditions (applied to designers' building models and used by building authority agents). An evaluation of the escape route analysis method is presented in Section 5.

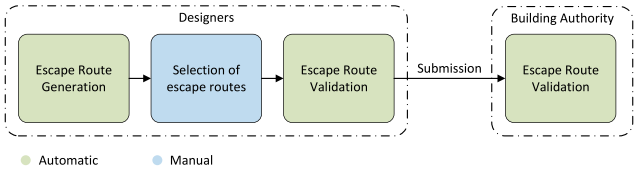


Fig. 3. The escape route analysis process.

4. Escape route analysis

In this study, a process which allows human interference between two automated calculations was developed (Fig. 3). The first automated step is the escape route generation. It determines the k shortest possible escape routes from each starting location. These shortest routes are individually checked. This indicates that if two routes use the same door, the door width is checked separately for the number of people in each route and not combined. Escape route generation is a tool that assists designers in selecting desired escape routes. To optimise assistance, the generation can either show the k shortest routes that passed the check or routes that failed the check. This helps designers understand why a route is rejected and suggests model adaptations.

The second automated process is the escape route validation. This method automatically generates escape routes along predefined sequences of doors. Therefore, designers must select the preferred escape routes between the two automated steps by defining their door sequences. This input makes designers liable for escape routes, and thus, the escape route analysis method is applicable to the new BIM-based permission process of the Vienna Building Authority. The information is stored either in a list, which can be imported into the rule or directly in the digital model. The second approach was followed in the BRISE-Vienna project. This required a new property of doors: a reference to the next door on the escape route. After the escape routes are reconstructed from the predefined door sequences, element checking is performed for the cumulative number of people on all routes passing an element. Therefore, escape route validation ensures that the building design provides a valid network of escape routes. It is used by designers to validate their selected escape route network before submission and by the building authority agents for code checking after submission. Based on the validation results, the building authority agents can make final decisions. The following sections describe the operating principle of the escape route analysis method, which can be broken down into three core components:

- Generating an indoor navigation model,
- applying a pathfinding algorithm, and
- checking of the elements along the escape routes.

A new requirement for digital building models was identified while developing the checking rule. Building regulations allow doors to be used as starting elements, requiring the determination of the number of people starting from these locations. Therefore, a new property, that is, the occupancy of doors, was included in the BRISE-Vienna project.

4.1. Navigation model

The first component of escape route analysis is the navigation model. In the proposed escape route analysis, the navigation model uses the visibility-based approach, similar to the UCN of Lee et al. [25]. Our developed node network consists of separate room-by-room node networks, which are connected by special access nodes at doors and openings for the horizontal connection, and stairs and ramps for the vertical connection. Elevators were not included in the navigation model because they are not used in the case of an emergency. Each room network consists of nodes at the corners of the outside boundary and the corners of holes from, for example, the columns inside

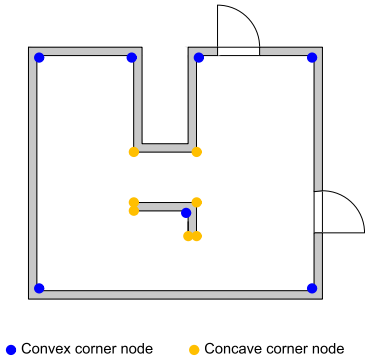


Fig. 4. Placement of the concave (in all rooms) and convex (only in starting rooms) corner nodes.

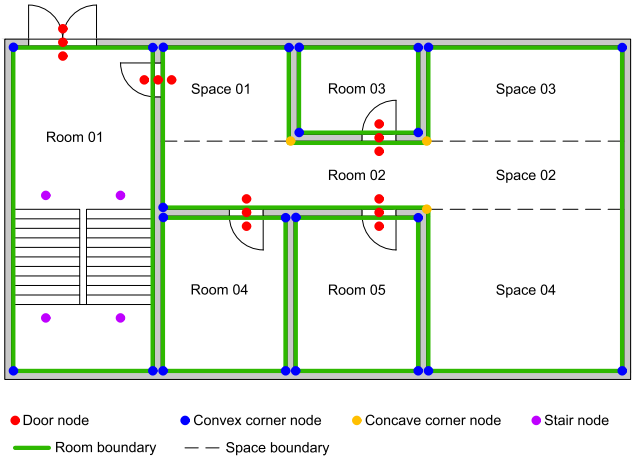


Fig. 5. Creation of nodes at room bounding polygons, doors, and stairs.

the room, as well as the access nodes of the room. The difference from the UCN is that our approach is not restricted to door-to-door measurements using only concave corner nodes. Concave corner nodes are the intermediate coupling points between two nodes whose line of sight is obstructed [25]. Building codes surveyed in Lee et al. [25] and the Vienna building code indicate that in a starting room, the escape route begins at the most remote point from the door. This point is always located at a convex corner. Therefore, our room network is not restricted to concave corner nodes (such as the UCN), but also contains convex corner nodes for the starting rooms (Fig. 4).

4.1.1. Node creation

The creation of room-by-room node networks is based on five IfcEntities: *IfcSpace* for room nodes, *IfcDoor* and *IfcOpening* for horizontal access nodes, and *IfcStair* and *IfcRamp* for vertical access nodes. The room nodes are derived from space boundary polygons. Each corner point represents either a concave or convex room node. Special attention is paid to interconnected spaces. Designers commonly divide spatial objects into subspaces without walls between them. Such interconnected spaces are merged into one room with one room-boundary polygon (see room 02 in Fig. 5). In this way, the room boundary polygon can be used for neighbour node assignment between all nodes of the interconnected spaces.

For a horizontal room connection, both doors and openings require at least two nodes, one in each room. The nodes are placed at a buffer distance from the centre of the element perpendicular to the wall. As reported in Lee et al. [25], this is the most reasonable approach for passing through doors. The Vienna Building Authority also shares the same view. The buffer distance is at least half the wall thickness to ensure that the nodes are inside the adjoining space object. Additionally, doors include a third node in the door centre to represent the start or end node of an escape route. Openings do not require this extra node, as they are not suitable as starting or destination elements as they are not valid room-separating objects.

The representation of stairs and ramps is more complex because the geometry can vary depending on their type. Stairs and ramps comprise flights and landings. We restrict the movement on the stairs and ramps to a flight-by-flight movement. This implies that when a flight is entered, one must follow it to its end. On every landing, one can either continue with the subsequent flight or leave the stairs if another adjoining element exists. Thus, every flight requires only bottom and top nodes. Similar to horizontal access nodes, flight nodes are placed in front of flights at a buffer distance to ensure that they are inside the rooms which they connect. Although these nodes can simply be connected on straight flight elements, curved elements require elaborate calculations. Our approach simulates curved stair flights using polylines with vertices at the centre of each tread nose. The polylines provide both flight length and geometry for visualisation. However, curved ramp flights were not considered in the study.

4.1.2. Neighbour node assignment

In our concept, neighbour nodes are considered as nodes with a direct connection to each other that complies with the width requirements of the building code. The developed indoor navigation network is a room-by-room node network; therefore, the neighbour assignment is also room-based. This implies that every node, not only the room nodes, must be inside a room. The horizontal and vertical access nodes have automatic neighbour relations with the second node of their elements. This is also the case for flights of one IfcStair object because checking the geometry of intermediate landings is part of a separate checking rule. However, access nodes must also be assigned to a room in order to be related to other nodes in the room. The advantage of a room-by-room node network is that neighbour node assignment is only performed for all nodes inside a room. Nodes in other rooms are not considered, because they can only be reached by access nodes. Neighbour assignment inside a room is a two-step method:

1. Visibility check and
2. flexible-tube width check.

These two steps are performed for each pair of nodes inside the room. The visibility check narrows down possible neighbour nodes by creating a straight line between the two observed nodes and checks whether this line intersects the space-bounding polygon or obstacles (hole polygons). If no intersection is detected, the nodes are visible to each other (Fig. 6). However, visibility alone is insufficient for determining neighbouring nodes. Vienna building regulations define the minimum width required by pedestrians to walk along the escape routes. Visibility does not indicate anything about the unobstructed width around the direct connection between two nodes. Therefore, the unobstructed width around the connection must also be checked. This was realised through a flexible-tube width check. A flexible tube is defined as the minimum free area required around a direct connection between two nodes. Because room nodes are created exactly at the corner points, this free area cannot be divided equally between the two sides of the connection. One side of the tube is completely obstructed at the corners. The flexible tube provides the functionality to continuously shift the free area from one side of the straight connection to the other (Fig. 7). For this purpose, an observation area A_O with the minimum required width w_{min} on both sides of the straight connection is created (Fig. 8, left). The observation area is defined as the area in which a flexible tube could be placed. The sharpness of the shift from one side to the other is restricted by a circular arc with a radius of w_{min} (Fig. 8, centre). This ensures that a circle with a diameter equal to the free-area width fits into the tube at every position (Fig. 8, right). If such a flexible

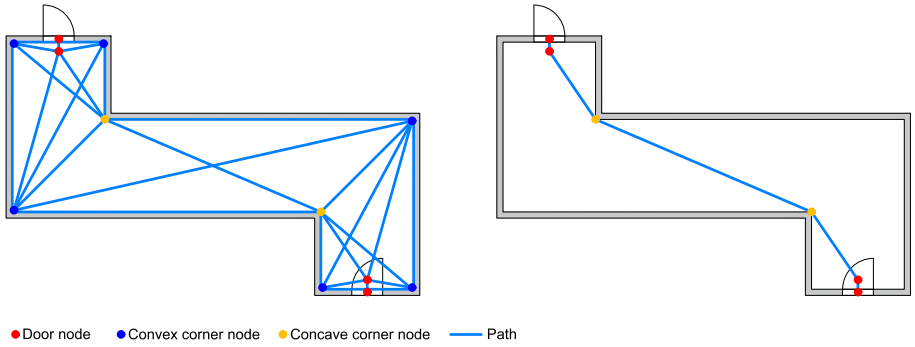


Fig. 6. Comparison between node connections based on visibility in starting rooms with convex corner nodes (left) and arbitrary rooms without convex corner nodes (right).

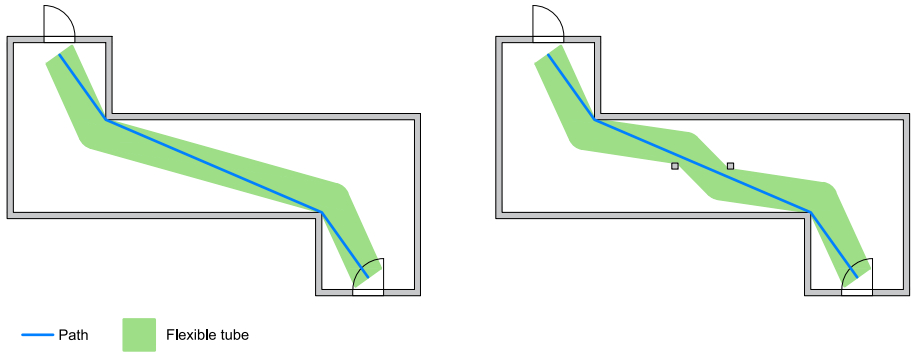


Fig. 7. Shift of the flexible tube along the escape route: complete shift without (left) and with (right) obstacles.

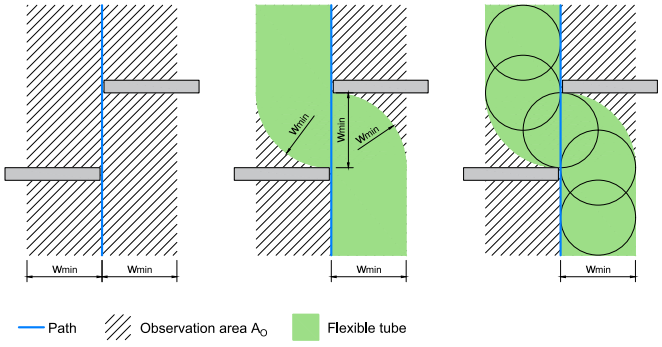


Fig. 8. Flexible-tube concept: observation area A_O around the direct connection (left), sharpness of the shift (centre), and circles fitting into the tube at every position (right).

Algorithm 1: Neighbour Node Assignment

Input : Set of nodes N of a room with length n; Set of boundary and hole polygons P of a room; minimum required width w_{min}
Output: Graph G(N,E) of a room

```
1  add all nodes in N to G
2  for i = 0 to n - 2 do
3      for j = i + 1 to n - 1 do
4          check whether the direct connection between nodes
            N[i] and N[j] is obstructed by any polygon p in P
5          if the direct connection is not obstructed then
6              define observation area A_O with w_min on both sides
                of the direct connection
7              if a circle with diameter w_min can be placed
                continuously along the direct connection inside A_O
                without intersecting any polygon p in P then
8                  add edge e(N[i],N[j]) to G
9              end if
10         end if
11     end for
12 end for
```

tube can be created around the straight direction without colliding with any building elements, the two observed nodes are related as neighbour nodes. Algorithm 1 shows the pseudocode for neighbour node assignment. Lines 5–10 in this algorithm indicate the flexible-tube width check, which represents the novelty of the algorithm.

However, the flexible-tube concept is limited to concave corner nodes and access nodes. It cannot be applied to convex corner nodes because the possible width around a point approaches zero at the convex corners. Therefore, the neighbour assignment for starting rooms with convex corner nodes uses only the first step of the developed method, that is, a visibility check.

4.2. Pathfinding in the escape route analysis

The pathfinding algorithm is the centrepiece of escape route analysis because it defines the logic of how to navigate through a given navigation model. In general, pathfinding algorithms are used in graph theory to solve shortest path problems. They connect pairs of nodes to the best path based on predefined criteria [58]. This functionality makes them extremely important in navigation systems, video games, robotics, logistics, and crowd simulations [59]. Different use cases have different characteristics, which can be classified into several shortest-path problems, such as the single-source shortest-path problem, single-destination shortest-path problem, or all-pairs shortest-path problem [58]. Different algorithms with advantages for different use cases have been developed to optimise the handling of different

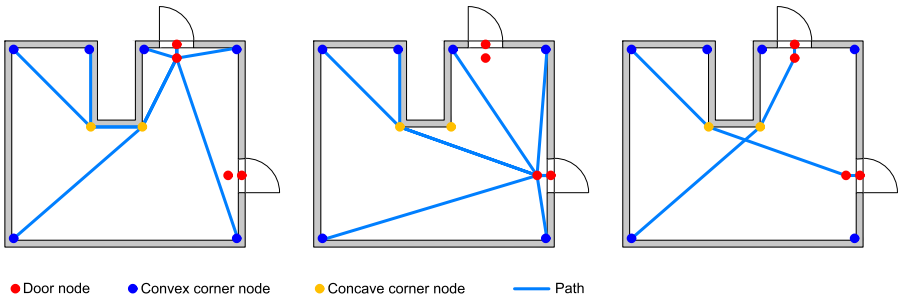


Fig. 9. Finding the source nodes in a starting room: Paths to all convex corner nodes from the top door (left), to all convex corner nodes from the right door (centre), and to the two most remote convex corner nodes (right).

pathfinding problems. Therefore, it is crucial for selecting an algorithm to determine the corresponding shortest-path problem. Considering the characteristics of escape route analysis, the corresponding shortest path problem lies between those mentioned above. First, we do not examine a single source or destination. However, the number of sources and destinations is minimal compared to the total number of nodes in the network. Noting that there is no clear solution for classifying escape routes, the research team considered the single-source shortest-path problem run for each starting location to be a suitable representation.

The foundation for solving single-source shortest-path problems is Dijkstra's algorithm [60], which was devised in 1959. Dijkstra's algorithm iterates through all nodes in a graph until the shortest path to all nodes is determined [58].

In our escape route analysis method, Dijkstra's algorithm supports different stages of the route detection process. The first application of Dijkstra's algorithm is the definition of source nodes inside a room. For a specific exit door, the most remote point inside the room is the node with the longest shortest path to the exit-door node. This graph problem is called the single-destination shortest-path problem [58]. This can be transformed into a single-source shortest-path problem by changing the source and destination nodes; thus, it can be solved using Dijkstra's algorithm [58]. The algorithm returns the shortest path from the exit door to each node in the room. The destination node with the longest shortest path is the source node for the actual escape route analysis. For rooms with multiple doors, the relevant source nodes depend on the selected exit door. Therefore, in rooms with multiple doors, each door is connected separately to a source node (Fig. 9).

The second application of Dijkstra's algorithm is in the shortest path problem between given starting locations and destinations. While escape route generation solves the shortest path problem for the entire graph, escape route validation divides the path determination into several smaller shortest path problems, one for each pair of doors of the given door sequence. In this manner, the shortest path from door-to-door is computed and finally joined into one path along the predefined door sequence.

After the shortest path calculation, escape route generation requires additional calculations. Next, the paths from one starting location to different destinations are sorted in order of length. This list already contains as many paths as the destinations (e.g. several fire exits). Nevertheless, they cannot be taken as the k shortest paths from the source node since the second shortest path to the "nearest" destination could be shorter than the shortest to the second "nearest" destination. Therefore, an actual k-shortest-path problem from one source node to each destination node is required. This is implemented using Yen's algorithm [61]. It can determine the k shortest paths from a source node to a destination node without any loops in these paths. It is an enhancement of shortest path algorithms and can be implemented based on any shortest path algorithm. In Yen's algorithm, the proper definition of the minimum deviation between different paths is critical. Typically, in Yen's algorithm, changing a single node leads to a new path [61]. In escape route analysis, this is insufficient because a path would be considered new, even if it uses the same rooms and doors, by simply taking a diversion inside a room. Therefore, a more suitable minimum deviation is the use of different doors. To implement this, the logic of Yen's algorithm must be applied only to door nodes. Thus, the k-shortest path does not pass the same sequence of doors as any previous path.

4.3. Element checking

The third core function of escape route analysis is to check the dimensions of the elements along the routes. The relevant elements include rooms, doors, openings, stairs, and ramps. For all five relevant elements, the width requirements depend on the number of people using the element. In addition, doors have person-dependent requirements for the opening direction. Escape route generation provides no information on which routes belonging to one route network. Therefore, it makes no sense to consider the interactions between routes. Instead, the elements of each route are checked based on the people using the particular route. In contrast, escape route validation is performed on a defined route network with interacting routes and can therefore check the elements along the routes based on all people using the element. This requires identifying all routes using a particular element and summing all their people. Subsequently, the width of an element can be compared with the required width retrieved from the cumulative number of people. However, doors, openings, stairs, and ramps provide explicit information about their width, which allows a simple comparison of the two values. The width of a room is not available. A feasible measure to check for the minimum required space in a room is the use of the developed flexible-tube concept. For application in the room-width check, the required width of the flexible tube is computed using the number of people in the room and is then applied to all the room's route sections. There are additional element requirements; however, these are independent of the number of people. Examples include the heights of the doors and openings. These can be checked equally for generation and validation by simply comparing the actual and required values.

5. Solibri implementation

The proposed concept for automated escape route analysis was developed in the BRISE-Vienna project for use in the building permission process based on digital building models. The BRISE-Vienna project used the software Solibri Office for code checking. Solibri provides an API based on the programming language Java to extend its functionality by customised rule templates. The escape route analysis method was implemented in Solibri Office using this API. The developed rule template is parametric, which is important for use in the building permission process. Building codes change over time. For example, the analysed OIB guidelines (building regulations) are revised after four years. The parametric design facilitates the maintenance of the checking rule because minor changes can be implemented without any

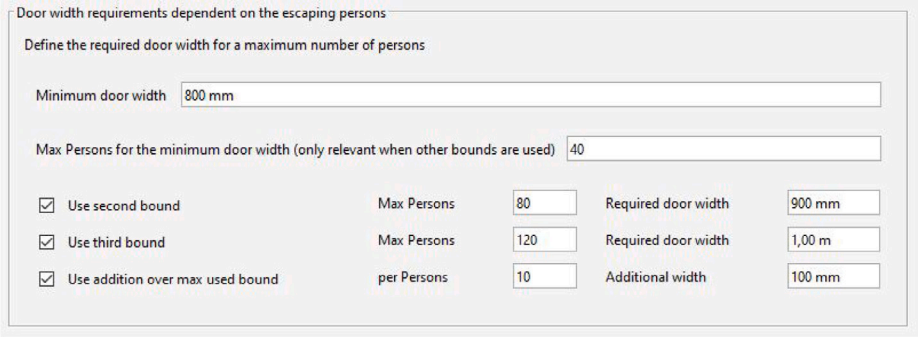
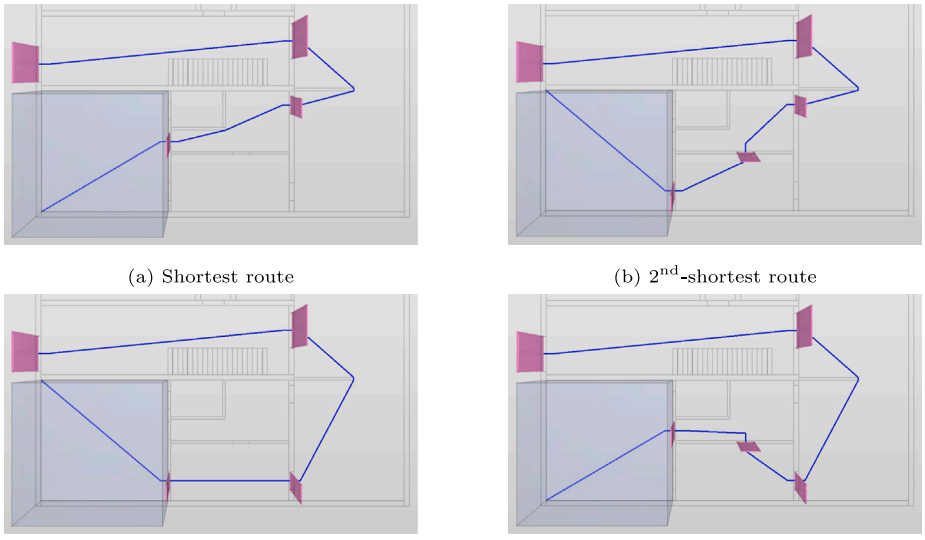


Fig. 10. Parametric input of the thresholds for the door width in the Vienna building code.



(a) Shortest route (b) 2nd-shortest route

(c) 3rd-shortest route (d) 4th-shortest route

Fig. 11. Generation of the four shortest routes from one starting room (in blue) in the escape route test model.

programming effort. It also allows the use of the rule template for other building authorities with slightly different regulations (e.g. different thresholds). Fig. 10 shows the parametric input of the thresholds for the door width depending on the number of people passing the object. In Vienna, two thresholds are set at 80 and 120 people. Above this, the requirements increase by 10 cm for every 10 people escaping through the door.

In this study, 17 test models were used to validate the developed escape route analysis method: a specific escape route test model and 16 real-world models. The escape route test model was a fictional five-story building that included test cases for special requirements relevant to the concept: starting doors and starting rooms with several different escape routes, interconnected spaces, an underground parking garage, and different dimensions of the elements along the routes. Sixteen real-world models are related to actual building submissions. Eleven of these were modelled based on the 2D plans of recent submissions. The other five were modelled for new submissions from designers who participated in the pilot phase of BRISE-Vienna. The models were developed using three different authoring software packages: Archicad (10 models), Revit (four models), and Allplan (three models). For their use in Solibri Office all models were exported in the IFC4 format. The escape route test model [62] and one of the real-world models [63] are publicly available in an open repository.

The escape route test model served to test the functionality of the two developed concepts: escape route generation and escape route validation. First, escape route generation is used by designers to automatically generate a pool of possible routes per starting location.

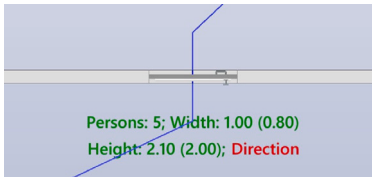


Fig. 12. Failure of a door in Solibri Office, including the number of people, width, person-dependent required width, height, required height, and opening direction.

Fig. 11 shows the possible routes from a starting room on the ground floor of the model. The starting room had two doors leading to passable rooms. As a result, two different starting points were defined for the room (the points furthest from each door). In general, designers should be provided with all valid routes from each starting location. For the BRISE-Vienna project, we implemented a functionality to show the generated routes with errors. Errors could concern, for example, the dimensions or opening directions of doors. This helps designers detect modelling errors. Fig. 11 shows this generation, including the failed routes. Doors must be opened in the direction of escape if the threshold for people using the door is exceeded. The 2nd-shortest and 4th-shortest (Figs. 11(b) and 11(d)) use the same second door, but pass it from different sides. Consequently, failure occurs because of the opening direction (Fig. 12). However, the four detected escape routes provide a pool of possible routes that designers can choose. Designers select a route by including a door sequence in the building model. Therefore,

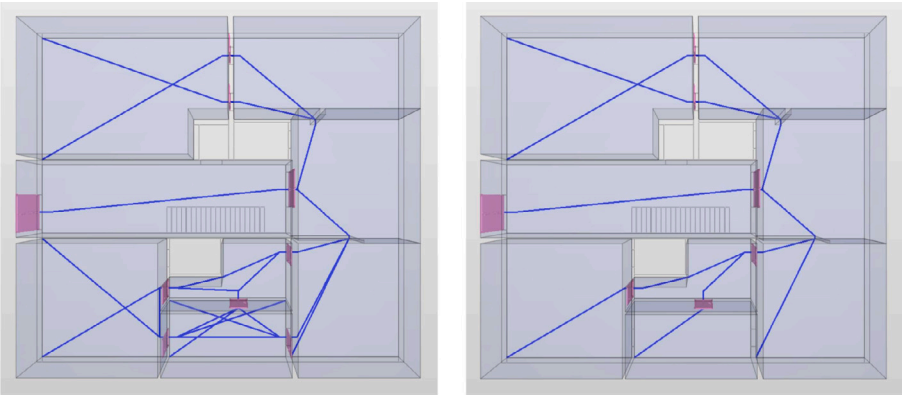


Fig. 13. Comparison between the escape route generation (left) and escape route validation (right) methods in the escape route test model.

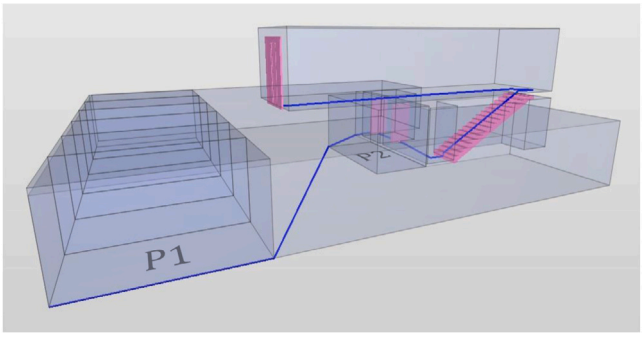


Fig. 14. Escape from an occupied parking space (P1) in the underground parking garage of the escape route test model (the small cuboids inside the parking garage are the parking spaces).

each door obtains a new property that refers to the next door on the escape route; for example, using a unique ID.

The second automated step was the escape route validation. The differences between escape route generation and escape route validation are shown in Fig. 13. The possible escape routes for the office rooms on the ground floor of the escape route test model are shown in Fig. 13 (left), and the selected routes (for each starting room) are shown in Fig. 13 (right). While all rooms at the bottom had only one selected route, those at the top left had two routes. The possibility of having more than one exit door is necessary for starting rooms with high occupancy. According to OIB guidelines, two doors are required for rooms with an occupancy of more than 120 people.

Another characteristic of escape route analysis is the escape from parking spaces inside parking garages. According to the building authority of Vienna, it must be assumed that all parking spaces are occupied and that someone is located at the most remote corner of the most remote parking space. Because all parking spaces are occupied, it is impossible to pass through them. An exception is the start parking space, because one needs to pass through it to leave it, but it is impossible to go diagonally across the parking space. Therefore, circulation at the start parking space is restricted to its border lines, and all other parking spaces are included as obstacles. The implementation results are presented in Fig. 14. The route started at the back corner of the most remote parking space (P1), followed its borderline, and ran around the parking space on the right side (P2) to reach the exit door. In conclusion, the route could start from an occupied parking space but avoided other parking spaces, thus fulfilling the requirements.

After ensuring the functionality of the developed checking rule, it was tested on 16 real-world models. Five of these models had serious modelling errors that prevented the generation and validation of escape routes. Examples of such errors include incorrect floor allocation of objects, intersections of space objects on different floors, and total absence of space objects. For nine of the remaining 11 models, escape routes were generated and validated in less than five seconds. In addition, there were two outliers at seven and 16 s. These models are related to large projects, with sizes exceeding the average. However, this duration is still short.

Fig. 15 shows 3D views and escape routes of four real-world models. The results of the published real-world model [63] (Fig. 16) are described as follows. This model included seven flats, an underground parking garage, and four storage rooms in the basement. For flats, the escape route started at the flat entrance door instead of the most remote point (in accordance with Austrian building regulations). All escape routes passed through a safe zone – the central staircase. This influenced the lengths of the escape routes because they are only considered to the safe zone. Hence, the lengths of the escape routes from the flats directly connected to the staircase were zero. Escape route generation detected one escape route for each starting position. Alternative routes were not found, because only one exit and one staircase were available. After adding the door sequences of the escape routes to the building model, escape route validation successfully reconstructed and verified the escape routes from the seven flats, the parking garage, and storage rooms (Fig. 16). The reconstructed routes coincided with those selected by the building authority agents as relevant escape routes.

The results demonstrate the functionality of the newly developed escape route analysis method. First, the combination of escape route generation and validation results in a high level of automation and still complies with Vienna's legal situation. The separation into two automated steps ensures that designers can predefine escape routes, and the building authority can check the given escape route network using the same checking rule. The approach also complies with requirements regarding the course and geometry of escape routes. Routes can start from flat entrance doors as well as the most remote points inside rooms, while also considering blocked parking spaces. The routes represented the shortest possible paths inside the rooms, as required by the building authority. This is ensured by the new navigation model that considers room nodes directly at the corners without a buffer distance. The connections between the nodes are based on a visibility check and the flexible-tube width check, which allow the route width to switch from one side of the route to the other. This functionality is essential when nodes are placed directly at the corners. Because one cannot move straight at a corner owing to the shape of the human body, the flexible tube checks the area around the corners to ensure that people can walk there. Furthermore, our approach includes geometric checks of elements along escape routes depending on the cumulative number of people passing them.

In addition to the proper representation of the building code, the duration of escape route analysis is an important evaluation criterion. Because of legal labour reasons, the authors did not obtain permission to evaluate the time expenditure of different tasks for building authority

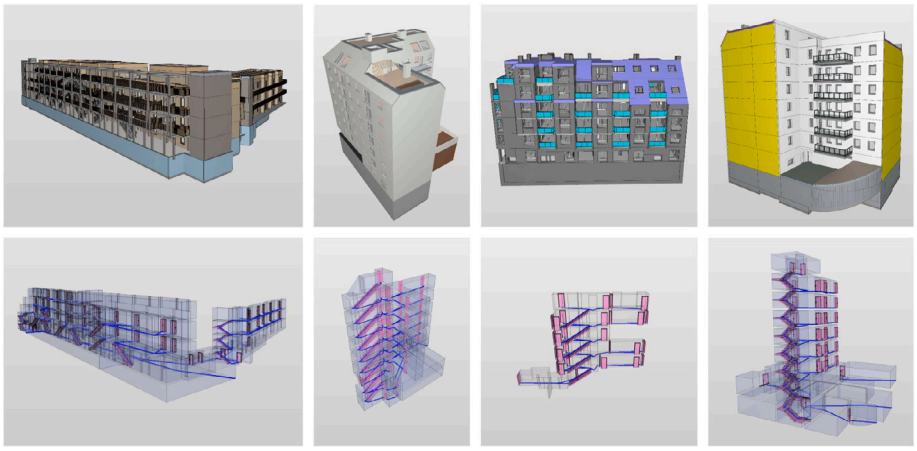


Fig. 15. Result of the escape route analysis for four real-world models. 3D-view (first row) and escape routes (second row) of the models.

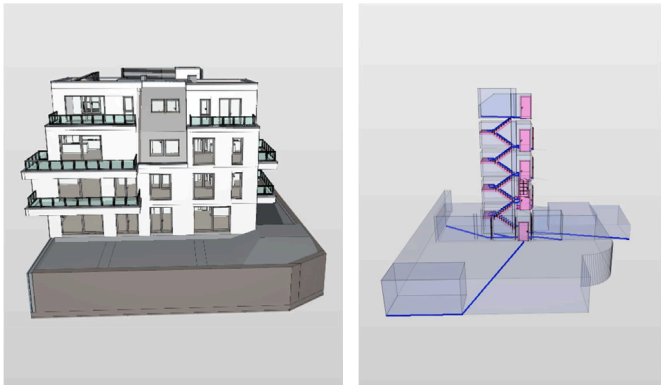


Fig. 16. Result of the escape route analysis for the published real-world model. 3D-view (left) and escape routes (right).

agents in detail. Therefore, no records on the duration of the current manual processes are available. Building authority agents classify escape route analysis as one of the most elaborate and time-consuming tasks. Therefore, the results of our verification phase, checking most models in less than five seconds with the longest check taking 16 s, indicated an improvement in time consumption.

Finally, the approach must be accepted by the relevant stakeholders for implementation in the BIM-based permission process. The placement of nodes directly attached to the geometry of the building model in combination with straight connections increases comprehensibility and manual reproducibility compared to a centreline-based approach. This is a significant advantage of the new approach in terms of implementation by a building authority. While escape route generation promises benefits for designers, manual selection of the final escape routes between escape route generation and escape route validation can be elaborate. The chosen approach for the Solibri implementation (to manually include the door sequences of the escape routes in the building model) takes additional time. However, the Vienna Building Authority and designers were both closely involved in the conception and validation of the checking rule; therefore, their needs were covered. Thus, all the prerequisites for practical implementation are provided.

6. Conclusion

In this study, we focused on the automation of escape route analysis as part of the BIM-based permission process developed for the BRISE-Vienna project. The potential of automation of code checking during this process is widely known. However, automation has not yet been established. The BRISE-Vienna project attempted to overcome the shortcomings of previous approaches by developing a complete BIM-based permission process using open standards.

To be implemented in the BRISE-Vienna process, the new escape route analysis method must comply with the legal situation in Vienna, provide results in an appropriate time, and be accepted by stakeholders of the process. The legal situation (in Vienna) requires a novel concept for escape route analysis consisting of two automated steps: escape route generation and escape route validation, with manual interference between them. Escape route generation is a tool used by designers to create a pool of possible escape routes from which the final escape route network can be selected. This manual process between the two automated steps ensures compliance with the provision that designers must predefine escape routes. Subsequently, the building authority checks this predefined network of escape routes through escape route validation. To comply with the required course and geometry of escape routes, we created a navigation model based on a visibility graph and enhanced the model by checking the route width using a flexible tube. The flexible tube represents the route width and can be switched from one side of the route to the other. After applying the algorithms proposed by Dijkstra [60] and Yen [61] to find the shortest routes, we checked the geometry of the elements along the escape routes depending on the cumulative number of people passing the elements. Thus, we implemented all the core functionalities of escape route analysis as part of a BIM-based building permission process and extended the functionality of previous approaches.

Furthermore, the test results showed an improvement in terms of time consumption over the manual checking process. The checking rule replaced the time-consuming and imprecise process of reconstructing and remeasuring escape routes drawn in 2D plans by automated validation in less than 16 s for each test model.

The close integration of the building authority and designers during the development process ensured that their needs were considered. However, there is still room for improvement in the designers' manual escape route selection. Enhancing the data transfer from escape route generation to escape route validation is essential for further research.

Future work will also include an extension of the flexible-tube width check. The current implementation is not applicable to convex corner points. Hence, it cannot be used to verify the width between the first and second nodes in the starting room. In our implementation, these connections were checked based only on visibility. Although we extended the functionality of the previous work, some special cases of the building code of Vienna, such as an allowed narrowing of the minimum room width by 10 cm for a maximum length of 1.20 m, were not included. This was decided together with the building authority to reduce the complexity of the checking rule and thus increase its maintainability. Another interesting topic for future work is the availability

of the generated escape routes. Currently, they exist only temporarily in a checking environment. Developing a permanently available escape route IFC model could enhance the work with digital building models. Finally, another important topic is the development of open-source checking software, particularly for the needs of building authorities. The authors of this paper are part of a European network for digital building permits, where this is a highly debated topic.

Declaration of competing interest

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

Data availability

Links to used BIM models are included in the paper.

Acknowledgements

Funding: This work was supported by Urban Innovative Actions [grant number UIA04-081] [21], an initiative of the European Union. The authors acknowledge TU Wien Bibliothek, Austria for financial support through its Open Access Funding Programme.

References

[1] C.C. Eichler, C. Schranz, T. Krischmann, H. Urban, BIMcert Handbook: Basic Knowledge openBIM. Edition 2023, Mironde-Verlag, 2023, http://dx.doi.org/10.34726/4162.

[2] R. Sacks, C.M. Eastman, G. Lee, P. Teicholz, BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facitily Managers, third ed., John Wiley & Sons, Inc, Hoboken, New Jersey, ISBN: 978-1-119-28754-4, 2018.

[3] S. Zhang, J. Teizer, J.-K. Lee, C.M. Eastman, M. Venugopal, Building Information Modeling (BIM) and Safety: Automatic Safety Checking of Construction Models and Schedules, Autom. Constr. 29 (2013) 183–195, http://dx.doi.org/10.1016/j.autcon.2012.05.006.

[4] K. Kim, Y.-C. Lee, Automated Generation of Daily Evacuation Paths in 4D BIM, Appl. Sci. 9 (9) (2019) 1789, http://dx.doi.org/10.3390/app9091789.

[5] E. Kamel, A.M. Memari, Review of BIM's application in energy simulation: Tools, issues, and solutions, Autom. Constr. 97 (2019) 164–180, http://dx.doi.org/10.1016/j.autcon.2018.11.008.

[6] A. Andriamamonjy, D. Saelens, R. Klein, A combined scientometric and conventional literature review to grasp the entire BIM knowledge and its integration with energy simulation, J. Build. Eng. 22 (2019) 513–527, http://dx.doi.org/10.1016/j.jobe.2018.12.021.

[7] S. Habibi, Role of BIM and energy simulation tools in designing zero-net energy homes, Constr. Innov. 22 (1) (2021) 101–119, http://dx.doi.org/10.1108/CI-12-2019-0143.

[8] T.H. Beach, J.-L. Hippolyte, Y. Rezgui, Towards the adoption of automated regulatory compliance checking in the built environment, Autom. Constr. 118 (2020) 103285, http://dx.doi.org/10.1016/j.autcon.2020.103285.

[9] C. Eastman, J.-M. Lee, Y.-S. Jeong, J.-K. Lee, Automatic rule-based checking of building designs, Autom. Constr. 18 (8) (2009) 1011–1033, http://dx.doi.org/10.1016/j.autcon.2009.07.002.

[10] C.S. Han, J.C. Kunz, K.H. Law, Client/Server Framework for On-Line Building Code Checking, J. Comput. Civ. Eng. 12 (4) (1998) 181–194, http://dx.doi.org/10.1061/(ASCE)0887-3801(1998)12:4(181).

[11] S. Macit İlal, H.M. Günaydın, Computer representation of building codes for automated compliance checking, Autom. Constr. 82 (2017) 43–58, http://dx.doi.org/10.1016/j.autcon.2017.06.018.

[12] M. Uhm, G. Lee, Y. Park, S. Kim, J. Jung, J.-K. Lee, Requirements for computational rule checking of requests for proposals (RFPs) for building designs in South Korea, Adv. Eng. Inform. 29 (3) (2015) 602–615, http://dx.doi.org/10.1016/j.aei.2015.05.006.

[13] N. Nisbet, J. Wix, D. Conover, The Future of Virtual Construction and Regulation Checking, in: P. Brandon, T. Kocatürk (Eds.), Virtual Futures for Design, Construction & Procurement, Blackwell Publishing Ltd, 2008, pp. 241–250, http://dx.doi.org/10.1002/9781444302349.ch17.

[14] J.-K. Lee, Building environment rule and analysis (BERA) language and its application for evaluating building circulation and spatial program (Ph.D. thesis), Georgia Institute of Technology, Atlanta, USA, 2011, https://smartech.gatech.edu/handle/1853/39482. (Accessed 31 March 2023).

[15] W. Solihin, J. Dimyadi, Y.-C. Lee, C. Eastman, R. Amor, Simplified schema queries for supporting BIM-based rule-checking applications, Autom. Constr. 117 (2020) 103248, http://dx.doi.org/10.1016/j.autcon.2020.103248.

[16] R. Amor, J. Dimyadi, The promise of automated compliance checking, Develop. Built Environ. 5 (2021) 100039, http://dx.doi.org/10.1016/j.dibe.2020.100039.

[17] S. Malsane, J. Matthews, S. Lockley, P.E. Love, D. Greenwood, Development of an object model for automated compliance checking, Autom. Constr. 49 (2015) 51–58, http://dx.doi.org/10.1016/j.autcon.2014.10.004.

[18] F. Noardo, D. Guler, J. Fauth, G. Malacarne, S. Mastrolembo Ventura, M. Azenha, P.-O. Olsson, L. Senger, Unveiling the actual progress of Digital Building Permit: Getting awareness through a critical state of the art review, Build. Environ. 213 (2022) 108854, http://dx.doi.org/10.1016/j.buildenv.2022.108854.

[19] J. Fauth, Ein handlungsorientiertes Entscheidungsmodell zur Feststellung der Genehmigungsfähigkeit von Bauvorhaben (An action-oriented decision model for determining the approvability of building projects) (Ph.D. thesis), Bauhaus-Universität Weimar, Weimar, Germany, 2021, http://dx.doi.org/10.25643/bauhaus-universitaet.4509.

[20] B.-H. Goh, E-Government for Construction: The Case of Singapore's CORENET Project, in: L.D. Xu, A.M. Tjoa, S.S. Chaudhry (Eds.), Research and Practical Issues of Enterprise Information Systems II, in: IFIP —The International Federation for Information Processing, vol. 254, Springer US, Boston, Massachusetts, 2007, pp. 327–336, http://dx.doi.org/10.1007/978-0-387-75902-9_34.

[21] UIA. – Urban Innovative Actions, BRISE-Vienna – Building Regulations Information for Submission Envolvement, 2019, https://www.uia-initiative.eu/en/uia-cities/vienna-call4. (Accessed 22 June 2022).

[22] T. Krischmann, H. Urban, C. Schranz, Entwicklung eines openBIM-Bewilligungsverfahrens (Development of an openBIM submission process), Bauingenieur 95 (9) (2020) 335–344, http://dx.doi.org/10.37544/0005-6650-2020-09-61.

[23] C. Schranz, H. Urban, A. Gerger, Potentials of Augmented Reality in a BIM based building submission process, J. Inf. Technol. Constr. 26 (2021) 441–457, http://dx.doi.org/10.36680/j.itcon.2021.024.

[24] A. Gerger, H. Urban, C. Schranz, Augmented Reality for Building Authorities: A Use Case Study in Austria, Buildings 13 (6) (2023) 1462, http://dx.doi.org/10.3390/buildings13061462.

[25] J.-K. Lee, C.M. Eastman, J. Lee, M. Kannala, Y.-S. Jeong, Computing walking distances within buildings using the universal circulation network, Environ. Plan. B: Plann. Des. 37 (4) (2010) 628–645, http://dx.doi.org/10.1068/b35124.

[26] M. Kannala, Escape Route Analysis Based on Building Information Models: Design and Implementation (MSc thesis), Department of Computer Science and Engineering, Helsinki University of Technology, Helsinki, Finland, 2005, http://mattikannala.fi/projects/masters-thesis/Escape-Route-Analysis-Based-on-Building-Information-Models-Design-and-Implementation.pdf. (Accessed 31 March 2023).

[27] J. Dimyadi, R. Amor, M. Spearpoint, Using BIM to Support Simulation of Compliant Building Evacuation, in: Proceedings of the 11th European Conference on Product and Process Modelling, ECPPM2016, CRC Press, Limassol, Cyprus, 2016, http://dx.doi.org/10.1201/9781315386904.

[28] Y.-H. Lin, Y.-S. Liu, G. Gao, X.-G. Han, C.-Y. Lai, M. Gu, The IFC-based path planning for 3D indoor spaces, Adv. Eng. Inform. 27 (2) (2013) 189–205, http://dx.doi.org/10.1016/j.aei.2012.10.001.

[29] Solibri Inc, Solibri Office, 2022, https://www.solibri.com/solibri-office. (Accessed 04 June 2022).

[30] Austrian Institute of Construction Engineering, OIB Guidelines, 2019, https://www.oib.or.at/en/oib-guidelines. (Accessed 14 June 2022).

[31] Austrian Institute of Construction Engineering, OIB Guideline 2: Safety in case of fire, 2019, https://www.oib.or.at/en/node/5699261. (Accessed 14 June 2022).

[32] Austrian Institute of Construction Engineering, OIB Guideline 2.2: Safety in case of fire in garages, roofed parking spaces and multi-storey car parks, 2019, https://www.oib.or.at/en/node/5699264. (Accessed 14 June 2022).

[33] Austrian Institute of Construction Engineering, OIB Guideline 4: Safety in use and accessibility, 2019, https://www.oib.or.at/en/node/5699267. (Accessed 14 June 2022).

[34] S. Taneja, B. Akinci, J.H. Garrett, L. Soibelman, Algorithms for automated generation of navigation models from building information models to support indoor map-matching, Autom. Constr. 61 (2016) 24–41, http://dx.doi.org/10.1016/j.autcon.2015.09.010.

[35] M. Fu, R. Liu, B. Qi, R.R. Issa, Generating straight skeleton-based navigation networks with Industry Foundation Classes for indoor way-finding, Autom. Constr. 112 (2020) 103057, http://dx.doi.org/10.1016/j.autcon.2019.103057.

[36] M. Goetz, A. Zipf, Formal definition of a user-adaptive and length-optimal routing graph for complex indoor environments, Geo-spatial Inf. Sci. 14 (2) (2011) 119–128, http://dx.doi.org/10.1007/s11806-011-0474-3.

[37] W.Y. Lin, P.H. Lin, Intelligent generation of indoor topology (i-GIT) for human indoor pathfinding based on IFC models and 3D GIS technology, Autom. Constr. 94 (2018) 340–359, http://dx.doi.org/10.1016/j.autcon.2018.07.016.

[38] W. Yuan, M. Schneider, Supporting 3D route planning in indoor space based on the LEGO representation, in: ISA '10: Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness, in: ISA '10, Association for Computing Machinery, New York, USA, 2010, pp. 16–23, http://dx.doi.org/10.1145/1865885.1865890.

[39] L. Yang, M. Worboys, Generation of navigation graphs for indoor space, Int. J. Geogr. Inf. Sci. 29 (10) (2015) 1737–1756, http://dx.doi.org/10.1080/13658816.2015.1041141.

[40] L. Liu, B. Li, S. Zlatanova, P. van Oosterom, Indoor navigation supported by the Industry Foundation Classes (IFC): A survey, Autom. Constr. 121 (2021) 103436, http://dx.doi.org/10.1016/j.autcon.2020.103436.

[41] M. Höcker, V. Berkhahn, A. Kneidl, A. Borrmann, W. Klein, Graph-based approaches for simulating pedestrian dynamics in building models, in: EWork and EBusiness in Architecture, Engineering and Construction – Proceedings of the European Conference on Product and Process Modelling, CRC Press, 2010, pp. 389–394, http://dx.doi.org/10.1201/b10527-65.

[42] A. Borrmann, A. Kneidl, G. Köster, S. Ruzika, M. Thiemann, Bidirectional coupling of macroscopic and microscopic pedestrian evacuation models, Saf. Sci. 50 (8) (2012) 1695–1703, http://dx.doi.org/10.1016/j.ssci.2011.12.021.

[43] A. Kneidl, A. Borrmann, D. Hartmann, Generation and use of sparse navigation graphs for microscopic pedestrian simulation models, Adv. Eng. Inform. 26 (4) (2012) 669–680, http://dx.doi.org/10.1016/j.aei.2012.03.006.

[44] A. Kneidl, D. Hartmann, A. Borrmann, A hybrid multi-scale approach for simulation of pedestrian dynamics, Transp. Res. C 37 (2013) 223–237, http://dx.doi.org/10.1016/j.trc.2013.03.005.

[45] A. Kneidl, A. Borrmann, D. Hartmann, Generating sparse navigation graphs for microscopic pedestrian simulation models, in: EG-ICE 2011, European Group for Intelligent Computing in Engineering, ISBN: 978-90-365-3216-7, 2014.

[46] M. Xu, I. Hijazi, A. Mebarki, R.E. Meouche, M. Abune'meh, Indoor guided evacuation: TIN for graph generation and crowd evacuation, Geomat. Nat. Hazards Risk 7 (sup1) (2016) 47–56, http://dx.doi.org/10.1080/19475705.2016.1181343.

[47] A.Y. Chen, J.C. Chu, TDVRP and BIM Integrated Approach for In-Building Emergency Rescue Routing, J. Comput. Civ. Eng. 30 (5) (2016) C4015003, http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000522.

[48] M. Fu, R. Liu, BIM-based automated determination of exit sign direction for intelligent building sign systems, Autom. Constr. 120 (2020) 103353, http://dx.doi.org/10.1016/j.autcon.2020.103353.

[49] A. Turner, M. Doxa, D. O'Sullivan, A. Penn, From Isovists to Visibility Graphs: A Methodology for the Analysis of Architectural Space, Environ. Plan. B: Plann. Des. 28 (1) (2001) 103–121, http://dx.doi.org/10.1068/b2684.

[50] M. Duckham, L. Kulik, "Simplest" Paths: Automated Route Selection for Navigation, in: W. Kuhn, M.F. Worboys, S. Timpf (Eds.), Spatial Information Theory. Foundations of Geographic Information Science. COSIT 2003, in: Lecture Notes in Computer Science, vol. 2825, Springer, Berlin, Heidelberg, 2003, pp. 169–185, http://dx.doi.org/10.1007/978-3-540-39923-0_12.

[51] P. Boguslawski, L. Mahdjoubi, V. Zverovich, F. Fadli, Automated construction of variable density navigable networks in a 3D indoor environment for emergency response, Autom. Constr. 72 (2016) 115–128, http://dx.doi.org/10.1016/j.autcon.2016.08.041.

[52] J. Wang, G. Wei, X. Dong, A dynamic fire escape path planning method with BIM, J. Ambient Intell. Humaniz. Comput. 12 (11) (2021) 10253–10265, http://dx.doi.org/10.1007/s12652-020-02794-2.

[53] A. Ammar, H. Nassereddine, N. AbdulBaky, A. AbouKansour, J. Tannoury, H. Urban, C. Schranz, Digital Twins in the Construction Industry: A Perspective of Practitioners and Building Authority, Front. Built Environ. 8 (2022) 834671, http://dx.doi.org/10.3389/fbuil.2022.834671.

[54] H. Urban, C. Schranz, T. Krischmann, H. Asmera, B. Pinter, Einsatz von openBIM und KI im Bewilligungsverfahren der Stadt Wien (Use of openBIM and AI in the approval process of the City of Vienna), ÖIAZ – Österreichische Ingenieur- und Architektenzeitschrift 166 (2021) 1–9, http://hdl.handle.net/20.500.12708/137851. (Accessed 31 March 2023).

[55] International Organization for Standardization (Ed.), ISO 16739-1:2018 Industry Foundation Classes (IFC) for Data Sharing in the Construction and Facility Management Industries — Part 1: Data Schema, Genf, Switzerland, 2018, https://www.iso.org/standard/70303.html. (Accessed 31 March 2023).

[56] City of Vienna, Bauordnung für Wien (Vienna Building Code), 2022, https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=LrW&Gesetzesnummer=20000006. (Accessed 10 October 2022).

[57] City of Vienna, Wiener Garagengesetz 2008 (Vienna Garage Law 2008), 2022, https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=LrW&Gesetzesnummer=20000052. (Accessed 10 October 2022).

[58] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, fourth ed., MIT Press, Cambridge, Massachusetts, ISBN: 978-0-262-36750-9, 2022.

[59] Z. Abd Algfoor, M.S. Sunar, H. Kolivand, A Comprehensive Study on Pathfinding Techniques for Robotics and Video Games, Int. J. Comput. Games Technol. 2015 (2015) 736138, http://dx.doi.org/10.1155/2015/736138.

[60] E.W. Dijkstra, A Note on Two Problems in Connexion with Graphs, Numer. Math. 1 (1) (1959) 269–271, http://dx.doi.org/10.1007/BF01386390.

[61] J.Y. Yen, Finding the k shortest loopless paths in a network, Manage. Sci. 17 (11) (1971) 712–716, http://dx.doi.org/10.1287/mnsc.17.11.712.

[62] S. Fischer, C. Schranz, H. Urban, D. Pfeiffer, Custom Escape Route Test Model in IFC format, TU Wien, 2023, http://dx.doi.org/10.48436/ra5g9-tbb65, [Data set].

[63] S. Fischer, C. Schranz, H. Urban, D. Pfeiffer, Real-World Escape Route Test Model in IFC format, TU Wien, 2023, http://dx.doi.org/10.48436/2heb4-xqb83, [Data set].